PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

Assignee 1

Complete this section if assignee information, including non-applicant assignee information, is desired to be included on the patent application publication. An assignee-applicant identified in the "Applicant Information" section will appear on the patent application publication as an applicant. For an assignee-applicant, complete this section only if identification as an assignee is also desired on the patent application publication.

Remove

If the Assignee or Non-Applicant Assignee is an Organization check here. ☐

Prefix	**Given Name**	Middle Name	**Family Name**	Suffix

Mailing Address Information For Assignee including Non-Applicant Assignee:

Address 1			
Address 2			
City		**State/Province**	
Country i		Postal Code	
Phone Number		Fax Number	
Email Address			

Additional Assignee or Non-Applicant Assignee Data may be generated within this form by selecting the Add button. Add

Signature:

Remove

NOTE: This Application Data Sheet must be signed in accordance with 37 CFR 1.33(b). **However, if this Application Data Sheet is submitted with the INITIAL filing of the application and either box A or B is not checked in subsection 2 of the "Authorization or Opt-Out of Authorization to Permit Access" section, then this form must also be signed in accordance with 37 CFR 1.14(c).**

This Application Data Sheet **must** be signed by a patent practitioner if one or more of the applicants is a **juristic entity** (e.g., corporation or association). If the applicant is two or more joint inventors, this form must be signed by a patent practitioner, **all** joint inventors who are the applicant, or one or more joint inventor-applicants who have been given power of attorney (e.g., see USPTO Form PTO/AIA/81) on behalf of **all** joint inventor-applicants.

See 37 CFR 1.4(d) for the manner of making signatures and certifications.

Signature	/Jasmin Cosic/			Date (YYYY-MM-DD)	2016-11-02
First Name	Jasmin	Last Name	Cosic	Registration Number	

Additional Signature may be generated within this form by selecting the Add button. Add

PTO/AIA/14 (11-15)
Approved for use through 04/30/2017. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

Application Data Sheet 37 CFR 1.76	Attorney Docket Number	
	Application Number	
Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION	

This collection of information is required by 37 CFR 1.76. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 23 minutes to complete, including gathering, preparing, and submitting the completed application data sheet form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C. 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether the Freedom of Information Act requires disclosure of these records.

2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.

3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.

4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).

5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent CooperationTreaty.

6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).

7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.

8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspections or an issued patent.

9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

PTO/AIA/01 (06-12)
Approved for use through 01/31/2014. OMB 0651-0032
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

DECLARATION (37 CFR 1.63) FOR UTILITY OR DESIGN APPLICATION USING AN APPLICATION DATA SHEET (37 CFR 1.76)

Title of Invention	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

As the below named inventor, I hereby declare that:

This declaration is directed to:

- [■] The attached application, or
- [ ] United States application or PCT international application number _____

  filed on _____.

The above-identified application was made or authorized to be made by me.

I believe that I am the original inventor or an original joint inventor of a claimed invention in the application.

I hereby acknowledge that any willful false statement made in this declaration is punishable under 18 U.S.C. 1001 by fine or imprisonment of not more than five (5) years, or both.

WARNING:

Petitioner/applicant is cautioned to avoid submitting personal information in documents filed in a patent application that may contribute to identity theft. Personal information such as social security numbers, bank account numbers, or credit card numbers (other than a check or credit card authorization form PTO-2038 submitted for payment purposes) is never required by the USPTO to support a petition or an application. If this type of personal information is included in documents submitted to the USPTO, petitioners/applicants should consider redacting such personal information from the documents before submitting them to the USPTO. Petitioner/applicant is advised that the record of a patent application is available to the public after publication of the application (unless a non-publication request in compliance with 37 CFR 1.213(a) is made in the application) or issuance of a patent. Furthermore, the record from an abandoned application may also be available to the public if the application is referenced in a published application or an issued patent (see 37 CFR 1.14). Checks and credit card authorization forms PTO-2038 submitted for payment purposes are not retained in the application file and therefore are not publicly available.

LEGAL NAME OF INVENTOR

Inventor: Jasmin Cosic Date (Optional) _____

Signature: _____

Note: An application data sheet (PTO/SB/14 or equivalent), including naming the entire inventive entity, must accompany this form or must have been previously filed. Use an additional PTO/AIA/01 form for each additional inventor.

This collection of information is required by 35 U.S.C. 115 and 37 CFR 1.63. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to take 1 minute to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION

FIELD

The disclosure generally relates to computing enabled devices and/or systems. The disclosure includes devices, apparatuses, systems, and related methods for providing advanced learning, anticipating, decision making, automation, and/or other functionalities.

COPYRIGHT NOTICE

A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

BACKGROUND

Devices or systems commonly operate by receiving a user's operating directions in various circumstances. Instructions are then executed to effect the operation of a device or system based on user's operating directions. Hence, devices or systems rely on the user to direct their behaviors. Commonly employed device or system operating techniques lack a way to learn operation of a device or system and enable autonomous operation of a device or system.

SUMMARY

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises a processor circuit configured to execute instruction sets for operating a device. The system may further include a memory unit configured to store data. The system may further include a sensor configured to detect objects. The system may further include an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit,

Autonomous Ex. 2004 p. 305 of 510
Tesla v. Autonomous
IPR2024-00745

wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some embodiments, at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of, operating on, or coupled to the device. In further embodiments, the device includes one or more devices. In further embodiments, the device includes a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, an appliance, a toy, a robot, a ground vehicle, an aerial vehicle, an aquatic vehicle, a computer, a smartphone, a control device, or a computing enabled device. In further embodiments, the processor circuit includes one or more processor circuits. In further embodiments, the processor circuit includes a logic circuit. The logic circuit may include a microcontroller. The one or more instruction sets may include one or more inputs into or one or more outputs from the logic circuit.

In certain embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include inputs into the logic circuit, and executing instruction sets for operating the device includes performing logic operations on the inputs into the logic circuit and producing outputs for operating the device. The logic circuit includes a microcontroller.

In some embodiments, the processor circuit includes a logic circuit, the instruction sets for operating the device include outputs from the logic circuit for operating the device, and executing instruction sets for operating the device includes performing logic operations on inputs into the logic circuit and producing the outputs from the logic circuit for operating the device.

In certain embodiments, the memory unit includes one or more memory units. In further embodiments, the memory unit resides on a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. The remote computing device or the remote computing system may include a server, a cloud, a computing device, or a computing system accessible over the network or the interface.

In some embodiments, the sensor includes one or more sensors. In further embodiments, the sensor includes a camera, a microphone, a lidar, a radar, a sonar, or a detector. In further embodiments, the sensor is part of a remote device. In further embodiments, the sensor is configured to detect objects in the device's surrounding.

In certain embodiments, the artificial intelligence unit is coupled to the sensor. In further embodiments, the artificial intelligence unit is coupled to the memory unit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to the processor circuit. In further embodiments, the system further comprises: an additional processor circuit, wherein the artificial intelligence unit is part of, operating on, or coupled to the additional processor circuit. In further embodiments, the artificial intelligence unit is part of, operating on, or coupled to a remote computing device or a remote computing system, the remote computing device or the remote computing system coupled to the processor circuit via a network or an interface. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system attachable to the device. In further embodiments, the artificial intelligence unit is attachable to an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial

Autonomous Ex. 2004 p. 306 of 510
Tesla v. Autonomous
IPR2024-00745

intelligence unit includes a circuit, a computing apparatus, or a computing system built into the processor circuit. In further embodiments, the artificial intelligence unit includes a circuit, a computing apparatus, or a computing system built into the device. In further embodiments, the artificial intelligence unit is built into an application for operating the device, the application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of an application running on the processor circuit. In further embodiments, the artificial intelligence unit is provided as a feature of the device. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to the processor circuit. In further embodiments, the artificial intelligence unit is further configured to: take control from, share control with, or release control to an application or an object of the application, the application running on the processor circuit.

In some embodiments, the first collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the new collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the first collection of object representations includes a stream of collections of object representations. In further embodiments, the new collection of object representations includes a stream of collections of object representations. In further embodiments, the first or the new collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in the device's surrounding. In further embodiments, the first or the new collection of object representations includes one or more representations of objects in a remote device's surrounding. In further embodiments, an object representation of the one or more object representations includes one or more object properties. In further embodiments, the first or the new collection of object representations includes one or more object properties. In further embodiments, the first collection of object representations includes a comparative collection of object representations whose at least one portion can be used for comparisons with at least one portion of collections of object representations subsequent to the first collection of object representations, the collections of object representations subsequent to the first collection of object representations comprising the new collection of object representations. In further embodiments, the first collection of object representations includes a comparative collection of object representations that can be used for comparisons with the new collection of object representations. In further embodiments, the new collection of object representations includes an anticipatory collection of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed at a time of generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed prior to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations. In further embodiments, the one or more instruction

Autonomous Ex. 2004 p. 307 of 510
Tesla v. Autonomous
IPR2024-00745

sets that temporally correspond to the first collection of object representations include one or more instruction sets executed subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first collection of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first collection of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first collection of object representations or a threshold period of time subsequent to generating the first collection of object representations.

In some embodiments, the first one or more instruction sets for operating the device are executed by the processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of the processor circuit. In further embodiments, the first one or more instruction sets for operating the device include at least one of: a command, a keyword, a symbol, an instruction, an operator, a variable, a value, an object, a data structure, a function, a parameter, a state, a signal, an input, an output, a character, a digit, or a reference thereto. In further embodiments, the first one or more instruction sets for operating the device include a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the first one or more instruction sets for operating the device include one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the processor circuit includes a logic circuit. The first one or more instruction sets for operating the device may include one or more inputs into a logic circuit. The first one or more instruction sets for operating the device may include one or more outputs from a logic circuit.

In certain embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes obtaining the first one or more instruction sets for operating the device from the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a register or an element of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of

Autonomous Ex. 2004 p. 308 of 510
Tesla v. Autonomous
IPR2024-00745

processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the processor circuit includes a logic circuit, and wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, a structured query language (SQL) code, or a machine code. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of the processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of the processor circuit or tracing, profiling, or instrumentation of a component of the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application, the application running on the processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the

Autonomous Ex. 2004 p. 309 of 510
Tesla v. Autonomous
IPR2024-00745

receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of one or more of code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a user input. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) tracing, profiling, or instrumentation. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing at least one of: a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, a logging tool, or an independent tool for obtaining instruction sets. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing an assembly language. In further embodiments, the receiving the first one or more instruction sets for operating the device includes utilizing a branch or a jump. In further embodiments, the receiving the first one or more instruction sets for operating the device includes a branch tracing or a simulation tracing.

In further embodiments, the system further comprises: an interface configured to receive instruction sets, wherein the first one or more instruction sets for operating the device are received by the interface. The interface may include an acquisition interface.

In some embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first collection of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first collection of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first collection of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

Autonomous Ex. 2004 p. 310 of 510
Tesla v. Autonomous
IPR2024-00745

correlated with the first one or more instruction sets for operating the device into the memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each of the plurality of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the at least one portion of the new collection of object representations include at least one object representation or at least one object property of the new collection of object representations. In further embodiments, the at least one portion of the first collection of object representations include at least one object representation or at least one object property of the first collection of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one object representation from the new collection of object representations with at least one object representation from the first collection of object representations. In further embodiments, the comparing at least one object representation from the new collection of object representations with at least one object representation from the first

Autonomous Ex. 2004 p. 311 of 510
Tesla v. Autonomous
IPR2024-00745

collection of object representations includes comparing at least one object property of the at least one object representation from the new collection of object representations with at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between one or more portions of the new collection of object representations and one or more portions of the first collection of object representations. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining a substantial similarity between at least one portion of the new collection of object representations and at least one portion of the first collection of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new collection of object representations and the at least one portion of the first collection of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new collection of object representations and portions of the first collection of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new collection of object representations and from the first collection of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new collection of object representations and from the first collection of object representations may be determined factoring in at least one of: an association of an object property with an object representation, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, the determining that there is at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there

Autonomous Ex. 2004 p. 312 of 510
Tesla v. Autonomous
IPR2024-00745

is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations. The determining that there is at least a partial match between at least one object representation from the new collection of object representations and at least one object representation from the first collection of object representations includes determining that there is at least a partial match between at least one object property of the at least one object representation from the new collection of object representations and at least one object property of the at least one object representation from the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In some embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit

Autonomous Ex. 2004 p. 313 of 510
Tesla v. Autonomous
IPR2024-00745

includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit.

In some embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying the application.

In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a

Autonomous Ex. 2004 p. 314 of 510
Tesla v. Autonomous
IPR2024-00745

storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or

Autonomous Ex. 2004 p. 315 of 510
Tesla v. Autonomous
IPR2024-00745

more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations include one or more operations with or by a computing enabled device. In further embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprising: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system of further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first collection of object representations correlated with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include correlating the first collection of object representations with the at least one extra information. The learning the first collection of object representations correlated with at least one extra information may include storing the first collection of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first

Autonomous Ex. 2004 p. 316 of 510
Tesla v. Autonomous
IPR2024-00745

collection of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include comparing an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations may include determining that a similarity between an extra information correlated with the new collection of object representations and an extra information correlated with the first collection of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first collection of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

Autonomous Ex. 2004 p. 317 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations responsive to a user confirmation. In further embodiments, the fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first collection of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second collection of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, the second collection of object representations includes a stream of collections of object representations. In further embodiments, the second collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or be associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device. The updating the connection between the first collection of object representations correlated with the first one or more instruction sets for operating the device and the second collection of object representations correlated with the second one or more instruction sets for operating the device may include updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device includes storing the second collection of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. The knowledgebase may be stored in the memory unit. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include creating a connection

Autonomous Ex. 2004 p. 318 of 510
Tesla v. Autonomous
IPR2024-00745

between the first node and the second node. The learning the first collection of object representations correlated with the first one or more instruction sets for operating the device and the learning the second collection of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first node and the second node. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first collection of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second collection of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further include: receiving a first one or more instruction sets for operating a device. The operations may further include: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further include: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further include: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further include: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

Autonomous Ex. 2004 p. 319 of 510
Tesla v. Autonomous
IPR2024-00745

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further include: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further include: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further include: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further include: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further include: (f) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further include: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

In certain embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In some embodiments, the first one or more instruction sets for operating the device are executed by a processor circuit. In further embodiments, the first one or more instruction sets for operating the device are part of an application for operating the device. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into or one or more outputs from a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include values or states of one or more registers or elements of a processor circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more inputs into a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more outputs from a logic circuit. In further embodiments, the first one or more instruction sets for operating the device include one or more instruction sets for operating an application or an object of the application.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device as they are executed by a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first

Autonomous Ex. 2004 p. 320 of 510
Tesla v. Autonomous
IPR2024-00745

one or more instruction sets for operating the device from a register or an element of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from at least one of: the memory unit, the device, a virtual machine, a runtime engine, a hard drive, a storage device, a peripheral device, a network connected device, or a user. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a plurality of processor circuits, applications, memory units, devices, virtual machines, runtime engines, hard drives, storage devices, peripheral devices, network connected devices, or users.

In some embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from a logic circuit. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving the first one or more instruction sets for operating the device from an element of the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more inputs into the logic circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device from the logic circuit includes receiving one or more outputs from the logic circuit.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application including instruction sets for operating the device. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an element that is part of, operating on, or coupled to a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a register of a processor circuit, the memory unit, a storage, or a repository where the first one or more instruction sets for operating the device are stored. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit, the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a processing element. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of a processor circuit or tracing, profiling, or instrumentation of a component of a processor circuit. In further embodiments, the receiving the first one or more instruction sets for operating the device includes at least one of: tracing, profiling, or instrumentation of an application or an object of the application. In further embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device by an interface. The interface may include an acquisition interface.

In some embodiments, the learning the first collection of object representations correlated with the first one or more instruction sets for operating the device includes storing the first collection of object representations

Autonomous Ex. 2004 p. 321 of 510
Tesla v. Autonomous
IPR2024-00745

correlated with the first one or more instruction sets for operating the device into a memory unit, the first collection of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first collection of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first collection of object representations into an element of the logic circuit. In further embodiments, the executing, by

Autonomous Ex. 2004 p. 322 of 510
Tesla v. Autonomous
IPR2024-00745

the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first collection of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes modifying one or more code segments, lines of

Autonomous Ex. 2004 p. 323 of 510
Tesla v. Autonomous
IPR2024-00745

code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first collection of object representations includes executing the first one or more instruction sets for operating the device correlated with the first collection of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first collection of object representations correlated with the at least one extra information.

Autonomous Ex. 2004 p. 324 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first collection of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second collection of object representations, the second collection of object representations including one or more representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second collection of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first collection of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least

Autonomous Ex. 2004 p. 325 of 510
Tesla v. Autonomous
IPR2024-00745

a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a first processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the first processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for

Autonomous Ex. 2004 p. 326 of 510
Tesla v. Autonomous
IPR2024-00745

operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by a sensor. The

Autonomous Ex. 2004 p. 327 of 510
Tesla v. Autonomous
IPR2024-00745

operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of collections of object representations correlated with one or more instruction sets for operating the device including a first collection of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first collection of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the

Autonomous Ex. 2004 p. 328 of 510
Tesla v. Autonomous
IPR2024-00745

device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In certain embodiments, each collection of object representations includes one or more representations of objects detected by the sensor at a time. In further embodiments, each collection of object representations includes one or more of object representations. In further embodiments, each collection of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the first stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the new stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in the device's surrounding. In further embodiments, the first or the new stream of collections of object representations includes one or more collections of representations of objects in a remote device's surrounding. In further embodiments, an object representation of a stream of collections of object representations includes one or more object properties. In further embodiments, the first or the new stream of collections of object representations includes one or more object properties. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations whose at least one portion can be used for comparisons with at least one portion of streams of collections of object representations subsequent to the first stream of collections of object representations, the streams of collections of object representations subsequent to the first stream of collections of object representations comprising the new stream of collections of object representations. In further embodiments, the first stream of collections of object representations includes a comparative stream of collections of object representations that can be used for comparisons with the new stream of collections of object representations. In further embodiments, the new stream of collections of object representations includes an anticipatory stream of collections of object representations whose correlated one or more instruction sets can be used for anticipation of one or more instruction sets to be executed by the processor circuit.

In some embodiments, the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed at a time of generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed prior to generating

Autonomous Ex. 2004 p. 329 of 510
Tesla v. Autonomous
IPR2024-00745

the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time subsequent to generating the first stream of collections of object representations. In further embodiments, the one or more instruction sets that temporally correspond to the first stream of collections of object representations include one or more instruction sets executed within a threshold period of time prior to generating the first stream of collections of object representations or a threshold period of time subsequent to generating the first stream of collections of object representations.

In certain embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are included in a neuron, a node, a vertex, or an element of a knowledgebase. In further embodiments, the knowledgebase includes a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, some of the neurons, nodes, vertices, or elements are interconnected. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are structured into a knowledge cell. In further embodiments, the knowledge cell is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes generating a knowledge cell, the knowledge cell comprising the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. In further embodiments, the correlating the first stream of collections of object representations with the first one or more instruction sets for operating the device includes structuring a knowledge of how the device operated in a circumstance. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes learning a user's knowledge, style, or methodology of operating the device in a circumstance.

In some embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or

Autonomous Ex. 2004 p. 330 of 510
Tesla v. Autonomous
IPR2024-00745

more instruction sets for operating the device stored in the memory unit. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are organized into a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledgebase, a knowledge structure, or a data structure. In further embodiments, each stream of collections of object representations correlated with one or more instruction sets for operating the device of the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device is included in a neuron, a node, a vertex, or an element of a knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. Some of the neurons, nodes, vertices, or elements may be interconnected. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include a user's knowledge, style, or methodology of operating the device in circumstances. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device are stored on a remote computing device or a remote computing system. In further embodiments, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device include an artificial intelligence system for knowledge structuring, storing, or representation. The artificial intelligence system for knowledge structuring, storing, or representation may include at least one of: a deep learning system, a supervised learning system, an unsupervised learning system, a neural network, a search-based system, an optimization-based system, a logic-based system, a fuzzy logic-based system, a tree-based system, a graph-based system, a hierarchical system, a symbolic system, a sub-symbolic system, an evolutionary system, a genetic system, a multi-agent system, a deterministic system, a probabilistic system, or a statistical system.

In certain embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one portion of the new stream of collections of object representations with at least one portion of the first stream of collections of object representations. In further embodiments, the at least one portion of the new stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the new stream of collections of object representations. In further embodiments, the at least one portion of the first stream of collections of object representations include at least one collection of object representations, at least one object representation, or at least one object property of the first stream of collections of object representations. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations. In further

Autonomous Ex. 2004 p. 331 of 510
Tesla v. Autonomous
IPR2024-00745

embodiments, the comparing at least one collection of object representations from the new stream of collections of object representations with at least one collection of object representations from the first stream of collections of object representations includes comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The comparing at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include comparing at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations with at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

In some embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between one or more portions of the new stream of collections of object representations and one or more portions of the first stream of collections of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations exceeds a similarity threshold. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining a substantial similarity between at least one portion of the new stream of collections of object representations and at least one portion of the first stream of collections of object representations. The substantial similarity may be achieved when a similarity between the at least one portion of the new stream of collections of object representations and the at least one portion of the first stream of collections of object representations exceeds a similarity threshold. The substantial similarity may be achieved when a number or a percentage of matching or partially matching portions of the new stream of collections of object representations and portions of the first stream of collections of object representations exceeds a threshold number or threshold percentage. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching collections of object representations from the new stream of collections of object representations and from the first stream of collections of object representations

Autonomous Ex. 2004 p. 332 of 510
Tesla v. Autonomous
IPR2024-00745

may be determined factoring in at least one of: an importance of a collection of object representations, an order of a collection of object representations, a threshold for a similarity in a collection of object representations, or a threshold for a difference in a collection of object representations. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object representations from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object representation with a collection of object representations, a type of an object representation, an importance of an object representation, a threshold for a similarity in an object representation, or a threshold for a difference in an object representation. In further embodiments, the determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that a number or a percentage of matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations exceeds a threshold number or threshold percentage. The matching or partially matching object properties from the new stream of collections of object representations and from the first stream of collections of object representations may be determined factoring in at least one of: an association of an object property with an object representation, an association of an object property with a collection of object representations, a category of an object property, an importance of an object property, a threshold for a similarity in an object property, or a threshold for a difference in an object property. In further embodiments, determining that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one collection of object representations from the new stream of collections of object representations and at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations. The determining that there is at least a partial match between at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object representation of the at least one collection of object representations from the first stream of collections of object representations may include determining that there is at least a partial match between at least one object property of the at least one object representation of the at least one collection of object representations from the new stream of collections of object representations and at least one object property of the at least one object representation of the at least one collection of object representations from the first stream of collections of object representations.

Autonomous Ex. 2004 p. 333 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to the processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to the processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to the processor circuit.

In certain embodiments, the processor circuit includes a logic circuit, and wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device

Autonomous Ex. 2004 p. 334 of 510
Tesla v. Autonomous
IPR2024-00745

correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the logic circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes causing an application for operating the device to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application including instruction sets for operating the device, the application running on the processor circuit, wherein the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying the application.

In some embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the application running on the processor circuit, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of the processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of

Autonomous Ex. 2004 p. 335 of 510
Tesla v. Autonomous
IPR2024-00745

object representations includes modifying one or more instruction sets for operating an application or an object of the application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of the processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object

Autonomous Ex. 2004 p. 336 of 510
Tesla v. Autonomous
IPR2024-00745

representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application, the application running on the processor circuit. In further embodiments, the causing the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application, the application running on the processor circuit. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an interface configured to cause execution of instruction sets, wherein the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is caused by the interface. The interface may include a modification interface.

In some embodiments, the performing the one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance.

In certain embodiments, the system further comprises: an application running on the processor circuit.

In some embodiments, the instruction sets for operating the device are part of an application for operating the device, the application running on the processor circuit.

In certain embodiments, the system further comprises: an application for operating the device, the application running on the processor circuit. The application for operating the device may include the instruction sets for operating the device.

In some embodiments, the artificial intelligence unit is further configured to: receive at least one extra information. In further embodiments, the at least one extra information include one or more of: a time information, a location information, a computed information, or a contextual information. In further embodiments, the at least one extra information include one or more of: an information on an object, an information on an object representation, an information on a collection of object representations, an information on a stream of collections of object representations, an information on a device's circumstance, an information on an instruction set, an information on an application, an information on the processor circuit, an information on the device, or an information on an user. In further embodiments, the artificial intelligence unit is further configured to: learn the first stream of collections of object representations correlated with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include correlating the first stream of collections of object representations with the at least one extra information. The learning the first stream of collections of object representations correlated with at least one extra information may include storing the first stream of collections of object representations correlated with the at least one extra information into the memory unit. In further embodiments, the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations includes anticipating the first one or more instruction sets for operating the device correlated with the first stream of

Autonomous Ex. 2004 p. 337 of 510
Tesla v. Autonomous
IPR2024-00745

collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include comparing an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations. The anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations may include determining that a similarity between an extra information correlated with the new stream of collections of object representations and an extra information correlated with the first stream of collections of object representations exceeds a similarity threshold.

In certain embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: receive, via the user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the artificial intelligence unit is further configured to: rate the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include causing a user interface to display the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations along with one or more rating values as options to be selected by a user. The rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations may include rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user input.

In some embodiments, the system further comprises: a user interface, wherein the artificial intelligence unit is further configured to: cause the user interface to present a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the canceling the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes restoring the processor circuit or the device to a prior state. The restoring the processor circuit or the device to a prior state may include saving the state of the processor circuit or the device prior to executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the system further comprises: an input device configured to receive a user's operating directions, the user's operating directions for instructing the processor circuit on how to operate the device.

Autonomous Ex. 2004 p. 338 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the autonomous device operating includes a partially or a fully autonomous device operating. The partially autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations responsive to a user confirmation. The fully autonomous device operating may include executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations without a user confirmation.

In certain embodiments, the artificial intelligence unit is further configured to: receive a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receive a second one or more instruction sets for operating the device; and learn the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the second stream of collections of object representations includes one or more collections of representations of objects detected by the sensor over time. In further embodiments, the second stream of collections of object representations includes or is associated with a time stamp, an order, or a time related information. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include creating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. The connection may include or is associated with at least one of: an occurrence count, a weight, a parameter, or a data. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device include updating a connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device. In further embodiments, the updating the connection between the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes updating at least one of: an occurrence count, a weight, a parameter, or a data included in or associated with the connection. In further embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a first node of a knowledgebase, and wherein the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device includes storing the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device into a second node of the knowledgebase. The knowledgebase may include a neural network, a graph, a collection of sequences, a sequence, a collection of knowledge cells, a knowledge structure, or a data structure. In further embodiments, the knowledgebase may be stored in the memory unit. The learning the first stream of collections of

Autonomous Ex. 2004 p. 339 of 510
Tesla v. Autonomous
IPR2024-00745

object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include creating a connection between the first node and the second node. The learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device and the learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device may include updating a connection between the first node and the second node. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a neural network and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the neural network. The first node and the second node may be connected by a connection. The first node may be part of a first layer of the neural network and the second node may be part of a second layer of the neural network. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a graph and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the graph. The first node and the second node may be connected by a connection. In further embodiments, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device is stored into a first node of a sequence and the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device is stored into a second node of the sequence.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device

Autonomous Ex. 2004 p. 340 of 510
Tesla v. Autonomous
IPR2024-00745

correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In certain embodiments, the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the one or more processor circuits or from another one or more processor circuits. In further embodiments, the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

In some embodiments, the receiving of (b) includes receiving the first one or more instruction sets for operating the device from the processor circuit or from another processor circuit. In further embodiments, the executing of (f) is performed by the processor circuit or by another processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable as well as the following embodiments.

In certain embodiments, the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into a memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

Autonomous Ex. 2004 p. 341 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations instead of or prior to an instruction set that would have been executed next. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into a register or an element of a processor circuit. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting a processor circuit to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes transmitting, to a processor circuit for execution, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes issuing an interrupt to a processor circuit and executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations following the interrupt. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element that is part of, operating on, or coupled to a processor circuit.

In certain embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by a logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the logic circuit includes a microcontroller. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes inserting the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations into an element of the logic circuit. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting the logic circuit to the first one or more instruction sets for operating the device correlated with

Autonomous Ex. 2004 p. 342 of 510
Tesla v. Autonomous
IPR2024-00745

the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing inputs into the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing, by the logic circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes replacing outputs from the logic circuit with the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing, by an application for operating the device, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying an application, the application including instruction sets for operating the device. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes redirecting an application to one or more alternate instruction sets, the alternate instruction sets comprising the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying a source code, a bytecode, an intermediate code, a compiled code, an interpreted code, a translated code, a runtime code, an assembly code, or a machine code. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: the memory unit, a register of a processor circuit, a storage, or a repository where instruction sets are stored or used. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets for operating an application or an object of the application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying at least one of: an element of a processor circuit, an element of the device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more instruction sets at a source code write time, a compile time, an interpretation time, a translation time, a linking time, a loading time, or a runtime. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes modifying one or more code segments, lines of code, statements, instructions, functions, routines, subroutines, or basic blocks. In further

Autonomous Ex. 2004 p. 343 of 510
Tesla v. Autonomous
IPR2024-00745

embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing one or more of a .NET tool, a .NET application programming interface (API), a Java tool, a Java API, an operating system tool, or an independent tool for modifying instruction sets. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic, an interpreted, or a scripting programming language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic code, a dynamic class loading, or a reflection. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing an assembly language. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a metaprogramming, a self-modifying code, or an instruction set modification tool. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes utilizing at least one of: a dynamic expression creation, a dynamic expression execution, a dynamic function creation, or a dynamic function execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes adding or inserting additional code into a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: modifying, removing, rewriting, or overwriting a code of an application. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes at least one of: branching, redirecting, extending, or hot swapping a code of an application. The branching or redirecting the code may include inserting at least one of: a branch, a jump, or a means for redirecting an execution. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes implementing a user's knowledge, style, or methodology of operating the device in a circumstance. In further embodiments, the executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations includes executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations via an interface. The interface may include a modification interface.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving at least one extra information. In further embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: learning the first stream of collections of object representations correlated with the at least one extra information.

Autonomous Ex. 2004 p. 344 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via a user interface, a user's selection to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: rating the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: presenting, via a user interface, a user with an option to cancel the execution of the executed first one or more instruction sets for operating the device correlated with the first stream of collections of object representations.

In some embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving, via an input device, a user's operating directions, the user's operating directions for instructing a processor circuit on how to operate the device.

In certain embodiments, the operations of the non-transitory computer storage medium and/or the method further comprise: receiving a second stream of collections of object representations, the second stream of collections of object representations including one or more collections of representations of objects detected by the sensor; receiving a second one or more instruction sets for operating the device; and learning the second stream of collections of object representations correlated with the second one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving, by a first processor circuit of the one or more processor circuits, a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a first one or more instruction sets for operating a device. The operations may further comprise: learning, by the first processor circuit of the one or more processor circuits, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device. The operations may further comprise: receiving, by the first processor circuit of the one or more processor circuits, a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The operations may further comprise: anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing, by the first processor circuit of the one or more processor circuits, an execution, by a second processor circuit of the one or more processor circuits, of the first one or more instruction sets for operating the device correlated with the first stream of collections of object

Autonomous Ex. 2004 p. 345 of 510
Tesla v. Autonomous
IPR2024-00745

representations, the causing performed in response to the anticipating, by the first processor circuit of the one or more processor circuits, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first stream of collections of object representations by a first processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the first processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the first processor circuit. The method may further comprise: (d) receiving a new stream of collections of object representations by the first processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the first processor circuit. The method may further comprise: (f) executing, by a second processor circuit, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

Autonomous Ex. 2004 p. 346 of 510
Tesla v. Autonomous
IPR2024-00745

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more instruction sets for operating a device. The operations may further comprise: learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device.

In some aspects, the disclosure relates to a non method comprising: (a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more instruction sets for operating a device by the processor circuit. The method may further comprise: (c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a processor circuit configured to execute instruction sets for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: access the memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The artificial intelligence unit may be further configured to: receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The artificial intelligence unit may be further configured to: cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits

Autonomous Ex. 2004 p. 347 of 510
Tesla v. Autonomous
IPR2024-00745

cause the one or more processor circuits to perform operations comprising: accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device. The operations may further comprise: receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The operations may further comprise: anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. The operations may further comprise: causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

In some aspects, the disclosure relates to a method comprising: (a) accessing a memory unit that comprises a plurality of streams of collections of object representations correlated with one or more instruction sets for operating a device, the plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device including a first stream of collections of object representations correlated with a first one or more instruction sets for operating the device, the accessing of (a) performed by a processor circuit. The method may further comprise: (b) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by a sensor. The method may further comprise: (c) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (c) performed by the processor circuit. The method may further comprise: (d) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (d) performed in response to the anticipating of (c). The method may further comprise: (e) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (d).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices.

Autonomous Ex. 2004 p. 348 of 510
Tesla v. Autonomous
IPR2024-00745

In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some embodiments, the logic circuit configured to receive inputs and produce outputs includes a logic circuit configured to produce outputs based at least in part on logic operations performed on the inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes correlating the first collection of object representations with the first one or more inputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more inputs includes storing the first collection of object representations correlated with the first one or more inputs into the memory unit, the first collection of object representations correlated with the first one or more inputs being part of a plurality of collections of object representations correlated with one or more inputs stored in the memory unit. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes transmitting, to the logic circuit, the first one or more inputs correlated with the first collection of object representations. In further embodiments, the causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations includes replacing one or more inputs into the logic circuit with the first one or more inputs correlated with the first collection of object representations.

Autonomous Ex. 2004 p. 349 of 510
Tesla v. Autonomous
IPR2024-00745

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the logic circuit to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the device performs one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) receiving, by the logic circuit, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the device, one or more operations defined by one or more outputs for operating the device produced by the logic circuit.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Autonomous Ex. 2004 p. 350 of 510
Tesla v. Autonomous
IPR2024-00745

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: a logic circuit configured to receive inputs and produce outputs, wherein the outputs are used for operating a device. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: receive a first one or more outputs, the first one or more outputs transmitted from the logic circuit. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more outputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit.

In some embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes correlating the first collection of object representations with the first one or more outputs. In further embodiments, the learning the first collection of object representations correlated with the first one or more outputs includes storing the first collection of object representations correlated with the first one or more outputs into the memory unit, the first collection of object representations correlated with the first one or more outputs being part of a plurality of collections of object representations correlated with one or more outputs stored in the memory unit. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes comparing at least one portion of the new collection of object representations with at least one portion of the first collection of object representations. In further embodiments, the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations includes determining that there is at least a partial match between the new collection of object representations and the first collection of object representations. In further embodiments, the causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations includes replacing one or more outputs from the logic circuit with the first one or more outputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects

Autonomous Ex. 2004 p. 351 of 510
Tesla v. Autonomous
IPR2024-00745

detected by a sensor. The operations may further comprise: receiving a first one or more outputs, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The operations may further comprise: learning the first collection of object representations correlated with the first one or more outputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the device to perform one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more outputs by the processor circuit, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more outputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more outputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the processor circuit. The method may further comprise: (f) performing, by the device, one or more operations defined by the first one or more outputs correlated with the first collection of object representations, the one or more operations by the device performed in response to the anticipating of (e).

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

In some aspects, the disclosure relates to a system for learning and using a device's circumstances for autonomous device operating. The system may be implemented at least in part on one or more computing devices. In some embodiments, the system comprises: an actuator configured to receive inputs and perform motions. The system may further comprise: a memory unit configured to store data. The system may further comprise: a sensor configured to detect objects. The system may further comprise: an artificial intelligence unit. The artificial intelligence unit may be configured to: receive a first collection of object representations, the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence

Autonomous Ex. 2004 p. 352 of 510
Tesla v. Autonomous
IPR2024-00745

unit may be further configured to: receive a first one or more inputs, wherein the first one or more inputs are also received by the actuator. The artificial intelligence unit may be further configured to: learn the first collection of object representations correlated with the first one or more inputs. The artificial intelligence unit may be further configured to: receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may be further configured to: anticipate the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may be further configured to: cause the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising: receiving a first collection of object representations, the first collection of object representations including one or more representations of objects detected by a sensor. The operations may further comprise: receiving a first one or more inputs, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The operations may further comprise: learning the first collection of object representations correlated with the first one or more inputs. The operations may further comprise: receiving a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The operations may further comprise: anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The operations may further comprise: causing the actuator to receive the first one or more inputs correlated with the first collection of object representations, the causing performed in response to the anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, wherein the actuator performs one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

In some aspects, the disclosure relates to a method comprising: (a) receiving a first collection of object representations by a processor circuit, the first collection of object representations including one or more representations of objects detected by a sensor. The method may further comprise: (b) receiving a first one or more inputs by the processor circuit, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. The method may further comprise: (c) learning the first collection of object representations correlated with the first one or more inputs, the learning of (c) performed by the processor circuit. The method may further comprise: (d) receiving a new collection of object representations by the processor circuit, the new collection of object representations including one or more representations of objects detected by the sensor. The method may further comprise: (e) anticipating the first one or more inputs correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations, the anticipating of (e) performed by the

Autonomous Ex. 2004 p. 353 of 510
Tesla v. Autonomous
IPR2024-00745

processor circuit. The method may further comprise: (f) receiving, by the actuator, the first one or more inputs correlated with the first collection of object representations, the receiving of (f) performed in response to the anticipating of (e). The method may further comprise: (g) performing, by the actuator, one or more motions defined by the first one or more inputs correlated with the first collection of object representations.

The operations or steps of the non-transitory computer storage medium and/or the method may be performed by any of the elements of the above described systems as applicable. The non-transitory computer storage medium and/or the method may include any of the operations, steps, and embodiments of the above described systems, non-transitory computer storage media, and/or methods as applicable.

Other features and advantages of the disclosure will become apparent from the following description, including the claims and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates a block diagram of Computing Device 70 that can provide processing capabilities used in some of the disclosed embodiments.

Fig. 2 illustrates an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100).

Figs. 3A-3E illustrate various embodiments of Sensors 92 and elements of Object Processing Unit 93.

Figs. 4A-4B, illustrate an exemplary embodiment of Objects 615 detected in Device's 98 surrounding, and resulting Collection of Object Representations 525.

Fig. 5 illustrates some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Processor 11 registers, memory, or other computing system components.

Figs. 6A-6B illustrate some embodiments of obtaining instruction sets, data, and/or other information through tracing, profiling, or sampling of Logic Circuit 250.

Figs. 7A-7E illustrate some embodiments of Instruction Sets 526.

Figs. 8A-8B illustrate some embodiments of Extra Information 527.

Fig. 9 illustrates an embodiment where DCADO Unit 100 is part of or operating on Processor 11.

Fig. 10 illustrates an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95.

Fig. 11 illustrates an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation.

Fig. 12 illustrates an embodiment of Artificial Intelligence Unit 110.

Fig. 13 illustrates an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 14 illustrates another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 15 illustrates an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Fig. 16 illustrates another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527.

Autonomous Ex. 2004 p. 354 of 510
Tesla v. Autonomous
IPR2024-00745

Fig. 17 illustrates various artificial intelligence methods, systems, and/or models that can be utilized in DCADO Unit 100 embodiments.

Fig. 18A-18C illustrate embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853.

Fig. 19 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d.

Fig. 20 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a.

Fig. 21 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853.

Fig. 22 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b.

Fig. 23 illustrates an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c.

Fig. 24 illustrates an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800.

Fig. 25 illustrates an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800.

Fig. 26 illustrates an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons.

Fig. 27 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a.

Fig. 28 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b.

Fig. 29 illustrates an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c.

Fig. 30 illustrates some embodiments of modifying execution and/or functionality of Processor 11 through modification of Processor 11 registers, memory, or other computing system components.

Figs. 31A-31B illustrate some embodiments of modifying execution and/or functionality of Logic Circuit 250 through modification of inputs and/or outputs of Logic Circuit 250.

Fig. 32 illustrates a flow chart diagram of an embodiment of method 9100 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 33 illustrates a flow chart diagram of an embodiment of method 9200 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 34 illustrates a flow chart diagram of an embodiment of method 9300 for learning and/or using a device's circumstances for autonomous device operation.

Autonomous Ex. 2004 p. 355 of 510
Tesla v. Autonomous
IPR2024-00745

Fig. 35 illustrates a flow chart diagram of an embodiment of method 9400 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 36 illustrates a flow chart diagram of an embodiment of method 9500 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 37 illustrates a flow chart diagram of an embodiment of method 9600 for learning and/or using a device's circumstances for autonomous device operation.

Fig. 38 illustrates an exemplary embodiment of Loader 98a.

Fig. 39 illustrates an exemplary embodiment of Boat 98b.

Fig. 40 illustrates an exemplary embodiment of utilizing Area of Interest 450 around Boat 98b.

Like reference numerals in different figures indicate like elements. Horizontal or vertical "..." or other such indicia may be used to indicate additional instances of the same type of element. n, m, x, or other such letters or indicia represent integers or other sequential numbers that follow the sequence where they are indicated. It should be noted that n, m, x, or other such letters or indicia may represent different numbers in different elements even where the elements are depicted in the same figure. In general, n, m, x, or other such letters or indicia may follow the sequence and/or context where they are indicated. Any of these or other such letters or indicia may be used interchangeably depending on context and space available. The drawings are not necessarily to scale, with emphasis instead being placed upon illustrating the embodiments, principles, and concepts of the disclosure. A line or arrow between any of the disclosed elements comprises an interface that enables the coupling, connection, and/or interaction between the elements.

DETAILED DESCRIPTION

The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation comprise apparatuses, systems, methods, features, functionalities, and/or applications that enable learning a device's circumstances including objects with various properties along with correlated instruction sets for operating the device, storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, etc.), and operating a device autonomously. The disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, any of their elements, any of their embodiments, or a combination thereof can generally be referred to as DCADO, DCADO Unit, or as other suitable name or reference.

Referring now to Fig. 1, an embodiment is illustrated of Computing Device 70 (also referred to simply as computing device, computing system, or other suitable name or reference, etc.) that can provide processing capabilities used in some embodiments of the forthcoming disclosure. Later described devices, systems, and methods, in combination with processing capabilities of Computing Device 70, enable learning and/or using a device's circumstances for autonomous device operation and/or other functionalities described herein. Various embodiments of the disclosed devices, systems, and methods include hardware, functions, logic, programs, and/or a combination thereof that can be implemented using any type or form of computing, computing enabled, or other device or system such as a mobile device, a computer, a computing enabled telephone, a server, a gaming device, a television device, a digital camera, a GPS receiver, a media player, an embedded device, a supercomputer, a wearable device, an implantable device, a cloud, or any other type or form of computing, computing enabled, or

Autonomous Ex. 2004 p. 356 of 510
Tesla v. Autonomous
IPR2024-00745

other device or system capable of performing the operations described herein.

In some designs, Computing Device 70 comprises hardware, processing techniques or capabilities, programs, or a combination thereof. Computing Device 70 includes one or more central processing units, which may also be referred to as processors 11. Processor 11 includes one or more memory ports 10 and/or one or more input-output ports, also referred to as I/O ports 15, such as I/O ports 15A and 15B. Processor 11 may be special or general purpose. Computing Device 70 may further include memory 12, which can be connected to the remainder of the components of Computing Device 70 via bus 5. Memory 12 can be connected to processor 11 via memory port 10. Computing Device 70 may also include display device 21 such as a monitor, projector, glasses, and/or other display device. Computing Device 70 may also include Human-machine Interface 23 such as a keyboard, a pointing device, a mouse, a touchscreen, a joystick, a remote controller, and/or other input device. In some implementations, Human-machine Interface 23 can be connected with bus 5 or directly connected with specific elements of Computing Device 70. Computing Device 70 may include additional elements such as one or more input/output devices 13. Processor 11 may include or be interfaced with cache memory 14. Storage 27 may include memory, which provides an operating system 17 (i.e. also referred to as OS 17, etc.), additional application programs 18, and/or data space 19 in which additional data or information can be stored. Alternative memory device 16 can be connected to the remaining components of Computing Device 70 via bus 5. Network interface 25 can also be connected with bus 5 and be used to communicate with external computing devices via a network. Some or all described elements of Computing Device 70 can be directly or operatively connected or coupled with each other using any other connection means known in art. Other additional elements may be included as needed, or some of the disclosed ones may be excluded, or a combination thereof may be utilized in alternate implementations of Computing Device 70.

Processor 11 includes one or more circuits or devices that can execute instructions fetched from memory 12 and/or other element. Processor 11 may include any combination of hardware and/or processing techniques or capabilities for executing or implementing logic functions or programs. Processor 11 may include a single core or a multi core processor. Processor 11 includes the functionality for loading operating system 17 and operating any application programs 18 thereon. In some embodiments, Processor 11 can be provided in a microprocessing or a processing unit, such as, for example, Snapdragon processor produced by Qualcomm Inc., processor by Intel Corporation of Mountain View, California, processor manufactured by Motorola Corporation of Schaumburg, Ill.; processor manufactured by Transmeta Corporation of Santa Clara, Calif.; processor manufactured by International Business Machines of White Plains, N.Y.; processor manufactured by Advanced Micro Devices of Sunnyvale, California, or any computing circuit or device for performing similar functions. In other embodiments, processor 11 can be provided in a graphics processing unit (GPU), visual processing unit (VPU), or other highly parallel processing circuit or device such as, for example, nVidia GeForce line of GPUs, AMD Radeon line of GPUs, and/or others. Such GPUs or other highly parallel processing circuits or devices may provide superior performance in processing operations on neural networks, graphs, and/or other data structures. In further embodiments, processor 11 can be provided in a micro controller such as, for example, Texas instruments, Atmel, Microchip Technology, ARM, Silicon Labs, Intel, and/or other lines of micro controllers. In further embodiments, processor 11 can be provided in a quantum processor such as, for example, D-Wave Systems, Microsoft, Intel, IBM, Google, Toshiba, and/or other lines of quantum processors. In further embodiments, processor 11 can be provided in a biocomputer

Autonomous Ex. 2004 p. 357 of 510
Tesla v. Autonomous
IPR2024-00745

such as DNA-based computer, protein-based computer, molecule-based computer, and/or others. In further embodiments, processor 11 includes any circuit or device for performing logic operations. Processor 11 can be based on any of the aforementioned or other available processors capable of operating as described herein. Computing Device 70 may include one or more of the aforementioned or other processors. In some designs, processor 11 can communicate with memory 12 via a system bus 5. In other designs, processor 11 can communicate directly with memory 12 via a memory port 10.

Memory 12 includes one or more circuits or devices capable of storing data. In some embodiments, Memory 12 can be provided in a semiconductor or electronic memory chip such as static random access memory (SRAM), Flash memory, Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), Ferroelectric RAM (FRAM), and/or others. In other embodiments, Memory 12 includes any volatile memory. In general, Memory 12 can be based on any of the aforementioned or other available memories capable of operating as described herein.

Storage 27 includes one or more devices or mediums capable of storing data. In some embodiments, Storage 27 can be provided in a device or medium such as a hard drive, flash drive, optical disk, and/or others. In other embodiments, Storage 27 can be provided in a biological storage device such as DNA-based storage device, protein-based storage device, molecule-based storage device, and/or others. In further embodiments, Storage 27 can be provided in an optical storage device such as holographic storage, and/or others. In further embodiments, Storage 27 may include any non-volatile memory. In general, Storage 27 can be based on any of the aforementioned or other available storage devices or mediums capable of operating as described herein. In some aspects, Storage 27 may include any features, functionalities, and embodiments of Memory 12, and vice versa, as applicable.

Processor 11 can communicate directly with cache memory 14 via a connection means such as a secondary bus which may also sometimes be referred to as a backside bus. In some embodiments, processor 11 can communicate with cache memory 14 using the system bus 5. Cache memory 14 may typically have a faster response time than main memory 12 and can include a type of memory which is considered faster than main memory 12 such as, for example, SRAM, BSRAM, or EDRAM. Cache memory includes any structure such as multilevel caches, for example. In some embodiments, processor 11 can communicate with one or more I/O devices 13 via a system bus 5. Various busses can be used to connect processor 11 to any of the I/O devices 13 such as a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, a NuBus, and/or others. In some embodiments, processor 11 can communicate directly with I/O device 13 via HyperTransport, Rapid I/O, or InfiniBand. In further embodiments, local busses and direct communication can be mixed. For example, processor 11 can communicate with an I/O device 13 using a local interconnect bus and communicate with another I/O device 13 directly. Similar configurations can be used for any other components described herein.

Autonomous Ex. 2004 p. 358 of 510
Tesla v. Autonomous
IPR2024-00745

Computing Device 70 may further include alternative memory such as a SD memory slot, a USB memory stick, an optical drive such as a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive or a BlueRay disc, a hard-drive, and/or any other device comprising non-volatile memory suitable for storing data or installing application programs. Computing Device 70 may further include a storage device 27 comprising any type or form of non-volatile memory for storing an operating system (OS) such as any type or form of Windows OS, Mac OS, Unix OS, Linux OS, Android OS, iPhone OS, mobile version of Windows OS, an embedded OS, or any other OS that can operate on Computing Device 70. Computing Device 70 may also include application programs 18, and/or data space 19 for storing additional data or information. In some embodiments, alternative memory 16 can be used as or similar to storage device 27. Additionally, OS 17 and/or application programs 18 can be operable from a bootable medium such as, for example, a flash drive, a micro SD card, a bootable CD or DVD, and/or other bootable medium.

Application Program 18 (also referred to as program, computer program, application, script, code, or other suitable name or reference) comprises instructions that can provide functionality when executed by processor 11. As such, Application Program 18 may be used to operate (i.e. perform operations on/with) or control a device or system. Application program 18 can be implemented in a high-level procedural or object-oriented programming language, or in a low-level machine or assembly language. Any language used can be compiled, interpreted, or otherwise translated into machine language. Application program 18 can be deployed in any form including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing system. Application program 18 does not necessarily correspond to a file in a file system. A program can be stored in a portion of a file that may hold other programs or data, in a single file dedicated to the program, or in multiple files (i.e. files that store one or more modules, sub programs, or portions of code, etc.). Application Program 18 can be delivered in various forms such as, for example, executable file, library, script, plugin, addon, applet, interface, console application, web application, application service provider (ASP)-type application, operating system, and/or other forms. Application program 18 can be deployed to be executed on one computing device or on multiple computing devices (i.e. cloud, distributed, or parallel computing, etc.), or at one site or distributed across multiple sites interconnected by a communication network or an interface.

Network interface 25 can be utilized for interfacing Computing Device 70 with other devices via a network through a variety of connections including telephone lines, wired or wireless connections, LAN or WAN links (i.e. 802.11, T1, T3, 56 kb, X.25, etc.), broadband connections (i.e. ISDN, Frame Relay, ATM, etc.), or a combination thereof. Examples of networks include the Internet, an intranet, an extranet, a local area network (LAN), a wide area network (WAN), a personal area network (PAN), a home area network (HAN), a campus area network (CAN), a metropolitan area network (MAN), a global area network (GAN), a storage area network (SAN), virtual network, a virtual private network (VPN), a Bluetooth network, a wireless network, a wireless LAN, a radio network, a HomePNA, a power line communication network, a G.hn network, an optical fiber network, an Ethernet network, an active networking network, a client-server network, a peer-to-peer network, a bus network, a star network, a ring network, a mesh network, a star-bus network, a tree network, a hierarchical topology network, and/or other networks. Network interface 25 may include a built-in network adapter, network interface card, PCMCIA network card, card bus network adapter, wireless network adapter, Bluetooth network adapter, WiFi network adapter, USB network adapter, modem, and/or any other device suitable for interfacing Computing Device 70 with any type of network capable of communication and/or operations described herein.

Autonomous Ex. 2004 p. 359 of 510
Tesla v. Autonomous
IPR2024-00745

I/O devices 13 may be present in various shapes or forms in Computing Device 70. Examples of I/O device 13 capable of input include a joystick, a keyboard, a mouse, a trackpad, a trackpoint, a touchscreen, a trackball, a microphone, a drawing tablet, a glove, a tactile input device, a still or video camera, and/or other input device. Examples of I/O device 13 capable of output include a video display, a touchscreen, a projector, a glasses, a speaker, a tactile output device, and/or other output device. Examples of I/O device 13 capable of input and output include a disk drive, an optical storage device, a modem, a network card, and/or other input/output device. I/O device 13 can be interfaced with processor 11 via an I/O port 15, for example. In some aspects, I/O device 13 can be a bridge between system bus 5 and an external communication bus such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, a Serial Attached small computer system interface bus, and/or other bus.

An output interface (not shown) such as a graphical user interface, an acoustic output interface, a tactile output interface, any device driver (i.e. audio, video, or other driver), and/or other output interface or system can be utilized to process output from elements of Computing Device 70 for conveyance on an output device such as Display 21. In some aspects, Display 21 or other output device itself may include an output interface for processing output from elements of Computing Device 70. Further, an input interface (not shown) such as a keyboard listener, a touchscreen listener, a mouse listener, any device driver (i.e. audio, video, keyboard, mouse, touchscreen, or other driver), and/or other input interface or system can be utilized to process input from Human-machine Interface 23 or other input device for use by elements of Computing Device 70. In some aspects, Human-machine Interface 23 or other input device itself may include an input interface for processing input for use by elements of Computing Device 70.

Computing Device 70 may include or be connected to multiple display devices 21. Display devices 21 can each be of the same or different type or form. Computing Device 70 and/or its elements comprise any type or form of suitable hardware, programs, or a combination thereof to support, enable, or provide for the connection and use of multiple display devices 21. In one example, Computing Device 70 includes any type or form of video adapter, video card, driver, and/or library to interface, communicate, connect, or otherwise use display devices 21. In some aspects, a video adapter may include multiple connectors to interface to multiple display devices 21. In other aspects, Computing Device 70 includes multiple video adapters, with each video adapter connected to one or more display devices 21. In some embodiments, Computing Device's 70 operating system can be configured for using multiple displays 21. In other embodiments, one or more display devices 21 can be provided by one or more other computing devices such as remote computing devices connected to Computing Device 70 via a network or an interface.

Computing Device 70 can operate under the control of operating system 17, which may support Computing Device's 70 basic functions, interface with and manage hardware resources, interface with and manage peripherals, provide common services for application programs, schedule tasks, and/or perform other functionalities. A modern operating system enables features and functionalities such as a high resolution display, graphical user interface (GUI), touchscreen, cellular network connectivity (i.e. mobile operating system, etc.), Bluetooth connectivity, WiFi connectivity, global positioning system (GPS) capabilities, mobile navigation, microphone, speaker, still picture camera, video camera, voice recorder, speech recognition, music player, video player, near field communication,

Autonomous Ex. 2004 p. 360 of 510
Tesla v. Autonomous
IPR2024-00745

personal digital assistant (PDA), and/or other features, functionalities, or applications. For example, Computing Device 70 can use any conventional operating system, any embedded operating system, any real-time operating system, any open source operating system, any video gaming operating system, any proprietary operating system, any online operating system, any operating system for mobile computing devices, or any other operating system capable of running on Computing Device 70 and performing operations described herein. Example of operating systems include Windows XP, Windows 7, Windows 8, Windows 10, etc. manufactured by Microsoft Corporation of Redmond, Wash.; Mac OS, iPhone OS, etc. manufactured by Apple Computer of Cupertino, Calif.; OS/2 manufactured by International Business Machines of Armonk, N.Y.; Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah; or any type or form of a Unix operating system, and/or others. Any operating systems such as the ones for Android devices can similarly be utilized.

Computing Device 70 can be implemented as or be part of various model architectures such as web services, distributed computing, grid computing, cloud computing, and/or other architectures. For example, in addition to the traditional desktop, server, or mobile operating system architectures, a cloud-based operating system can be utilized to provide the structure on which embodiments of the disclosure can be implemented. Other aspects of Computing Device 70 can also be implemented in the cloud without departing from the spirit and scope of the disclosure. For example, memory, storage, processing, and/or other elements can be hosted in the cloud. In some embodiments, Computing Device 70 can be implemented on multiple devices. For example, a portion of Computing Device 70 can be implemented on a mobile device and another portion can be implemented on wearable electronics.

Computing Device 70 can be or include any mobile device, a mobile phone, a smartphone (i.e. iPhone, Windows phone, Blackberry phone, Android phone, etc.), a tablet, a personal digital assistant (PDA), wearable electronics, implantable electronics, and/or other mobile device capable of implementing the functionalities described herein. Computing Device 70 can also be or include an embedded device, which can be any device or system with a dedicated function within another device or system. Embedded systems range from the simplest ones dedicated to one task with no user interface to complex ones with advanced user interface that may resemble modern desktop computer systems. Examples of devices comprising an embedded device include a mobile telephone, a personal digital assistant (PDA), a gaming device, a media player, a digital still or video camera, a pager, a television device, a set-top box, a personal navigation device, a global positioning system (GPS) receiver, a portable storage device (i.e. a USB flash drive, etc.), a digital watch, a DVD player, a printer, a microwave oven, a washing machine, a dishwasher, a gateway, a router, a hub, an automobile entertainment system, an automobile navigation system, a refrigerator, a washing machine, a factory automation device, an assembly line device, a factory floor monitoring device, a thermostat, an automobile, a factory controller, a telephone, a network bridge, and/or other devices. An embedded device can operate under the control of an operating system for embedded devices such as MicroC/OS-II, QNX, VxWorks, eCos, TinyOS, Windows Embedded, Embedded Linux, and/or other embedded device operating systems.

Various implementations of the disclosed devices, systems, and methods can be realized in digital electronic circuitry, integrated circuitry, logic gates, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), computer hardware, firmware, programs, virtual machines, and/or combinations thereof including their structural, logical, and/or physical equivalents.

Autonomous Ex. 2004 p. 361 of 510
Tesla v. Autonomous
IPR2024-00745

The disclosed devices, systems, and methods may include clients and servers. A client and server are generally, but not always, remote from each other and typically, but not always, interact via a network or an interface. The relationship of a client and server may arise by virtue of computer programs running on their respective computers and having a client-server relationship to each other, for example.

The disclosed devices, systems, and methods can be implemented in a computing system that includes a back end component, a middleware component, a front end component, or any combination thereof. The components of the system can be interconnected by any form or medium of digital data communication such as, for example, a network.

Computing Device 70 may include or be interfaced with a computer program product comprising instructions or logic encoded on a computer-readable medium. Such instructions or logic, when executed, may configure or cause one or more processors to perform the operations and/or functionalities disclosed herein. For example, a computer program can be provided or encoded on a computer-readable medium such as an optical medium (i.e. DVD-ROM, etc.), flash drive, hard drive, any memory, firmware, or other medium. Computer program can be installed onto a computing device to cause the computing device to perform the operations and/or functionalities disclosed herein. Machine-readable medium, computer-readable medium, or other such terms may refer to any computer program product, apparatus, and/or device for providing instructions and/or data to one or more programmable processors. As such, machine-readable medium includes any medium that can send and/or receive machine instructions as a machine-readable signal. Examples of a machine-readable medium include a volatile and/or non-volatile medium, a removable and/or non-removable medium, a communication medium, a storage medium, and/or other medium. A communication medium, for example, can transmit computer readable instructions and/or data in a modulated data signal such as a carrier wave or other transport technique, and may include any other form of information delivery medium known in art. A non-transitory machine-readable medium comprises all machine-readable media except for a transitory, propagating signal.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented entirely or in part in a device (i.e. microchip, circuitry, logic gates, electronic device, computing device, special or general purpose processor, etc.) or system that comprises (i.e. hard coded, internally stored, etc.) or is provided with (i.e. externally stored, etc.) instructions for implementing DCADO functionalities. As such, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include the processing, memory, storage, and/or other features, functionalities, and embodiments of Computing Device 70 or elements thereof. Such device or system can operate on its own (i.e. standalone device or system, etc.), be embedded in another device or system (i.e. an industrial machine, a robot, a vehicle, a toy, a smartphone, a television device, an appliance, and/or any other device or system capable of housing the elements needed for DCADO functionalities), work in combination with other devices or systems, or be available in any other configuration. In other embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, may include Alternative Memory 16 that provides instructions for implementing DCADO functionalities to one or more Processors 11. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be

Autonomous Ex. 2004 p. 362 of 510
Tesla v. Autonomous
IPR2024-00745

implemented entirely or in part as a computer program and executed by one or more Processors 11. Such program can be implemented in one or more modules or units of a single or multiple computer programs. Such program may be able to attach to or interface with, inspect, and/or take control of another application program to implement DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented as a network, web, distributed, cloud, or other such application accessed on one or more remote computing devices (i.e. servers, cloud, etc.) via Network Interface 25, such remote computing devices including processing capabilities and instructions for implementing DCADO functionalities. In further embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be (1) attached to or interfaced with any computing device or application program, (2) included as a feature of an operating system, (3) built (i.e. hard coded, etc.) into any computing device or application program, and/or (4) available in any other configuration to provide its functionalities.

In some embodiments, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, can be implemented at least in part in a computer program such as Java application or program. Java provides a robust and flexible environment for application programs including flexible user interfaces, robust security, built-in network protocols, powerful application programming interfaces, database or DBMS connectivity and interfacing functionalities, file manipulation capabilities, support for networked applications, and/or other features or functionalities. Application programs based on Java can be portable across many devices, yet leverage each device's native capabilities. Java supports the feature sets of most smartphones and a broad range of connected devices while still fitting within their resource constraints. Various Java platforms include virtual machine features comprising a runtime environment for application programs. Java platforms provide a wide range of user-level functionalities that can be implemented in application programs such as displaying text and graphics, playing and recording audio content, displaying and recording visual content, communicating with another computing device, and/or other functionalities. It should be understood that the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation, or elements thereof, are programming language, platform, and operating system independent. Examples of programming languages that can be used instead of or in addition to Java include C, C++, Cobol, Python, Java Script, Tcl, Visual Basic, Pascal, VB Script, Perl, PHP, Ruby, and/or other programming languages capable of implementing the functionalities described herein.

Where a reference to a specific file or file type is used herein, other files or file types can be substituted.

Where a reference to a data structure is used herein, it should be understood that any variety of data structures can be used such as, for example, array, list, linked list, doubly linked list, queue, tree, heap, graph, map, grid, matrix, multi-dimensional matrix, table, database, database management system (DBMS), file, neural network, and/or any other type or form of a data structure including a custom one. A data structure may include one or more fields or data fields that are part of or associated with the data structure. A field or data field may include a data, an object, a data structure, and/or any other element or a reference/pointer thereto. A data structure can be stored in one or more memories, files, or other repositories. A data structure and/or elements thereof, when stored in a memory, file, or other repository, may be stored in a different arrangement than the arrangement of the data structure and/or elements thereof. For example, a sequence of elements can be stored in an arrangement other

Autonomous Ex. 2004 p. 363 of 510
Tesla v. Autonomous
IPR2024-00745

than a sequence in a memory, file, or other repository.

Where a reference to a repository is used herein, it should be understood that a repository may be or include one or more files or file systems, one or more storage locations or structures, one or more storage systems, one or more memory locations or structures, and/or other file, storage, memory, or data arrangements.

Where a reference to an interface is used herein, it should be understood that the interface comprises any hardware, device, system, program, method, and/or combination thereof that enable direct or operative coupling, connection, and/or interaction of the elements between which the interface is indicated. A line or arrow shown in the figures between any of the depicted elements comprises such interface. Examples of an interface include a direct connection, an operative connection, a wired connection (i.e. wire, cable, etc.), a wireless connection, a device, a network, a bus, a circuit, a firmware, a driver, a bridge, a program, a combination thereof, and/or others.

Where a reference to an element coupled or connected to another element is used herein, it should be understood that the element may be in communication or other interactive relationship with the other element. Furthermore, an element coupled or connected to another element can be coupled or connected to any other element in alternate implementations. Terms coupled, connected, interfaced, or other such terms may be used interchangeably herein depending on context.

Where a reference to an element matching another element is used herein, it should be understood that the element may be equivalent or similar to the other element. Therefore, the term match or matching can refer to total equivalence or similarity depending on context.

Where a reference to a device is used herein, it should be understood that the device may include or be referred to as a system, and vice versa depending on context, since a device may include a system of elements and a system may be embodied in a device.

Where a reference to a collection of elements is used herein, it should be understood that the collection of elements may include one or more elements. In some aspects or contexts, a reference to a collection of elements does not imply that the collection is an element itself.

Where a reference to an object is used herein, it should be understood that the object may be a physical object (i.e. object detected in a device's surrounding, etc.), an electronic object (i.e. object in an object oriented application program, etc.), and/or other object depending on context.

Where a mention of a function, method, routine, subroutine, or other such procedure is used herein, it should be understood that the function, method, routine, subroutine, or other such procedure comprises a call, reference, or pointer to the function, method, routine, subroutine, or other such procedure.

Where a mention of data, object, data structure, item, element, or thing is used herein, it should be understood that the data, object, data structure, item, element, or thing comprises a reference or pointer to the data, object, data structure, item, element, or thing.

Referring to Fig. 2, an embodiment of Device 98 comprising Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit 100) is illustrated. Device 98 also comprises interconnected Processor 11, Human-machine Interface 23, Sensor 92, Object Processing Unit 93, Memory 12, and Storage 27. Processor 11 includes or executes Application Program 18. DCADO Unit 100 comprises interconnected Artificial Intelligence Unit 110, Acquisition Interface 120, and Modification Interface 130. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized

Autonomous Ex. 2004 p. 364 of 510
Tesla v. Autonomous
IPR2024-00745

in alternate embodiments.

In one example, the teaching presented by the disclosure can be implemented in a device or system for learning and/or using a device's circumstances for autonomous device operation. The device or system may include a processor circuit (i.e. Processor 11, etc.) configured to execute instruction sets (i.e. Instruction Sets 526, etc.) for operating a device. The device or system may further include a memory unit (i.e. Memory 12, etc.) configured to store data. The device or system may further include a sensor (i.e. Sensor 92, etc.) configured to detect objects (i.e. Objects 615, etc.). The device or system may further include an artificial intelligence unit (i.e. Artificial Intelligence Unit 110, etc.). The artificial intelligence unit may be configured to receive a first collection of object representations (i.e. Collection of Object Representations 525, etc.), the first collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to receive a first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to learn the first collection of object representations correlated with the first one or more instruction sets for operating the device. The artificial intelligence unit may also be configured to receive a new collection of object representations, the new collection of object representations including one or more representations of objects detected by the sensor. The artificial intelligence unit may also be configured to anticipate the first one or more instruction sets for operating the device correlated with the first collection of object representations based on at least a partial match between the new collection of object representations and the first collection of object representations. The artificial intelligence unit may also be configured to cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first collection of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations, the one or more operations performed in response to the executing by the processor circuit. Any of the operations of the aforementioned elements can be performed repeatedly and/or in different orders in alternate embodiments. In some embodiments, a stream of collections of object representations can be used instead of or in addition to any collection of object representations such as, for example, using a first stream of collections of object representations instead of the first collection of object representations. In other embodiments, a logic circuit (i.e. Logic Circuit 250, etc.) may be used instead of the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into or one or more outputs from the logic circuit. In further embodiments, an actuator may be included instead of or in addition to the processor circuit. In such embodiments, one or more instruction sets for operating the device (i.e. first one or more instruction sets for operating the device, etc.) may include or be substituted with one or more inputs into the actuator. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments. The device or system for learning and/or using a device's circumstances for autonomous device operation may include any actions or operations of any of the disclosed methods such as methods 9100, 9200, 9300, 9400, 9500, 9600, and/or others (all later described).

Device 98 comprises any hardware, programs, or a combination thereof. Although, Device 98 is referred to as a device herein, Device 98 may be or include a system as a system may be embodied in Device 98. Device 98 may include any features, functionalities, and embodiments of Computing Device 70, or elements thereof. In some

Autonomous Ex. 2004 p. 365 of 510
Tesla v. Autonomous
IPR2024-00745

embodiments, Device 98 includes a computing enabled device for performing mechanical or physical operations (i.e. via actuators, etc.). In other embodiments, Device 98 includes a computing enabled device for performing non-mechanical and/or other operations. Examples of Device 98 include an industrial machine, a toy, a robot, a vehicle, an appliance, a control device, a smartphone or other mobile computer, any computer, and/or other computing enabled device or machine. Such device or machine may be built for any function or purpose some examples of which are described later.

User 50 (also referred to simply as user or other suitable name or reference) comprises a human user or non-human user. A non-human User 50 includes any device, system, program, and/or other mechanism for operating or controlling Device 98 and/or elements thereof. In one example, User 50 may issue an operating direction to Application Program 18 responsive to which Application Program's 18 instructions or instruction sets may be executed by Processor 11 to perform a desired operation on Device 98. In another example, User 50 may issue an operating direction to Processor 11, Logic Circuit 250 (later described), and/or other processing element responsive to which Processor 11, Logic Circuit 250, and/or other processing element may implement logic to perform a desired operation on Device 98. User's 50 operating directions comprise any user inputted data (i.e. values, text, symbols, etc.), directions (i.e. move right, move up, move forward, copy an item, click on a link, etc.), instructions or instruction sets (i.e. manually inputted instructions or instruction sets, etc.), and/or other inputs or information. A non-human User 50 can utilize more suitable interfaces instead of, or in addition to, Human-machine Interface 23 and/or Display 21 for controlling Device 98 and/or elements thereof. Examples of such interfaces include an application programming interface (API), bridge (i.e. bridge between applications, devices, or systems, etc.), driver, socket, direct or operative connection, handle, function/routine/subroutine, and/or other interfaces.

In some embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect an actuator (not shown). Actuator comprises the functionality for implementing motion, actions, behaviors, maneuvers, and/or other mechanical or physical operations. Device 98 may include one or more actuators to enable Device 98 to perform mechanical, physical, or other operations and/or to interact with its environment. For example, an actuator may include or be coupled to an element such as a wheel, arm, or other element to act upon the environment. Examples of an actuator include a motor, a linear motor, a servomotor, a hydraulic element, a pneumatic element, an electro-magnetic element, a spring element, and/or other actuators. Examples of types of actuators include a rotary actuator, a linear actuator, and/or other types of actuators. In other embodiments, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing element may control or affect any other device or element instead of or in addition to an actuator.

Referring to Figs. 3A-3E, various embodiments of Sensors 92 and elements of Object Processing Unit 93 are illustrated.

Sensor 92 (also referred to simply as sensor or other suitable name or reference) comprises the functionality for obtaining or detecting information about its environment, and/or other functionalities. As such, one or more Sensors 92 can be used to detect objects and/or their properties in Device's 98 surrounding. In some aspects, Device's 98 surrounding may include exterior of Device 98. In other aspects, Device's 98 surrounding may include interior of Device 98 in case of hollow Device 98, Device 98 comprising compartments or openings, and/or other variously shaped Device 98. Examples of aspects of an environment that Sensor 92 can measure or be sensitive to include light (i.e. camera, lidar, etc.), electromagnetism/electromagnetic field (i.e. radar, etc.), sound (i.e.

Autonomous Ex. 2004 p. 366 of 510
Tesla v. Autonomous
IPR2024-00745

microphone, sonar, etc.), physical contact (i.e. tactile sensor, etc.), magnetism/magnetic field (i.e. compass, etc.), electricity/electric field, temperature, gravity, vibration, pressure, and/or others. In some aspects, a passive sensor (i.e. camera, microphone, etc.) measures signals or radiation emitted or reflected by an object. In other aspects, an active sensor (i.e. lidar, radar, sonar, etc.) emits signals or radiation and measures the signals or radiation reflected or backscattered from an object. A reference to a Sensor 92 herein includes a reference to one or more Sensors 92 as applicable. In some designs, a plurality of Sensors 92 may be used to detect objects and/or their properties from different angles or sides of Device 98. For example, four Cameras 92a can be placed on four corners of Device 98 to cover 360 degrees of view of Device's 98 surrounding. In other designs, a plurality of different types of Sensors 92 may be used to detect different types of objects and/or their properties. For example, one or more Cameras 92a can be used to detect and identify an object, whereas, Radar 92d can be used to determine distance and bearing/angle of the object relative to Device 98. In further designs, a signal-emitting element can be placed within or onto an object and Sensor 92 can detect the signal from the signal-emitting element, thereby detecting the object and/or its properties. For example, a radio-frequency identification (RFID) emitter may be placed within an object to help Sensor 92 detect, identify, and/or obtain other information about the object.

In some embodiments, Sensor 92 may be or include Camera 92a as shown in Fig. 3A. Camera 92a comprises the functionality for capturing one or more pictures, and/or other functionalities. As such, Camera 92a can be used to capture pictures of Device's 98 surrounding. Camera 92a may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Camera 92a may be or comprises a motion picture camera that can capture streams of pictures (i.e. motion pictures, videos, etc.). In other aspects, Camera 92a may be or comprises a still picture camera that can capture still pictures (i.e. photographs, etc.). In further aspects, Camera 92a may be or comprises a stereo camera (i.e. camera with multiple lenses, etc.) that can capture stereoscopic or range pictures. In further aspects, Camera 92a may be or comprises any other Camera 92a. In general, Camera 92a may capture any light (i.e. visible light, infrared light, ultraviolet light, x-ray light, etc.) across the electromagnetic spectrum onto a light-sensitive material. Any other technique known in art can be utilized to facilitate Camera 92a functionalities. In one example, a digital Camera 92a can utilize a charge coupled device (CCD), a complementary metal–oxide–semiconductor (CMOS) sensor, and/or other electronic image sensor to capture digital pictures that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Camera 92a can utilize an analog-to-digital converter to produce digital pictures. In some embodiments, Camera 92a can be built, embedded, or integrated in Device 98 and/or other disclosed element. In other embodiments, Camera 92a can be an external Camera 92a connected with Device 98 and/or other disclosed element. In further embodiments, Camera 92a comprises Computing Device 70 or elements thereof. In general, Camera 92a can be implemented in any suitable configuration to provide its functionalities. Camera 92a may capture one or more digital pictures. A digital picture may include a collection of color encoded pixels or dots. Examples of file formats that can be utilized to store a digital picture include JPEG, GIF, TIFF, PNG, PDF, and/or other digitally encoded picture formats. A stream of digital pictures (i.e. motion picture, video, etc.) may include one or more digital pictures. Examples of file formats that can be utilized to store a stream of digital pictures include MPEG, AVI, FLV, MOV, RM, SWF, WMV, DivX, and/or other digitally encoded motion picture formats.

Autonomous Ex. 2004 p. 367 of 510
Tesla v. Autonomous
IPR2024-00745

In other embodiments, Sensor 92 may be or include Microphone 92b as shown in Fig. 3B. Microphone 92b comprises the functionality for capturing one or more sounds, and/or other functionalities. As such, Microphone 92b can be used to capture sounds from Device's 98 surrounding. Microphone 92b may be useful in detecting existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In some aspects, Microphone 92b may be omnidirectional microphone that enables capturing sounds from any direction. In other aspects, Microphone 92b may be a directional (i.e. unidirectional, bidirectional, etc.) microphone that enables capturing sounds from one or more directions while ignoring or being insensitive to sounds from other directions. In general, Microphone 92b may utilize a membrane sensitive to air pressure and may produce electrical signal from air pressure variations. Samples of the electrical signal can then be read to produce a stream of digital sound samples. Any other technique known in art can be utilized to facilitate Microphone 92b functionalities. In one example, a digital Microphone 92b may include an integrated analog-to-digital converter to capture a stream of digital sound samples that can then be stored in a memory or storage, or transmitted to any of the disclosed or other elements for further processing. In another example, analog Microphone 92b may utilize an external analog-to-digital converter to produce a stream of digital sound samples. In some embodiments, Microphone 92b can be built, embedded, or integrated in Device 98. In other embodiments, Microphone 92b can be an external Microphone 92b connected with Device 98. In further embodiments where used in water, Microphone 92b may be or include a hydrophone. In further embodiments, Microphone 92b comprises Computing Device 70 or elements thereof. In general, Microphone 92b can be implemented in any suitable configuration to provide its functionalities. Examples of file formats that can be utilized to store a stream of digital sound samples include WAV, WMA, AIFF, MP3, RA, OGG, and/or other digitally encoded sound formats.

In further embodiments, Sensor 92 may be or include Lidar 92c as shown in Fig. 3C. Lidar 92c may be useful in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Lidar 92c may emit a light signal (i.e. laser beam, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Lidar 92c functionalities.

In further embodiments, Sensor 92 may be or include a Radar 92d as shown in Fig. 3D. Radar 92d may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Radar 92d may emit a radio signal (i.e. radio wave, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Radar 92d functionalities.

In further embodiments, Sensor 92 may be or include Sonar 92e as shown in Fig. 3E. Sonar 92e may be useful in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In some aspects, Sonar 92e may emit a sound signal (i.e. sound pulse, etc.) and listen for a signal that is reflected or backscattered from an object. Any other technique known in art can be utilized to facilitate Sonar 92e functionalities.

One of ordinary skill in art will understand that the aforementioned sensors are described merely as examples of a variety of possible implementations, and that while all possible sensors are too voluminous to describe, other sensors known in art that can facilitate detecting of objects and/or their properties in Device's 98

Autonomous Ex. 2004 p. 368 of 510
Tesla v. Autonomous
IPR2024-00745

surrounding are within the scope of this disclosure. Any combination of the aforementioned and/or other sensors can be used in various embodiments.

Object Processing Unit 93 comprises the functionality for processing output from Sensor 92 to obtain information of interest, and/or other functionalities. As such, Object Processing Unit 93 can be used to process output from Sensor 92 to detect objects and/or their properties in Device's 98 surrounding. In some embodiments, Object Processing Unit 93 comprises the functionality for creating or generating Collection of Object Representations 525 (also referred to as Coll of Obj Rep or other suitable name or reference) and storing one or more Object Representations 625 (also referred to simply as object representations, representations of objects, or other suitable name or reference), Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information into the Collection of Object Representations 525. As such, Collection of Object Representations 525 comprises the functionality for storing one or more Object Representations 625, Object Properties 630, and/or other elements or information. Object Representation 625 may include an electronic representation of an object (i.e. Object 615 [later described], etc.) detected in Device's 98 surrounding. In some aspects, Collection of Object Representations 525 includes one or more Object Representations 625, Object Properties 630, and/or other elements or information related to objects detected in Device's 98 surrounding at a particular time. Collection of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties at a particular time. In some designs, a Collection of Object Representations 525 may include or be associated with a time stamp (not shown), order (not shown), or other time related information. For example, one Collection of Object Representations 525 may be associated with time stamp t1, another Collection of Object Representations 525 may be associated with time stamp t2, and so on. Time stamps t1, t2, etc. may indicate the times of generating Collections of Object Representations 525, for instance. In other embodiments, Object Processing Unit 93 comprises the functionality for creating or generating a stream of Collections of Object Representations 525. A stream of Collections of Object Representations 525 may include one Collection of Object Representations 525 or a group, sequence, or other plurality of Collections of Object Representations 525. In some aspects, a stream of Collections of Object Representations 525 includes one or more Collections of Object Representations 525, and/or other elements or information related to objects detected in Device's 98 surrounding over time. A stream of Collections of Object Representations 525 may, therefore, include knowledge (i.e. unit of knowledge, etc.) of Device's 98 circumstances including objects with various properties over time. As circumstances including objects with various properties in Device's 98 surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this change may be captured in a stream of Collections of Object Representations 525. In some designs, each Collection of Object Representations 525 in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. For example, one Collection of Object Representations 525 in a stream may be associated with order 1, a next Collection of Object Representations 525 in the stream may be associated with order 2, and so on. Orders 1, 2, etc. may indicate the orders or places of Collections of Object Representations 525 within a stream (i.e. sequence, etc.), for instance. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as

Autonomous Ex. 2004 p. 369 of 510
Tesla v. Autonomous
IPR2024-00745

an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. Type of an object, for example, may include any classification of objects ranging from detailed such as person, cat, vehicle, building, street, tree, rock, etc. to generalized such as biological object, nature object, manmade object, etc., and/or others including their sub-types. Location of an object, for example, can include a relative location such as one defined by distance and bearing/angle from a known point or location (i.e. Device 98 location, etc.). Location of an object, for example, can also include absolute location such as one defined by object coordinates. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with Device 98, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. In some implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be included in Sensor 92. In other implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate on Processor 11. In further implementations, Object Processing Unit 93 and/or any of its elements or functionalities can be embedded into or operate in DCADO Unit 100, and/or other disclosed elements. Object Processing Unit 93 may be provided in any suitable configuration. Object Processing Unit 93 may include any signal processing techniques or elements known in art as applicable.

In some embodiments, Object Processing Unit 93 may include Picture Recognizer 94a as shown in Fig. 3A. Picture Recognizer 94a comprises the functionality for detecting or recognizing objects and/or their properties in visual data, and/or other disclosed functionalities. Visual data includes digital motion pictures, digital still pictures, and/or other visual data. Examples of file formats that can be utilized to store visual data include AVI, DivX, MPEG, JPEG, GIF, TIFF, PNG, PDF, and/or other file formats. For example, Picture Recognizer 94a can be used for detecting or recognizing objects and/or their properties in one or more digital pictures captured by one or more Cameras 92a. Picture Recognizer 94a can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Picture Recognizer 94a can be used for any operation supported by Picture Recognizer 94a. Picture Recognizer 94a may detect or recognize an object and/or its properties as well as track the object and/or its properties in one or more digital pictures or streams of digital pictures (i.e. motion pictures, video, etc.). In the case of a person, Picture Recognizer 94a may detect or recognize a human head or face, upper body, full body, or portions/combinations thereof. In some aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing regions of pixels from the digital picture with collections of pixels comprising known objects and/or their properties. The collections of pixels comprising known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The collections of pixels comprising known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Picture

Autonomous Ex. 2004 p. 370 of 510
Tesla v. Autonomous
IPR2024-00745

Recognizer 94a may detect or recognize objects and/or their properties from a digital picture by comparing features (i.e. lines, edges, ridges, corners, blobs, regions, etc.) of the digital picture with features of known objects and/or their properties. The features of known objects and/or their properties can be learned or manually, programmatically, or otherwise defined. The features of known objects and/or their properties can be stored in any data structure or repository (i.e. neural network, one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. Typical steps or elements in a feature oriented picture recognition include pre-processing, feature extraction, detection/segmentation, decision-making, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Picture Recognizer 94a may detect or recognize multiple objects and/or their properties from a digital picture using the aforementioned pixel or feature comparisons, and/or other detection or recognition techniques. For example, a picture may depict two objects in two of its regions both of which Picture Recognizer 94a can detect simultaneously. In further aspects, where objects and/or their properties span multiple pictures, Picture Recognizer 94a may detect or recognize objects and/or their properties by applying the aforementioned pixel or feature comparisons and/or other detection or recognition techniques over a stream of digital pictures (i.e. motion picture, video, etc.). For example, once an object is detected in a digital picture (i.e. frame, etc.) of a stream of digital pictures (i.e. motion picture, video, etc.), the region of pixels comprising the detected object or the object's features can be searched in other pictures of the stream of digital pictures, thereby tracking the object through the stream of digital pictures. In further aspects, Picture Recognizer 94a may detect or recognize an object's activities by identifying and/or analyzing differences between a detected region of pixels of one picture (i.e. frame, etc.) and detected regions of pixels of other pictures in a stream of digital pictures. For example, a region of pixels comprising a person's face can be detected in multiple consecutive pictures of a stream of digital pictures (i.e. motion picture, video, etc.). Differences among the detected regions of the consecutive pictures may be identified in the mouth part of the person's face to indicate smiling or speaking activity. In further aspects, Picture Recognizer 94a may detect or recognize objects and/or their properties using one or more artificial neural networks, which may include statistical techniques. Examples of artificial neural networks that can be used in Picture Recognizer 94a include convolutional neural networks (CNNs), time delay neural networks (TDNNs), deep neural networks, and/or others. In one example, picture recognition techniques and/or tools involving convolutional neural networks may include identifying and/or analyzing tiled and/or overlapping regions or features of a digital picture, which may then be used to search for pictures with matching regions or features. In another example, features of different convolutional neural networks responsible for spatial and temporal streams can be fused to detect objects and/or their properties in streams of digital pictures (i.e. motion pictures, videos, etc.). In general, Picture Recognizer 94a may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Picture Recognizer 94a can detect distance of a recognized object in a picture captured by a camera using structured light, sheet of light, or other lighting schemes, and/or by using phase shift analysis, time of flight, interferometry, or other techniques. In further aspects, Picture Recognizer 94a may detect distance of a recognized object in a picture captured by a stereo camera by using triangulation and/or other techniques. In further aspects, Picture Recognizer 94a may detect bearing/angle of a recognized object relative to the camera-facing direction by measuring the distance from the vertical centerline of the picture to a pixel in the recognized object based on known picture resolution and camera's angle of view. Any other techniques known in art can be utilized in Picture

Autonomous Ex. 2004 p. 371 of 510
Tesla v. Autonomous
IPR2024-00745

Recognizer 94a. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, object recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, CamFind API, Kooaba, 6px API, Dextro API, and/or others can be utilized for detecting or recognizing objects and/or their properties in digital pictures. In some aspects, picture recognition techniques and/or tools involve identifying and/or analyzing features such as lines, edges, ridges, corners, blobs, regions, and/or their relative positions, sizes, shapes, etc., which may then be used to search for pictures with matching features. For example, OpenCV library can detect an object (i.e. person, animal, vehicle, rock, etc.) and/or its properties in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, facial recognition techniques and/or tools such as OpenCV (Open Source Computer Vision) library, Animetrics FaceR API, Lambda Labs Facial Recognition API, Face++ SDK, Neven Vision (also known as N-Vision) Engine, and/or others can be utilized for detecting or recognizing faces in digital pictures. In some aspects, facial recognition techniques and/or tools involve identifying and/or analyzing facial features such as the relative position, size, and/or shape of the eyes, nose, cheekbones, jaw, etc., which may then be used to search for pictures with matching features. For example, FaceR API can detect a person's face in one or more digital pictures captured by Camera 92a or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

Various aspects or properties of digital pictures or pixels can be taken into account by Picture Recognizer 94a in any of the recognizing or comparisons. Examples of such aspects or properties include color adjustment, size adjustment, content manipulation, transparency (i.e. alpha channel, etc.), use of mask, and/or others. In some implementations, as digital pictures can be captured by various picture taking equipment, in various environments, and under various lighting conditions, Picture Recognizer 94a can adjust lighting or color of pixels or otherwise manipulate pixels before or during comparison. Lighting or color adjustment (also referred to as gray balance, neutral balance, white balance, etc.) may generally include manipulating or rebalancing the intensities of the colors (i.e. red, green, and/or blue if RGB color model is used, etc.) of one or more pixels. For example, Picture Recognizer 94a can adjust lighting or color of some or all pixels of one picture to make it more comparable to another picture. Picture Recognizer 94a can also incrementally adjust the pixels such as increasing or decreasing the red, green, and/or blue pixel values by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental adjustment levels. Any of the publically available, custom, or other lighting or color adjustment techniques or programs can be utilized such as color filters, color balancing, color correction, and/or others. In other implementations, Picture Recognizer 94a can resize or otherwise transform a digital picture before or during comparison. Such resizing or transformation may include increasing or decreasing the number of pixels of a digital picture. For example, Picture Recognizer 94a can increase or decrease the size of a digital picture proportionally (i.e. increase or decrease length and/or width keeping aspect ratio constant, etc.) to equate its size with the size of another digital picture. Picture Recognizer 94a can also incrementally resize a digital picture such as increasing or decreasing the size of the digital picture proportionally by a certain amount in each cycle of comparisons in order to find a substantially similar match at one of the incremental sizes. Any of the publically available, custom, or other digital picture resizing techniques or programs can be utilized such as nearest-neighbor

Autonomous Ex. 2004 p. 372 of 510
Tesla v. Autonomous
IPR2024-00745

interpolation, bilinear interpolation, bicubic interpolation, and/or others. In further implementations, Picture Recognizer 94a can manipulate content (i.e. all pixels, one or more regions, one or more depicted objects, etc.) of a digital picture before or during comparison. Such content manipulation may include moving, centering, aligning, resizing, transforming, and/or otherwise manipulating content of a digital picture. For example, Picture Recognizer 94a can move, center, or align content of one picture to make it more comparable to another picture. Any of the publically available, custom, or other digital picture manipulation techniques or programs can be utilized such as pixel moving, warping, distorting, aforementioned interpolations, and/or others. In further implementations, in digital pictures comprising transparency features or functionalities, Picture Recognizer 94a can utilize a threshold for acceptable number or percentage transparency difference. Alternatively, transparency can be applied to one or more pixels of a digital picture and color difference may then be determined between compared pixels taking into account the transparency related color effect. Alternatively, transparent pixels can be excluded from comparison. In further implementations, certain regions or subsets of pixels can be ignored or excluded during comparison using a mask. In general, any region or subset of a picture determined to contain no content of interest can be excluded from comparison using a mask. Examples of such regions or subsets include background, transparent or partially transparent regions, regions comprising insignificant content, or any arbitrary region or subset. Picture Recognizer 94a can perform any other pre-processing or manipulation of digital pictures or pixels before or during recognizing or comparison.

In other embodiments, Object Processing Unit 93 may include Sound Recognizer 94b as shown in Fig. 3B. Sound Recognizer 94b comprises the functionality for detecting or recognizing objects and/or their properties in audio data, and/or other disclosed functionalities. Audio data includes digital sound, and/or other audio data. Examples of file formats that can be utilized to store audio data include WAV, WMA, AIFF, MP3, RA, OGG, and/or other file formats. For example, Sound Recognizer 94b can be used for detecting or recognizing objects and/or their properties in a stream of digital sound samples captured by one or more Microphones 92b. In the case of a person, Sound Recognizer 94b may detect or recognize human voice. Sound Recognizer 94b can be utilized in detecting or recognizing existence of an object, type of an object, identity of an object, bearing/angle of an object, activity (i.e. motion, sounding, etc.) of an object, and/or other properties of an object. In general, Sound Recognizer 94b can be used for any operation supported by Sound Recognizer 94b. In some aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing collections of sound samples from the stream of digital sound samples with collections of sound samples of known objects and/or their properties. The collections of sound samples of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The collections of sound samples of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface. In other aspects, Sound Recognizer 94b may detect or recognize an object and/or its properties from a stream of digital sound samples by comparing features from the stream of digital sound samples with features of sounds of known objects and/or their properties. The features of sounds of known objects and/or their properties can be learned, or manually, programmatically, or otherwise defined. The features of sounds of known objects and/or their properties can be stored in any data structure or repository (i.e. one or more files, database, neural network, etc.) that resides locally on Device 98, or remotely on a remote computing device (i.e. server, cloud, etc.) accessible over

Autonomous Ex. 2004 p. 373 of 510
Tesla v. Autonomous
IPR2024-00745

a network or an interface. Typical steps or elements in a feature oriented sound recognition include pre-processing, feature extraction, acoustic modeling, language modeling, and/or others, or a combination thereof, each of which may include its own sub-steps or sub-elements depending on the application. In further aspects, Sound Recognizer 94b may detect or recognize a variety of sounds from a stream of digital sound samples using the aforementioned sound sample or feature comparisons, and/or other detection or recognition techniques. For example, sound of a person, animal, vehicle, and/or other sounds can be detected by Sound Recognizer 94b. In further aspects, Sound Recognizer 94b may detect or recognize sounds using Hidden Markov Models (HMM), Artificial Neural Networks, Dynamic Time Warping (DTW), Gaussian Mixture Models (GMM), and/or other models or techniques, or a combination thereof. Some or all of these models or techniques may include statistical techniques. Examples of artificial neural networks that can be used in Sound Recognizer 94b include recurrent neural networks, time delay neural networks (TDNNs), deep neural networks, convolutional neural networks, and/or others. In general, Sound Recognizer 94b may include any machine learning, deep learning, and/or other artificial intelligence techniques. In further aspects, Sound Recognizer 94b may detect bearing/angle of a recognized object by measuring the direction in which Microphone 92b is pointing when sound of maximum strength is received, by analyzing amplitude of the sound, by performing phase analysis (i.e. with microphone array, etc.) of the sound, and/or by utilizing other techniques. Any other techniques known in art can be utilized in Sound Recognizer 94b. For example, thresholds for similarity, statistical techniques, and/or optimization techniques can be utilized to determine a match in any of the above-described detection or recognition techniques. In some exemplary embodiments, operating system's Sound recognition functionalities such as iOS's Voice Services, Siri, and/or others can be utilized in Sound Recognizer 94b. For example, iOS Voice Services can detect an object (i.e. person, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. In other exemplary embodiments, Java Speech API (JSAPI) implementation such as The Cloud Garden, Sphinx, and/or others can be utilized in Sound Recognizer 94b. For example, Cloud Garden JSAPI can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements. Any other programming language's or platform's speech or sound processing API can similarly be utilized. In further exemplary embodiments, applications or engines providing Sound recognition functionalities such as HTK (Hidden Markov Model Toolkit), Kaldi, OpenEars, Dragon Mobile, Julius, iSpeech, CeedVocal, and/or others can be utilized in Sound Recognizer 94b. For example, Kaldi SDK can detect an object (i.e. person, animal, vehicle, etc.) and/or its properties in a stream of digital sound samples captured by Microphone 92b or stored in an electronic repository, which can then be utilized in DCADO Unit 100, Artificial Intelligence Unit 110, and/or other elements.

In further embodiments, Object Processing Unit 93 may include Lidar Processing Unit 94c as shown in Fig. 3C. Lidar Processing Unit 94c comprises the functionality for detecting or recognizing objects and/or their properties using light, and/or other disclosed functionalities. As such, Lidar Processing Unit 94c can be utilized in detecting existence of an object, type of an object, identity of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Lidar Processing Unit 94c can be used for any operation supported by Lidar Processing Unit 94c. In one example, Lidar Processing Unit 94c may detect distance of an object by measuring time delay between emission of a light

Autonomous Ex. 2004 p. 374 of 510
Tesla v. Autonomous
IPR2024-00745

signal (i.e. laser beam, etc.) and return of the light signal reflected from the object based on known speed of light. In another example, Lidar Processing Unit 94c may detect bearing/angle of an object by analyzing the amplitudes of a light signal received by an array of detectors (i.e. detectors arranged into a quadrant or other arrangement, etc.). In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. In a further example, Lidar Processing Unit 94c may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with light and acquiring a point cloud representation of the object. Lidar Processing Unit 94c may detect objects and/or their properties by utilizing any lidar or light-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Radar Processing Unit 94d as shown in Fig. 3D. Radar Processing Unit 94d comprises the functionality for detecting or recognizing objects and/or their properties using radio waves, and/or other disclosed functionalities. As such, Radar Processing Unit 94d can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Radar Processing Unit 94d can be used for any operation supported by Radar Processing Unit 94d. In one example, Radar Processing Unit 94d may detect existence of an object by emitting a radio signal and listening for the radio signal reflected from the object. In another example, Radar Processing Unit 94d may detect distance of an object by measuring time delay between emission of a radio signal and return of the radio signal reflected from the object based on known speed of the radio signal. In a further example, Radar Processing Unit 94d may detect bearing/angle of an object by measuring the direction in which the antenna is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with antenna array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Radar Processing Unit 94d may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with radio waves and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Radar Processing Unit 94d may detect objects and/or their properties by utilizing any radar or radio-related techniques known in art.

In further embodiments, Object Processing Unit 93 may include Sonar Processing Unit 94e as shown in Fig. 3E. Sonar Processing Unit 94e comprises the functionality for detecting or recognizing objects and/or their properties using sound, and/or other disclosed functionalities. As such, Sonar Processing Unit 94e can be utilized in detecting existence of an object, type of an object, distance of an object, bearing/angle of an object, location of an object, shape/size of an object, activity of an object, and/or other properties of an object. In general, Sonar Processing Unit 94e can be used for any operation supported by Sonar Processing Unit 94e. In one example, Sonar Processing Unit 94e may detect existence of an object by emitting a sound signal and listening for the sound signal reflected from the object. In another example, Sonar Processing Unit 94e may detect distance of an object by measuring time delay between emission of a sound signal and return of the sound signal reflected from the object based on known speed of the sound signal. In a further example, Sonar Processing Unit 94e may detect bearing/angle of an object by measuring the direction in which the microphone is pointing when the return signal of maximum strength is received, by analyzing amplitude of the return signal, by performing phase analysis (i.e. with

Autonomous Ex. 2004 p. 375 of 510
Tesla v. Autonomous
IPR2024-00745

microphone array, etc.) of the return signal, and/or by utilizing any amplitude, phase, or other techniques. In a further example, Sonar Processing Unit 94e may detect existence, type, identity, shape/size, activity, and/or other properties of an object by illuminating the object with sound pulses and acquiring an image of the object, which can then be processed using some of the previously described or other picture recognition techniques. Sonar Processing Unit 94e may detect objects and/or their properties by utilizing any sonar or sound-related techniques known in art.

One of ordinary skill in art will understand that the aforementioned techniques for detecting or recognizing objects and/or their properties are described merely as examples of a variety of possible implementations, and that while all possible techniques for detecting or recognizing objects and/or their properties are too voluminous to describe, other techniques for detecting or recognizing objects and/or their properties known in art are within the scope of this disclosure. Also, any signal processing technique known in art that can facilitate the disclosed functionalities can be utilized in various embodiments. Any combination of the aforementioned and/or other sensors, object detecting or recognizing techniques, signal processing techniques, and/or other elements or techniques can be used in various embodiments.

Referring to Figs. 4A-4B, an exemplary embodiment of Objects 615 (also referred to simply as objects or other suitable name or reference) detected in Device's 98 surrounding, and resulting Collection of Object Representations 525 are illustrated.

As shown for example in Fig. 4A, Object 615a is detected. Object 615a may be recognized as a cat. Object 615a may be detected at a distance of 6m from Device 98. Object 615a may be detected at a bearing/angle of 56° from Device's 98 centerline. Furthermore, Object 615b is also detected. Object 615b may be recognized as a tree. Object 615b may be detected at a distance of 10m from Device 98. Object 615b may be detected at a bearing/angle of 131° from Device's 98 centerline. Furthermore, Object 615c is also detected. Object 615c may be recognized as a person. Object 615c may be identified as John Doe. Object 615c may be detected at a distance of 8m from Device 98. Object 615c may be detected at a bearing/angle of 287° from Device's 98 centerline. Any other Objects 615 instead of or in addition to Object 615a, Object 615b, and Object 615c may be detected. In some aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, and/or other sensors or techniques can be utilized for recognizing and/or identifying a person, a cat, a tree, and/or other Objects 615. In further aspects, any features, functionalities, and embodiments of Camera 92a/Picture Recognizer 94a, Microphone 92b/Sound Recognizer 94b, Lidar 92c/Lidar Processing Unit 94c, Radar 92d/Radar Processing Unit 94d, Sonar 92e/Sonar Processing Unit 94e, and/or other sensors or techniques can be utilized for detecting distance, bearing/angle, and/or other object properties.

As shown for example in Fig. 4B, Object Processing Unit 93 may create or generate Collection of Object Representations 525 including Object Representation 625a representing Object 615a, Object Representation 625b representing Object 615b, Object Representation 625c representing Object 615c, etc. For instance, Object Representation 625a may include Object Property 630aa "Cat" in Category 635aa "Type", Object Property 630ab "6m" in Category 635ab "Distance", Object Property 630ac "56°" in Category 635ac "Bearing", etc. Also, Object Representation 625b may include Object Property 630ba "Tree" in Category 635ba "Type", Object Property 630bb "10m" in Category 635bb "Distance", Object Property 630bc "131°" in Category 635bc "Bearing", etc. Also, Object Representation 625c may include Object Property 630ca "Person" in Category 635ca "Type", Object Property 630cb "John Doe" in Category 635cb "Identity", Object Property 630cc "8m" in Category 635cc "Distance", Object Property

Autonomous Ex. 2004 p. 376 of 510
Tesla v. Autonomous
IPR2024-00745

630cd "287°" in Category 635cd "Bearing", etc. Any number of Object Representations 625, and/or other elements or information can be included in Collection of Object Representations 525. Any number of Object Properties 630 (also referred to simply as object properties or other suitable name or reference), and/or other elements or information can be included in an Object Representation 625. In some aspects, a reference to Collection of Object Representations 525 comprises a reference to a collection of Object Properties 630 and/or other elements or information related to one or more Objects 615. Other additional Object Representations 625, Object Properties 630, elements, and/or information can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments of Collection of Object Representations 525.

Referring now to DCADO Unit 100, DCADO Unit 100 comprises any hardware, programs, or a combination thereof. DCADO Unit 100 comprises the functionality for learning the operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for structuring and/or storing this knowledge in a knowledgebase (i.e. neural network, graph, sequences, other repository, etc.). DCADO Unit 100 comprises the functionality for enabling autonomous operation of Device 98 in circumstances including objects with various properties. DCADO Unit 100 comprises the functionality for interfacing with or attaching to Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element. DCADO Unit 100 comprises the functionality for obtaining instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises the functionality for modifying instruction sets, data, and/or other information used, implemented, and/or executed by Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. DCADO Unit 100 comprises learning, anticipating, decision making, automation, and/or other functionalities disclosed herein. Statistical, artificial intelligence, machine learning, and/or other models or techniques are utilized to implement the disclosed devices, systems, and methods.

When DCADO Unit 100 functionalities are applied on Application Program 18, Processor 11, Logic Circuit 250 (later described), and/or other processing element of Device 98, Device 98 may become autonomous. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element to implement autonomous operation of Device 98. DCADO Unit 100 may take control from, share control with, and/or release control to Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element automatically or after prompting User 50 to allow it. In some aspects, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with anticipatory (also referred to as alternate or other suitable name or reference) instructions or instruction sets that User 50 did not issue or cause to be executed. Such anticipatory instructions or instruction sets include instruction sets that User 50 may want or is likely to issue or cause to be executed. Anticipatory instructions or instruction sets can be generated by DCADO Unit 100 or elements thereof based on Device's 98 circumstances including objects with various properties. As such, Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element of an autonomous Device 98 may include or be provided with some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. Therefore, autonomous Device 98 operating may include executing some or all original instructions or instruction sets and/or any anticipatory instructions or instruction sets generated by DCADO Unit 100. In one example, DCADO Unit 100 can overwrite or rewrite the original instructions or instruction sets of Application

Autonomous Ex. 2004 p. 377 of 510
Tesla v. Autonomous
IPR2024-00745

Program 18, Processor 11, Logic Circuit 250, and/or other processing element with DCADO Unit 100-generated instructions or instruction sets. In another example, DCADO Unit 100 can insert or embed DCADO Unit 100-generated instructions or instruction sets among the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, DCADO Unit 100 can branch, redirect, or jump to DCADO Unit 100-generated instructions or instruction sets from the original instructions or instruction sets of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element.

In some embodiments, autonomous Device 98 operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the user issuing or causing to be executed the next instruction or instruction set. In yet other embodiments, autonomous application operating comprises determining, by DCADO Unit 100, a next instruction or instruction set to be executed based on Device's 98 circumstances including objects with various properties prior to the system receiving the next instruction or instruction set.

In some embodiments, autonomous Device 98 operating includes a partially or fully autonomous operating. In an example involving partially autonomous Device 98 operating, a user confirms DCADO Unit 100-generated instructions or instruction sets prior to their execution. In an example involving fully autonomous application operating, DCADO Unit 100-generated instructions or instruction sets are executed without user or other system confirmation (i.e. automatically, etc.).

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Device's 98 operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Device's 98 operation. While operating autonomously under the control of DCADO Unit 100, Device 98 may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Device's 98 operation. DCADO Unit 100 may take control again when Device 98 encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Device's 98 operation in circumstances while User 50 and/or non-DCADO system is in control of Device 98, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Device's 98 operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Device 98 can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Device 98 leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Device 98 while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Device 98.

Autonomous Ex. 2004 p. 378 of 510
Tesla v. Autonomous
IPR2024-00745

It should be understood that a reference to autonomous operating of Device 98 may include autonomous operating of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element depending on context.

Referring now to Acquisition Interface 120, Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information from Processor 11, Application Program 18, Logic Circuit 250 (later described), and/or other processing element. Acquisition Interface 120 comprises the functionality for obtaining and/or receiving instruction sets, data, and/or other information at runtime. In some aspects, an instruction set may include any computer command, instruction, signal, or input used in Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Therefore, the terms instruction set, command, instruction, signal, input, or other such terms may be used interchangeably herein depending on context. Acquisition Interface 120 also comprises the functionality for attaching to or interfacing with Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In one example, Acquisition Interface 120 comprises the functionality to access and/or read runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Acquisition Interface 120 comprises the functionality to access and/or read memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read Processor 11 registers and/or other Processor 11 elements. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Acquisition Interface 120 comprises the functionality to access and/or read values, variables, parameters, and/or other data or information. Acquisition Interface 120 also comprises the functionality for transmitting the obtained instruction sets, data, and/or other information to Artificial Intelligence Unit 110 and/or other element. As such, Acquisition Interface 120 provides input into Artificial Intelligence Unit 110 for knowledge structuring, anticipating, decision making, and/or other functionalities later in the process. Acquisition Interface 120 also comprises other disclosed functionalities.

Acquisition Interface 120 can employ various techniques for obtaining instruction sets, data, and/or other information. In one example, Acquisition Interface 120 can attach to and/or obtain Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information through tracing or profiling techniques. Tracing or profiling may be used for outputting Processor's 11, Application Program's 18, Logic Circuit's 250, and/or other processing element's instruction sets, data, and/or other information at runtime. For instance, tracing or profiling may include adding trace code (i.e. instrumentation, etc.) to an application and/or outputting trace information to a specific target. The outputted trace information (i.e. instruction sets, data, and/or other information, etc.) can then be provided to or recorded into a file, data structure, repository, an application, and/or other system or target that may receive such trace information. As such, Acquisition Interface

Autonomous Ex. 2004 p. 379 of 510
Tesla v. Autonomous
IPR2024-00745

120 can utilize tracing or profiling to obtain instruction sets, data, and/or other information and provide them as input into Artificial Intelligence Unit 110. In some aspects, instrumentation can be performed in source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In other aspects, instrumentation can be performed in various elements of a computing system such as memory, virtual machine, runtime engine/environment, operating system, compiler, interpreter, translator, processor registers, execution stack, program counter, and/or other elements. In yet other aspects, instrumentation can be performed in various abstraction layers of a computing system such as in software layer (i.e. Application Program 18, etc.), in virtual machine (if VM is used), in operating system, in Processor 11, and/or in other layers or areas that may exist in a particular computing system implementation. In yet other aspects, instrumentation can be performed at various time periods in an application's execution such as source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, and/or other time periods. In yet other aspects, instrumentation can be performed at various granularities or code segments such as some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, some or all functions/routines/subroutines, and/or some or all other code segments.

In some embodiments, Application Program 18 can be automatically instrumented. For example, Acquisition Interface 120 can access Application Program's 18 source code, bytecode, or machine code and select instrumentation points of interest. Selecting instrumentation points may include finding locations in the source code, bytecode, or machine code corresponding to function calls, function entries, function exits, object creations, object destructions, event handler calls, new lines (i.e. to instrument all lines of code, etc.), thread creations, throws, and/or other points of interest. Instrumentation code can then be inserted at the instrumentation points of interest to output Application Program's 18 instruction sets, data, and/or other information. In response to executing instrumentation code, Application Program's 18 instruction sets, data, and/or other information may be received by Acquisition Interface 120. In some aspects, Application Program's 18 source code, bytecode, or machine code can be dynamically instrumented. For example, instrumentation code can be dynamically inserted into Application Program 18 at runtime.

In other embodiments, Application Program 18 can be manually instrumented. In one example, a programmer can instrument a function call by placing an instrumenting instruction immediately after the function call as in the following example.

```
Device1.moveForward(12);
traceApplication('Device1.moveForward(12);');
```

In another example, an instrumenting instruction can be placed immediately before the function call, or at the beginning, end, or anywhere within the function itself. A programmer may instrument all function calls or only function calls of interest. In a further example, a programmer can instrument all lines of code or only code lines of interest. In a further example, a programmer can instrument other elements utilized or implemented within Application Program 18 such as objects and/or any of their functions, data structures and/or any of their functions, event handlers and/or any of their functions, threads and/or any of their functions, and/or other elements or functions. Similar instrumentation as in the preceding examples can be performed automatically or dynamically. In some designs where manual code instrumentation is utilized, Acquisition Interface 120 can optionally be omitted and Application Program's 18 instruction sets, data, and/or other information may be transmitted directly to Artificial

Autonomous Ex. 2004 p. 380 of 510
Tesla v. Autonomous
IPR2024-00745

Intelligence Unit 110.

In some embodiments, DCADO Unit 100 can be selective in learning instruction sets, data, and/or other information to those implemented, utilized, or related to an object, data structure, repository, thread, function, and/or other element of Application Program 18. In some aspects, Acquisition Interface 120 can obtain Application Program's 18 instruction sets, data, and/or other information implemented, utilized, or related to a certain object in an object oriented Application Program 18.

In some embodiments, various computing systems and/or platforms may provide native tools for obtaining instruction sets, data, and/or other information. Also, independent vendors may provide portable tools with similar functionalities that can be utilized across different computing systems and/or platforms. These native and portable tools may provide a wide range of functionalities to obtain runtime and other information such as instrumentation, tracing or profiling, logging application or system messages, outputting custom text messages, outputting objects or data structures, outputting functions/routines/subroutines or their invocations, outputting variable or parameter values, outputting thread or process behaviors, outputting call or other stacks, outputting processor registers, providing runtime memory access, providing inputs and/or outputs, performing live application monitoring, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques to obtain instruction sets, data, and/or other information are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's native tools for application tracing or profiling such as System.Diagnostics.Trace, System.Diagnostics.Debug, and System.Diagnostics.TraceSource classes for tracing execution flow, and System.Diagnostics.Process, System.Diagnostics.EventLog, and System.Diagnostics.PerformanceCounter classes for profiling code, accessing local and remote processes, starting and stopping system processes, and interacting with Windows event logs, etc. For instance, a set of trace switches can be created that output an application's information. The switches can be configured using the .config file. For a Web application, this may typically be Web.config file associated with the project. In a Windows application, this file may typically be named applicationName.exe.config. Trace code can be added to application code automatically or manually as previously described. Appropriate listener can be created where the trace output is received. Trace code may output trace messages to a specific target such as a file, a log, a database, an object, a data structure, and/or other repository or system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can then read or obtain the trace information from these targets. In some aspects, trace code may output trace messages directly to Acquisition Interface 120. In other aspects, trace code may output trace messages directly to Artificial Intelligence Unit 110. In the case of outputting trace messages to Acquisition Interface 120 or directly to Artificial Intelligence Unit 110, custom listeners can be built to accommodate these specific targets. Other platforms, tools, and/or techniques can provide equivalent or similar functionalities as the above described ones.

In another example, obtaining instruction sets, data, and/or other information can be implemented through the .NET platform's Profiling API that can be used to create a custom profiler application for tracing, monitoring, interfacing with, and/or managing a profiled application. The Profiling API provides an interface that includes methods to notify the profiler of events in the profiled application. The Profiling API may also provide an interface to enable the profiler to call back into the profiled application to obtain information about the state of the profiled

Autonomous Ex. 2004 p. 381 of 510
Tesla v. Autonomous
IPR2024-00745

application. The Profiling API may further provide call stack profiling functionalities. Call stack (also referred to as execution stack, control stack, runtime stack, machine stack, the stack, etc.) includes a data structure that can store information about active subroutines of an application. The Profiling API may provide a stack snapshot method, which enables a trace of the stack at a particular point in time. The Profiling API may also provide a shadow stack method, which tracks the call stack at every instant. A shadow stack can obtain function arguments, return values, and information about generic instantiations. A function such as FunctionEnter can be utilized to notify the profiler that control is being passed to a function and can provide information about the stack frame and function arguments. A function such as FunctionLeave can be utilized to notify the profiler that a function is about to return to the caller and can provide information about the stack frame and function return value. An alternative to call stack profiling includes call stack sampling in which the profiler can periodically examine the stack. In some aspects, the Profiling API enables the profiler to change the in-memory code stream for a routine before it is just-in-time (JIT) compiled where the profiler can dynamically add instrumentation code to all or particular routines of interest. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through Java platform's APIs for application tracing or profiling such as Java Virtual Machine Profiling Interface (JVMPI), Java Virtual Machine Tool Interface (JVMTI), and/or other APIs or tools. These APIs can be used for instrumentation of an application, for notification of Java Virtual Machine (VM) events, and/or other functionalities. One of the tracing or profiling techniques that can be utilized includes bytecode instrumentation. The profiler can insert bytecodes into all or some of the classes. In application execution profiling, for example, these bytecodes may include methodEntry and methodExit calls. In memory profiling, for example, the bytecodes may be inserted after each new or after each constructor. In some aspects, insertion of instrumentation bytecode can be performed either by a post-compiler or a custom class loader. An alternative to bytecode instrumentation includes monitoring events generated by the JVMPI or JVMTI interfaces. Both APIs can generate events for method entry/exit, object allocation, and/or other events. In some aspects, JVMTI can be utilized for dynamic bytecode instrumentation where insertion of instrumentation bytecodes is performed at runtime. The profiler may insert the necessary instrumentation when a selected class is invoked in an application. This can be accomplished using the JVMTI's redefineClasses method, for example. This approach also enables changing of the level of profiling as the application is running. If needed, these changes can be made adaptively without restarting the application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through JVMTI's programming interface that enables creation of software agents that can monitor and control a Java application. An agent may use the functionality of the interface to register for notification of events as they occur in the application, and to query and control the application. A JVMTI agent may use JVMTI functions to extract information from a Java application. A JVMTI agent can be utilized to obtain an application's runtime information such as method calls, memory allocation, CPU utilization, lock contention, and/or other information. JVMTI may include functions to obtain information about variables, fields, methods, classes, and/or other information. JVMTI may also provide notification for numerous events such as method entry and exit, exception, field access and modification, thread start and end, and/or other events. Examples of JVMTI built-in methods include GetMethodName to obtain the name of an invoked method, GetThreadInfo to obtain information for a specific

Autonomous Ex. 2004 p. 382 of 510
Tesla v. Autonomous
IPR2024-00745

thread, GetClassSignature to obtain information about the class of an object, GetStackTrace to obtain information about the stack including information about stack frames, and/or other methods. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through java.lang.Runtime class that provides an interface for application tracing or profiling. Examples of methods provided in java.lang.Runtime that can be used to obtain an application's instruction sets, data, and/or other information include tracemethodcalls, traceinstructions, and/or other methods. These methods prompt the Java Virtual Machine to output trace information for a method or instruction in the virtual machine as it is executed. The destination of trace output may be system dependent and include a file, a listener, and/or other destinations where Acquisition Interface 120, Artificial Intelligence Unit 110, and/or other disclosed elements can access needed information. In addition to tracing or profiling tools native to their respective computing systems and/or platforms, many independent tools exist that provide tracing or profiling functionalities on more than one computing system and/or platform. Examples of these tools include Pin, DynamoRIO, KernInst, DynInst, Kprobes, OpenPAT, DTrace, SystemTap, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through logging tools of the platform and/or operating system on which an application runs. Some logging tools may include nearly full feature sets of the tracing or profiling tools previously described. In one example, Visual Basic enables logging of runtime messages through its Microsoft.VisualBasic.Logging namespace that provides a log listener where the log listener may direct logging output to a file and/or other target. In another example, Java enables logging through its java.util.logging class. In some aspects, obtaining an application's instruction sets, data, and/or other information can be implemented through logging capabilities of the operating system on which an application runs. For example, Windows NT features centralized log service that applications and operating-system components can utilize to report their events including any messages. Windows NT provides functionalities for system, application, security, and/or other logging. An application log may include events logged by applications. Windows NT, for example, may include support for defining an event source (i.e. application that created the event, etc.). Windows Vista, for example, supports a structured XML log-format and designated log types to allow applications to more precisely log events and to help interpret the events. Examples of different types of event logs include administrative, operational, analytic, debug, and/or other log types including any of their subcategories. Examples of event attributes that can be utilized include eventID, level, task, opcode, keywords, and/or other event attributes. Windows wevtutil tool enables access to events, their structures, registered event publishers, and/or their configuration even before the events are fired. Wevtutil supports capabilities such as retrieval of the names of all logs on a computing device; retrieval of configuration information for a specific log; retrieval of event publishers on a computing device; reading events from an event log, from a log file, or using a structured query; exporting events from an event log, from a log file, or using a structured query to a specific target; and/or other capabilities. Operating system logs can be utilized solely if they contain sufficient information on an application's instruction sets, data, and/or other information. Alternatively, operating system logs can be utilized in combination with another source of information (i.e. trace information, call stack, processor registers, memory, etc.) to reconstruct the application's instruction sets, data, and/or other information needed for Artificial Intelligence Unit 110 and/or other elements. In

Autonomous Ex. 2004 p. 383 of 510
Tesla v. Autonomous
IPR2024-00745

addition to logging capabilities native to their respective platforms and/or operating systems, many independent tools exist that provide logging on different platforms and/or operating systems. Examples of these tools include Log4j, Logback, SmartInspect, NLog, log4net, Microsoft Enterprise Library, ObjectGuy Framework, and/or others. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the operating system on which an application runs. As in tracing or profiling an application, one of the techniques that can be utilized includes adding instrumentation code to the operating system's source code. Such instrumentation code can be added to the operating system's source code before kernel compilation or recompilation, for instance. This type of instrumentation may involve defining or finding locations in the operating system's source code where instrumentation code may be inserted. Kernel instrumentation can also be performed without the need for kernel recompilation or rebooting. In some aspects, instrumentation code can be added at locations of interest through binary rewriting of compiled kernel code. In other aspects, kernel instrumentation can be performed dynamically where instrumentation code is added and/or removed where needed at runtime. For instance, dynamic instrumentation may overwrite kernel code with a branch instruction that redirects execution to instrumentation code or instrumentation routine. In yet other aspects, kernel instrumentation can be performed using just-in-time (JIT) dynamic instrumentation where execution may be redirected to a copy of kernel's code segment that includes instrumentation code. This type of instrumentation may include a JIT compiler and creation of a copy of the original code segment having instrumentation code or calls to instrumentation routines embedded into the original code segment. Instrumentation of the operating system may enable total system visibility including visibility into an application's behavior by enabling generation of low level trace information. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through tracing or profiling the processor on which an application runs. For example, some Intel processors provide Intel Processor Trace (i.e. Intel PT, etc.), a low-level tracing feature that enables recording executed instruction sets, and/or other data or information of one or more applications. Intel PT is facilitated by the Processor Trace Decoder Library along with its related tools. Intel PT is a low-overhead execution tracing feature that records information about application execution on each hardware thread using dedicated hardware facilities. The recorded execution/trace information is collected in data packets that can be buffered internally before being sent to a memory subsystem or another system or element (i.e. Acquisition Interface 120, Artificial Intelligence Unit 110, etc.) Intel PT also enables navigating the recorded execution/trace information via reverse stepping commands. Intel PT can be included in an operating system's core files and provided as a feature of the operating system. Intel PT can trace globally some or all applications running on an operating system. Acquisition Interface 120 or Artificial Intelligence Unit 110 can read or obtain the recorded execution/trace information from Intel PT. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through branch tracing or profiling. Branch tracing may include an abbreviated instruction trace in which only the successful branch instruction sets are traced or recorded. Branch tracing can be implemented through utilizing dedicated processor commands, for example. Executed branches may be saved into special branch trace store area of

Autonomous Ex. 2004 p. 384 of 510
Tesla v. Autonomous
IPR2024-00745

memory. With the availability and reference to a compiler listing of the application together with branch trace information, a full path of executed instruction sets can be reconstructed. The full path can also be reconstructed with a memory dump (containing the program storage) and branch trace information. In some aspects, branch tracing can be utilized for pre-learning or automated learning of an application's instruction sets, data, and/or other information where a number of application simulations (i.e. simulations of likely/common operations, etc.) are performed. As such, the application's operation can be learned automatically saving the time that would be needed to learn the application's operation directed by a user. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, obtaining instruction sets, data, and/or other information can be implemented through assembly language. Assembly language is a low-level programming language for a computer or other programmable device in which there is a strong correlation between the language and the architecture's machine instruction sets. Syntax, addressing modes, operands, and/or other elements of an assembly language instruction set may translate directly into numeric (i.e. binary, etc.) representations of that particular instruction set. Because of this direct relationship with the architecture's machine instruction sets, assembly language can be a powerful tool for tracing or profiling an application's execution in processor registers, memory, and/or other computing system components. For example, using assembly language, memory locations of a loaded application can be accessed, instrumented, and/or otherwise manipulated. In some aspects, assembly language can be used to rewrite or overwrite original in-memory instruction sets of an application with instrumentation instruction sets. In other aspects, assembly language can be used to redirect application's execution to instrumentation routine/subroutine or other code segment elsewhere in memory by inserting a jump into the application's in-memory code, by redirecting program counter, or by other techniques. Some operating systems may implement protection from changes to applications loaded into memory. Operating system, processor, or other low level commands such as Linux mprotect command or similar commands in other operating systems may be used to unprotect the protected locations in memory before the change. In yet other aspects, assembly language can be used to obtain instruction sets, data, and/or other information through accessing and/or reading instruction register, program counter, other processor registers, memory locations, and/or other components of a computing system. In yet other aspects, high-level programming languages may call or execute an external assembly language program to facilitate obtaining instruction sets, data, and/or other information as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, it may be sufficient to obtain user or other inputs, variables, parameters, and/or other data in some procedural, simple object oriented, or other applications. In one instance, a simple procedural application executes a sequence of instruction sets until the end of the program. During its execution, the application may receive user or other input, store the input in a variable, and perform calculations using the variable to reach a result. The value of the variable can be obtained or traced. In another instance, a more complex procedural application comprises one or more functions/routines/subroutines each of which may include a sequence of instruction sets. The application may execute a main sequence of instruction sets with a branch to a function/routine/subroutine. During its execution, the application may receive user or other input, store the input in a

Autonomous Ex. 2004 p. 385 of 510
Tesla v. Autonomous
IPR2024-00745

variable, and pass the variable as a parameter to the function/routine/subroutine. The function/routine/subroutine may perform calculations using the parameter and return a value that the rest of the application can use to reach a result. The value of the variable or parameter passed to the function/routine/subroutine, and/or return value can be obtained or traced. Values of user or other inputs, variables, parameters, and/or other items of interest can be obtained through previously described tracing, instrumentation, and/or other techniques. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 5, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of instruction sets or data in processor registers, memory, or other computing system components where instruction sets, data, and/or other information may be stored or utilized. For example, Instruction Register 212 may be part of Processor 11 and it may store the instruction set currently being executed or decoded. In some processors, Program Counter 211 (also referred to as instruction pointer, instruction address register, instruction counter, or part of instruction sequencer) may be incremented after fetching an instruction set, and it may hold or point to the memory address of the next instruction set to be executed. In a processor where the incrementation precedes the fetch, Program Counter 211 may point to the current instruction set being executed. In the instruction cycle, an instruction set may be loaded into Instruction Register 212 after Processor 11 fetches it from location in Memory 12 pointed to by Program Counter 211. Instruction Register 212 may hold the instruction set while it is decoded by Instruction Decoder 213, prepared, and executed. In some aspects, data (i.e. operands, etc.) needed for instruction set execution may be loaded from Memory 12 into a register within Register Array 214. In other aspects, the data may be loaded directly into Arithmetic Logic Unit 215. For instance, as instruction sets pass through Instruction Register 212 during application execution, they may be transmitted to Acquisition Interface 120 as shown. Examples of the steps in execution of a machine instruction set may include decoding the opcode (i.e. portion of a machine instruction set that may specify the operation to be performed), determining where the operands may be located (depending on architecture, operands may be in registers, the stack, memory, I/O ports, etc.), retrieving the operands, allocating processor resources to execute the instruction set (needed in some types of processors), performing the operation indicated by the instruction set, saving the results of execution, and/or other execution steps. Examples of the types of machine instruction sets that can be utilized include arithmetic, data handling, logical, program control, as well as special and/or other instruction set types. In addition to the ones described or shown, examples of other computing system or processor components that can be used during an instruction cycle include memory address register (MAR) that may hold the address of a memory block to be read from or written to; memory data register (MDR) that may hold data fetched from memory or data waiting to be stored in memory; data registers that may hold numeric values, characters, small bit arrays, or other data; address registers that may hold addresses used by instruction sets that indirectly access memory; general purpose registers (GPRs) that may store both data and addresses; conditional registers that may hold truth values often used to determine whether some instruction set should or should not be executed; floating point registers (FPRs) that may store floating point numbers; constant registers that may hold read-only values such as zero, one, or pi; special purpose registers (SPRs) such as status register, program counter, or stack pointer that may hold information on program state; machine-specific registers that may store data and settings related to a particular processor; Register Array 214 that may include an array of any number of processor registers; Arithmetic Logic Unit 215 that may perform arithmetic and logic operations; control unit that may direct processor's operation;

Autonomous Ex. 2004 p. 386 of 510
Tesla v. Autonomous
IPR2024-00745

and/or other circuits or components. Tracing, profiling, or sampling of processor registers, memory, or other computing system components can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of processor registers or any computing system components with marginal or no impact to computing overhead.

One of ordinary skill in art will recognize that Fig. 5 depicts one of many implementations of processor or computing system components, and that various additional components can be included, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate implementations. Processor or computing system components may be arranged or connected differently in alternate implementations. Processor or computing system components may also be connected with external elements using various connections. For instance, the connection between Instruction Register 212 and Acquisition Interface 120 may include any number or types of connections such as, for example, a dedicated connection for each bit of Instruction Register 212 (i.e. 32 connections for a 32 bit Instruction Register 212, etc.). Any of the described or other connections or interfaces may be implemented among any processor or computing system components and Acquisition Interface 120 or other elements.

Referring to Figs. 6A-6B, in yet another example, obtaining instruction sets, data, and/or other information may be implemented through tracing, profiling, or sampling of Logic Circuit 250. While Processor 11 includes any type or embodiment of logic circuit, Logic Circuit 250 is described separately here to offer additional detail on its functioning. Some Devices 98 may not need the processing capabilities of an entire Processor 11, but instead a more tailored Logic Circuit 250. Examples of such Devices 98 include home appliances, audio or video electronics, vehicle systems, toys, industrial machines, robots, and/or others. Logic Circuit 250 comprises the functionality for performing logic operations. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed on the inputs. Logic Circuit 250 may generally be implemented using transistors, diodes, and/or other electronic switches, but can also be constructed using vacuum tubes, electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecules, or even mechanical elements. In some aspects, Logic Circuit 250 may be or include a microcontroller, field-programmable gate array (FPGA), application-specific integrated circuit (ASIC), and/or other computing circuit or device. In other aspects, Logic Circuit 250 may be or include any circuit or device comprising one or more logic gates, one or more transistors, one or more switches, and/or one or more other logic components. In further aspects, Logic Circuit 250 may be or include any integrated or other circuit or device that can perform logic operations. Logic may generally refer to Boolean logic utilized in binary operations, but other logics can also be used. Input into Logic Circuit 250 may include or refer to a value inputted into the Logic Circuit 250, therefore, these terms may be used interchangeably herein depending on context. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the four input values are delivered to or received by Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the four hardwired connections as shown in Fig. 6A. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. As the two output values are generated by or transmitted out of Logic Circuit 250, they may be obtained by Acquisition Interface 120 through the two hardwired connections as shown in Fig. 6B. In a further example, instead of or in addition to obtaining input and/or output values of Logic Circuit 250, the state of Logic Circuit 250 may be obtained by reading or accessing values from one or more Logic Circuit's 250 internal

Autonomous Ex. 2004 p. 387 of 510
Tesla v. Autonomous
IPR2024-00745

components such as registers, memories, buses, and/or others (i.e. similar to the previously described tracing, profiling, and/or sampling of Processor 11 components, etc.). Tracing, profiling, or sampling of Logic Circuit 250 can be implemented in a program, combination of hardware and program, or purely hardware system. Dedicated hardware may be built to perform tracing, profiling, or sampling of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for tracing, profiling, or sampling of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements. In some designs, DCADO Unit 100 may include clamps and/or other elements to attach DCADO Unit 100 to inputs (i.e. input wires, etc.) into and/or outputs (i.e. output wires, etc.) from Logic Circuit 250. Such clamps and/or attachment elements enable seamless attachment of DCADO Unit 100 to any circuit or computing device without the need to redesign or alter the circuit or computing device.

In some embodiments, DCADO Unit 100 may learn input values directly from an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. As one or more input values or control signals are delivered to or received by the actuator, they may be obtained by Acquisition Interface 120 as previously described with respect to obtaining input values of Logic Circuit 250. Specifically, for instance, one or more input values or control signals of an actuator may be obtained by Acquisition Interface 120 via hardwired or other connections.

One of ordinary skill in art will recognize that Figs. 6A-6B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements may be utilized as needed for obtaining instruction sets, data, and/or other information, or some of the disclosed techniques or elements may be excluded, or a combination thereof may be utilized in alternate embodiments.

Referring to Figs. 7A-7E, some embodiments of Instruction Sets 526 are illustrated. In some aspects, Instruction Set 526 includes one or more instructions or commands of Application Program 18. For example, Instruction Set 526 may include one or more instructions or commands of a high-level programming language such as Java or SQL, a low-level language such as assembly or machine language, an intermediate language or construct such as bytecode, and/or any other language or construct. In other aspects, Instruction Set 526 includes one or more inputs into and/or outputs from Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In further aspects, Instruction Set 526 includes one or more values or states of registers and/or other components of Logic Circuit 250, Processor 11, and/or other processing element. In general, Instruction Set 526 may include one or more instructions, commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, characters, digits, references thereto, and/or other components for performing an operation.

In an embodiment shown in Fig. 7A, Instruction Set 526 includes code of a high-level programming language (i.e. Java, C++, etc.) comprising the following function call construct: Function1 (Parameter1, Parameter2,

Autonomous Ex. 2004 p. 388 of 510
Tesla v. Autonomous
IPR2024-00745

Parameter3, ...). An example of a function call applying the above construct includes the following Instruction Set 526: moveTo(Device1, 14, 8). The function or reference thereto "moveTo(Device1, 14, 8)" may be an Instruction Set 526 directing Device1 to move to a location with coordinates 14 and 8, for example. In another embodiment shown in Fig. 7B, Instruction Set 526 includes structured query language (SQL). In a further embodiment shown in Fig. 7C, Instruction Set 526 includes bytecode (i.e. Java bytecode, Python bytecode, CLR bytecode, etc.). In a further embodiment shown in Fig. 7D, Instruction Set 526 includes assembly code. In a further embodiment shown in Fig. 7E, Instruction Set 526 includes machine code.

Referring to Figs. 8A-8B, some embodiments of Extra Information 527 (also referred to as Extra Info 527) are illustrated. In an embodiment shown in Fig. 8A, Collection of Object Representations 525 may include or be associated with Extra Info 527. In an embodiment shown in Fig. 8B, Instruction Set 526 may include or be associated with Extra Info 527.

Extra Info 527 comprises the functionality for storing any information useful in comparisons or decision making performed in autonomous device operation, and/or other functionalities. One or more Extra Infos 527 can be stored in, appended to, or associated with a Collection of Object Representations 525, Instruction Set 526, and/or other element. In some embodiments, the system can obtain Extra Info 527 at a time of creating or generating Collection of Object Representations 525. In other embodiments, the system can obtain Extra Info 527 at a time of acquiring Instruction Set 526. In general, Extra Info 527 can be obtained at any time. Examples of Extra Info 527 include time information, location information, computed information, contextual information, and/or other information. Any information can be utilized that can provide information for enhanced comparisons or decision making performed in autonomous device operation. Which information is utilized and/or stored in Extra Info 527 can be set by a user, by DCADO system administrator, or automatically by the system. Extra Info 527 may include or be referred to as contextual information, and vice versa. Therefore, these terms may be used interchangeably herein depending on context.

In some aspects, time information (i.e. time stamp, etc.) can be utilized and/or stored in Extra Info 527. Time information can be useful in comparisons or decision making performed in autonomous device operation related to a specific time period as Device 98 may be required to perform specific operations at certain parts of day, month, year, and/or other time periods. Time information can be obtained from the system clock, online clock, oscillator, or other time source. In general, Extra Info 527 may include time information related to when Device 98 performed an operation. In other aspects, location information (i.e. coordinates, distance/angle from a known point, address, etc.) can be utilized and/or stored in Extra Info 527. Location information can be useful in comparisons or decision making performed in autonomous device operation related to a specific place as Device 98 may be required to perform specific operations at certain places. Location information can be obtained from a positioning system (i.e. radio signal triangulation, GPS capabilities, etc.), sensors, and/or other location system. In general, Extra Info 527 may include location information related to where Device 98 performed an operation. In further aspects, computed information can be utilized and/or stored in Extra Info 527. Computed information can be useful in comparisons or decision making performed in autonomous device operation where information can be calculated, inferred, or derived from other available information. DCADO Unit 100 and/or other disclosed elements may include computational functionalities to create Extra Info 527 by performing calculations or inferences using other information. In one example, Device's 98 speed can be computed or estimated from Device's 98 location and/or time

Autonomous Ex. 2004 p. 389 of 510
Tesla v. Autonomous
IPR2024-00745

information. In another example, Device's 98 bearing (i.e. angle or direction of movement, etc.) can be computed or estimated from Device's 98 location information by utilizing Pythagorean theorem, trigonometry, and/or other theorems, formulas, or disciplines. In a further example, speeds, bearings, distances, and/or other properties of objects around Device 98 can similarly be computed or inferred using known information. In further aspects, observed information can be utilized and/or stored in Extra Info 527. In further aspects, other information can be utilized and/or stored in Extra Info 527. Examples of such other information include user specific information (i.e. skill level, age, gender, etc.), group user information (i.e. access level, etc.), version of Application Program 18, type of Application Program 18, type of Processor 11, type of Logic Circuit 250, type of Device 98, and/or other information.

Referring to Fig. 9, an embodiment where DCADO Unit 100 is part of or operating on Processor 11 is illustrated. In one example, DCADO Unit 100 may be a hardware element or circuit embedded or built into Processor 11. In another example, DCADO Unit 100 may be a program operating on Processor 11.

Referring to Fig. 10, an embodiment where DCADO Unit 100 resides on Server 96 accessible over Network 95 is illustrated. Any number of Devices 98 may connect to such remote DCADO Unit 100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Devices 98 can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. A remote DCADO Unit 100 can be offered as a network service (i.e. online application, etc.). In some aspects, a remote DCADO Unit 100 (i.e. global DCADO Unit 100, etc.) may reside on the Internet and be available to all the world's Devices 98 configured to transmit their operations in circumstances including objects with various properties and/or configured to utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Devices 98 where the Devices 98 may be configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Device 98 in circumstances including objects with various properties. Server 96 may be or include any type or form of a remote computing device such as an application server, a network service server, a cloud server, a cloud, and/or other remote computing device. Server 96 may include any features, functionalities, and embodiments of the previously described Computing Device 70. It should be understood that Server 96 does not have to be a separate computing device and that Server 96, its elements, or its functionalities can be implemented on Device 98. Network 95 may include various networks, connection types, protocols, interfaces, APIs, and/or other elements or techniques known in art all of which are within the scope of this disclosure. Any of the previously described networks, network or connection types, networking interfaces, and/or other networking elements or techniques can similarly be utilized. Any of the disclosed elements may reside on Server 96 in alternate implementations. In one example, Artificial Intelligence Unit 110 can reside on Server 96 and Acquisition Interface 120 and/or Modification Interface 130 can reside on Device 98. In another example, Knowledgebase 530 can reside on Server 96 and the rest of the elements of DCADO Unit 100 can reside on Device 98. Any other combination of local and remote elements can be implemented.

Referring to Fig. 11, an embodiment of learning and/or using Remote Device's 97 circumstances for autonomous Remote Device 97 operation is illustrated. In such embodiments, in addition to providing input into Object Processing Unit 93 for learning functionalities herein, Sensor 92 (i.e. Camera 92a, Radar 92d, Sonar 92e, etc.) can provide input into Display 21 or other device for User's 50 perception of Remote Device's 97 surrounding.

Autonomous Ex. 2004 p. 390 of 510
Tesla v. Autonomous
IPR2024-00745

As User 50 operates Remote Device 97, DCADO Unit 100 may learn Remote Device's 97 operation in circumstances including objects with various properties. Such embodiments can be utilized in any situation where one device controls (i.e. remote controls, etc.) another device, any situation where some or all of the processing is on one device and sensor capabilities are on another device, and/or other situations. In one example, a drone controlling device (i.e. Device 98, etc.) may send control signals to operate a drone (i.e. Remote Device 97, etc.) and receive information on the drone's surrounding from Sensor 92 on the drone. In another example, a robot controlling device (i.e. Device 98, etc.) may send control signals to operate a robot (i.e. Remote Device 97, etc.) and receive information on the robot's surrounding from Sensor 92 on the robot. Any of the disclosed elements in addition to Sensor 92 may reside on Remote Device 97 in alternate implementations.

Referring to Fig. 12, an embodiment of Artificial Intelligence Unit 110 is illustrated. Artificial Intelligence Unit 110 comprises interconnected Knowledge Structuring Unit 520, Knowledgebase 530, Decision-making Unit 540, and Confirmation Unit 550. Other additional elements can be included as needed, or some of the disclosed ones can be excluded, or a combination thereof can be utilized in alternate embodiments.

Artificial Intelligence Unit 110 comprises the functionality for learning Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for learning one or more collections of object representations correlated with any instruction sets, data, and/or other information. In some aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Artificial Intelligence Unit 110 comprises the functionality for learning one or more Collections of Object Representations 525 some of which may not be correlated with any Instruction Sets 526 and/or Extra Info 527. Further, Artificial Intelligence Unit 110 comprises the functionality for anticipating Device's 98 operation in circumstances including objects with various properties. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more instruction sets, data, and/or other information. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 based on one or more incoming Collections of Object Representations 525. Artificial Intelligence Unit 110 comprises the functionality for anticipating one or more Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. Artificial Intelligence Unit 110 also comprises other disclosed functionalities.

Knowledge Structuring Unit 520, Knowledgebase 530, and Decision-making Unit 540 are described later.

Confirmation Unit 550 comprises the functionality for confirming, modifying, evaluating (i.e. rating, etc.), and/or canceling one or more anticipatory Instruction Sets 526, and/or other functionalities. Confirmation Unit 550 is an optional element that can be omitted depending on implementation. In some embodiments, Confirmation Unit 550 can serve as a means of confirming anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for confirmation. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of anticipatory Instruction Sets 526, etc.) to approve or confirm execution of the anticipatory Instruction Sets 526. In some aspects, Confirmation Unit 550 can automate User 50 confirmation. In one example, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be a perfect or highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be

Autonomous Ex. 2004 p. 391 of 510
Tesla v. Autonomous
IPR2024-00745

automatically executed without User's 50 confirmation. Conversely, if one or more incoming Collections of Object Representations 525 from Object Processing Unit 93 and one or more Collections of Object Representations 525 from a Knowledge Cell 800 were found to be less than a highly similar match, anticipatory Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800 can be presented to User 50 for confirmation and/or modifying. Any features, functionalities, and/or embodiments of Similarity Comparison 125 (later described) can be utilized for such similarity determination. In other embodiments, Confirmation Unit 550 can serve as a means of modifying or editing anticipatory Instruction Sets 526. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526 and provide them to User 50 for modification. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to modify the anticipatory Instruction Sets 526 before their execution. In further embodiments, Confirmation Unit 550 can serve as a means of evaluating or rating anticipatory Instruction Sets 526 if they matched User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. User 50 may be provided with an interface (i.e. graphical user interface, etc.) to rate (i.e. on a scale from 0 to 1, etc.) how well Decision-making Unit 540 predicted the executed anticipatory Instruction Sets 526. In some aspects, rating can be automatic and based on a particular function or method that rates how well the anticipatory Instruction Sets 526 matched the desired operation. In one example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were least modified in the confirmation process. In another example, a rating function or method can assign a higher rating to anticipatory Instruction Sets 526 that were canceled least number of times by User 50. Any other automatic rating function or method can be utilized. In yet other embodiments, Confirmation Unit 550 can serve as a means of canceling anticipatory Instruction Sets 526 if they did not match User's 50 intended operation. For example, Decision-making Unit 540 may determine one or more anticipatory Instruction Sets 526, which the system may automatically execute. The system may save the state of Device 98, Processor 11 (save its register values, etc.), Logic Circuit 250, Application Program 18 (i.e. save its variables, data structures, objects, location of its current instruction, etc.), and/or other processing elements before executing anticipatory Instruction Sets 526. User 50 may be provided with an interface (i.e. graphical user interface, selectable list of prior executed anticipatory Instruction Sets 526, etc.) to cancel one or more of the prior executed anticipatory Instruction Sets 526, and restore Device 98, Processor 11, Logic Circuit 250, Application Program 18, and/or other processing elements to a prior state. In some aspects, Confirmation Unit 550 can optionally be disabled or omitted in order to provide an uninterrupted operation of Device 98, Processor 11, Logic Circuit 250, and/or Application Program 18. For example, a thermostat may be suitable for implementing the user confirmation step, whereas, a vehicle may be less suitable for implementing such interrupting step due to the real time nature of vehicle operation.

Referring to Fig. 13, an embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. Knowledge Structuring Unit 520 comprises the functionality for structuring the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledge Structuring Unit 520 comprises the functionality for correlating one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 comprises the functionality for creating or generating Knowledge Cell 800 and storing one or more Collections of Object Representations 525 correlated with any

Autonomous Ex. 2004 p. 392 of 510
Tesla v. Autonomous
IPR2024-00745

Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. As such, Knowledge Cell 800 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledge Cell 800 includes knowledge (i.e. unit of knowledge, etc.) of how Device 98 operated in a circumstance including objects with various properties. Once created or generated, Knowledge Cells 800 can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, etc.) used for storing the knowledge of Device's 98 operation in circumstances including objects with various properties, thereby facilitating learning functionalities herein. It should be noted that Extra Info 527 may be optionally used in some implementations to enable enhanced comparisons or decision making in autonomous device operation where applicable, and that Extra Info 527 can be omitted in alternate implementations.

In some embodiments, Knowledge Structuring Unit 520 receives one or more Collections of Object Representations 525 from Object Processing Unit 93. Knowledge Structuring Unit 520 may also receive one or more Instruction Sets 526 from Acquisition Interface 120. Knowledge Structuring Unit 520 may further receive any Extra Info 527. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527. Knowledge Structuring Unit 520 may correlate one or more Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may then create Knowledge Cell 800 and store the one or more Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 into the Knowledge Cell 800. Knowledge Cell 800 may include any data structure or arrangement that can facilitate such storing. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it Collection of Object Representations 525a1 correlated with Instruction Sets 526a1-526a3 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a2 correlated with Instruction Set 526a4 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a3 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a4 correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a Collection of Object Representations 525a5 without a correlated Instruction Set 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may structure within Knowledge Cell 800ax additional Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above.

In some embodiments, Knowledge Structuring Unit 520 may correlate a Collection of Object Representations 525 with one or more temporally corresponding Instruction Sets 526 and/or Extra Info 527. This way, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation at or around the time of generating Collections of Object Representations 525. Such functionality enables spontaneous or seamless learning of Device's 98 operation in circumstances including objects with various properties as Device 98 is operated in real life situations. In some designs, Knowledge Structuring Unit 520 may receive a stream of Instruction Sets 526 used or executed to effect Device's 98 operations as well as a stream of Collections of Object Representations 525 as the

Autonomous Ex. 2004 p. 393 of 510
Tesla v. Autonomous
IPR2024-00745

operations are performed. Knowledge Structuring Unit 520 can then correlate Collections of Object Representations 525 from the stream of Collections of Object Representations 525 with temporally corresponding Instruction Sets 526 from the stream of Instruction Sets 526 and/or any Extra Info 527. Collections of Object Representations 525 without a temporally corresponding Instruction Set 526 may be uncorrelated, for instance. In some aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained at the time of generating the Collection of Object Representations 525. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within a certain time period before and/or after generating the Collection of Object Representations 525. For example, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained within 50 milliseconds, 1 second, 3 seconds, 20 seconds, 1 minute, 41 minutes, 2 hours, or any other time period before and/or after generating the Collection of Object Representations 525. Such time periods can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In other aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating the Collection of Object Representations 525 to the time of generating a next Collection of Object Representations 525. In further aspects, Instruction Sets 526 and/or Extra Info 527 that temporally correspond to a Collection of Object Representations 525 may include Instruction Sets 526 used and/or Extra Info 527 obtained from the time of generating a previous Collection of Object Representations 525 to the time of generating the Collection of Object Representations 525. Any other temporal relationship or correspondence between Collections of Object Representations 525 and correlated Instruction Sets 526 and/or Extra Info 527 can be implemented.

In some embodiments, Knowledge Structuring Unit 520 can structure the knowledge of Device's 98 operation in a circumstance including objects with various properties into any number of Knowledge Cells 800. In some aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In other aspects, Knowledge Structuring Unit 520 can structure into a Knowledge Cell 800 any number (i.e. 2, 4, 7, 17, 29, 87, 1415, 23891, 323674, 8132401, etc.) of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In a special case, Knowledge Structuring Unit 520 can structure all Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a single long Knowledge Cell 800. In further aspects, Knowledge Structuring Unit 520 can structure Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800. In a special case, Knowledge Structuring Unit 520 can store periodic streams of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 into a plurality of Knowledge Cells 800 such as hourly, daily, weekly, monthly, yearly, or other periodic Knowledge Cells 800.

In some embodiments, Device 98 may include a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Device 98. In another example, one or more Sensors 92 may be placed on different

Autonomous Ex. 2004 p. 394 of 510
Tesla v. Autonomous
IPR2024-00745

sub-devices, sub-systems, or elements of Device 98. Using multiple Sensors 92 and/or their corresponding Object Processing Units 93 may provide additional detail in learning and/or using Device's 98 circumstances for autonomous Device 98 operation. In some designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each Sensor 92 and its corresponding Object Processing Unit 93, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple Sensors 92 and/or their corresponding Object Processing Units 93 are utilized, collective Collections of Object Representations 525 from multiple Sensors 92 and their corresponding Object Processing Units 93 can be correlated with any Instruction Sets 526 and/or Extra Info 527.

In some embodiments, Device 98 may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. For example, each processing element may control a sub-device, sub-system, or an element of Device 98. Using multiple processing elements may provide enhanced control over Device's 98 operation. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, Collections of Object Representations 525 can be correlated with any Instruction Sets 526 and/or Extra Info 527 as previously described. In other designs where multiple processing elements are utilized, Collections of Object Representations 525 can be correlated with any collective Instruction Sets 526 and/or Extra Info 527 used or executed by a plurality of processing elements.

Any combination of the aforementioned multiple Sensors 92 and/or their corresponding Object Processing Units 93, multiple processing elements, and/or other elements can be implemented in alternate embodiments.

Referring to Fig. 14, another embodiment of Knowledge Structuring Unit 520 correlating individual Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single Collection of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Referring to Fig. 15, an embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. For example, Knowledge Structuring Unit 520 may create Knowledge Cell 800ax and structure within it a stream of Collections of Object Representations 525a1-525an correlated with Instruction Set 526a1 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525b1-525bn correlated with Instruction Sets 526a2-526a4 and/or and Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525c1-525cn without correlated Instruction Sets 526 and/or Extra Info 527. Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax a stream of Collections of Object Representations 525d1-525dn correlated with Instruction Sets 526a5-526a6 and/or any Extra Info 527 (not shown). Knowledge Structuring Unit 520 may further structure within Knowledge Cell 800ax additional streams of Collections of Object Representations 525 correlated with any number (including zero [i.e. uncorrelated]) of Instruction Sets 526 and/or Extra Info 527 by following similar logic as described above. The number of Collections of Object Representations 525 in some or all streams of Collections of Object Representations 525a1-525an, 525b1-525bn, etc. may be equal or different. It should be noted that n or other such letters or indicia may follow the sequence and/or context where

Autonomous Ex. 2004 p. 395 of 510
Tesla v. Autonomous
IPR2024-00745

they are indicated. Also, a same letter or indicia such as n may represent a different number in different elements of a drawing.

Referring to Fig. 16, another embodiment of Knowledge Structuring Unit 520 correlating streams of Collections of Object Representations 525 with any Instruction Sets 526 and/or Extra Info 527 is illustrated. In such embodiments, Knowledge Structuring Unit 520 may generate Knowledge Cells 800 each comprising a single stream of Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527.

Knowledgebase 530 comprises the functionality for storing the knowledge of a device's operation in circumstances including objects with various properties, and/or other functionalities. Knowledgebase 530 comprises the functionality for storing one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. Knowledgebase 530 comprises the functionality for storing one or more Knowledge Cells 800 each including one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527. In some aspects, Collections of Object Representations 525 correlated with Instruction Sets 526 and/or Extra Info 527 can be stored directly within Knowledgebase 530 without using Knowledge Cells 800 as the intermediary data structures. In some embodiments, Knowledgebase 530 may be or include Neural Network 530a (later described). In other embodiments, Knowledgebase 530 may be or include Graph 530b (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Sequences 530c (later described). In further embodiments, Knowledgebase 530 may be or include Sequence 533 (later described). In further embodiments, Knowledgebase 530 may be or include Collection of Knowledge Cells 530d (later described). In general, Knowledgebase 530 may be or include any data structure or arrangement capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase 530 may reside locally on Device 98, or remotely (i.e. remote Knowledgebase 530, etc.) on a remote computing device (i.e. server, cloud, etc.) accessible over a network or an interface.

In some embodiments, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. Therefore, the knowledge of Device's 98 operation in circumstances including objects with various properties learned on one Device 98 or DCADO Unit 100 can be transferred to one or more other Devices 98 or DCADO Units 100. In one example, Knowledgebase 530 can be copied or downloaded to a file or other repository from one Device 98 or DCADO Unit 100 and loaded or inserted into another Device 98 or DCADO Unit 100. In another example, Knowledgebase 530 from one Device 98 or DCADO Unit 100 can be available on a server accessible by other Devices 98 or DCADO Units 100 over a network or an interface. Once loaded into or accessed by a receiving Device 98 or DCADO Unit 100, the receiving Device 98 or DCADO Unit 100 can then implement the knowledge of Device's 98 operation in circumstances including objects with various properties learned on the originating Device 98 or DCADO Unit 100.

In some embodiments, multiple Knowledgebases 530 (i.e. Knowledgebases 530 from different Devices 98 or DCADO Units 100, etc.) can be combined to accumulate collective knowledge of operating Device 98 in circumstances including objects with various properties. In one example, one Knowledgebase 530 can be appended to another Knowledgebase 530 such as appending one Collection of Sequences 530c (later described) to another Collection of Sequences 530c, appending one Sequence 533 (later described) to another Sequence 533, appending one Collection of Knowledge Cells 530d (later described) to another Collection of Knowledge Cells 530d, and/or appending other data structures or elements thereof. In another example, one Knowledgebase 530 can be copied

Autonomous Ex. 2004 p. 396 of 510
Tesla v. Autonomous
IPR2024-00745

into another Knowledgebase 530 such as copying one Collection of Sequences 530c into another Collection of Sequences 530c, copying one Collection of Knowledge Cells 530d into another Collection of Knowledge Cells 530d, and/or copying other data structures or elements thereof. In a further example, in the case of Knowledgebase 530 being or including Graph 530b or graph-like data structure (i.e. Neural Network 530a, tree, etc.), a union can be utilized to combine two or more Graphs 530b or graph-like data structures. For instance, a union of two Graphs 530b or graph-like data structures may include a union of their vertex (i.e. node, etc.) sets and their edge (i.e. connection, etc.) sets. Any other operations or combination thereof on graphs or graph-like data structures can be utilized to combine Graphs 530b or graph-like data structures. In a further example, one Knowledgebase 530 can be combined with another Knowledgebase 530 through later described learning processes where Knowledge Cells 800 may be applied one at a time and connected with prior and/or subsequent Knowledge Cells 800 such as in Graph 530b or Neural Network 530a. In such embodiments, instead of Knowledge Cells 800 generated by Knowledge Structuring Unit 520, the learning process may utilize Knowledge Cells 800 from one Knowledgebase 530 to apply them onto another Knowledgebase 530. Any other techniques known in art including custom techniques for combining data structures can be utilized for combining Knowledgebases 530 in alternate implementations. In any of the aforementioned and/or other combining techniques, similarity of elements (i.e. nodes/vertices, edges/connections, etc.) can be utilized in determining whether an element from one Knowledgebase 530 matches an element from another Knowledgebase 530, and substantially or otherwise similar elements may be considered a match for combining purposes in some designs. Any features, functionalities, and embodiments of Similarity Comparison 125 (later described) can be used in such similarity determinations. A combined Knowledgebase 530 can be offered as a network service (i.e. online application, etc.), downloadable file, or other repository to all DCADO Units 100 configured to utilize the combined Knowledgebase 530. For example, a Device 98 including or interfaced with DCADO Unit 100 having access to a combined Knowledgebase 530 can use the collective knowledge learned from multiple Devices 98 for the Device's 98 autonomous operation.

Referring to Fig. 17, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include various artificial intelligence models and/or techniques. The disclosed systems, devices, and methods are independent of the artificial intelligence model and/or technique used and any model and/or technique can be utilized to facilitate the functionalities described herein. Examples of these models and/or techniques include deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other models and/or techniques.

In one example shown in Model A, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a neural network (also referred to as artificial neural network, etc.). As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a network of Nodes 852 (also referred to as neurons, etc.) and Connections 853 similar to that of a brain. Node 852 can store any data, object, data structure, and/or other item, or reference thereto. Node 852 may also include a function for transforming or manipulating any data, object, data structure, and/or other item. Examples of such transformation

Autonomous Ex. 2004 p. 397 of 510
Tesla v. Autonomous
IPR2024-00745

functions include mathematical functions (i.e. addition, subtraction, multiplication, division, sin, cos, log, derivative, integral, etc.), object manipulation functions (i.e. creating an object, modifying an object, deleting an object, appending objects, etc.), data structure manipulation functions (i.e. creating a data structure, modifying a data structure, deleting a data structure, creating a data field, modifying a data field, deleting a data field, etc.), and/or other transformation functions. Connection 853 may include or be associated with a value such as a symbolic label or numeric attribute (i.e. weight, cost, capacity, length, etc.). A computational model can be utilized to compute values from inputs based on a pre-programmed or learned function or method. For example, a neural network may include one or more input neurons that can be activated by inputs. Activations of these neurons can then be passed on, weighted, and transformed by a function to other neurons. Neural networks may range from those with only one layer of single direction logic to multi-layer of multi-directional feedback loops. A neural network can use weights to change the parameters of the network's throughput. A neural network can learn by input from its environment or from self-teaching using written-in rules. A neural network can be utilized as a predictive modeling approach in machine learning. An exemplary embodiment of a neural network (i.e. Neural Network 530a, etc.) is described later.

In another example shown in Model B, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a graph or graph-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 (also referred to as vertices or points, etc.) and Connections 853 (also referred to as edges, arrows, lines, arcs, etc.) organized as a graph. In general, any Node 852 in a graph can be connected to any other Node 852. A Connection 853 may include unordered pair of Nodes 852 in an undirected graph or ordered pair of Nodes 852 in a directed graph. Nodes 852 can be part of the graph structure or external entities represented by indices or references. A graph can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a graph may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, and vice versa. An exemplary embodiment of a graph (i.e. Graph 530b, etc.) is described later.

In a further example shown in Model C, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a tree or tree-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include Nodes 852 and Connections 853 (also referred to as references, edges, etc.) organized as a tree. In general, a Node 852 in a tree can be connected to any number (i.e. including zero, etc.) of children Nodes 852. A tree can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a tree may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network and/or graph, and vice versa.

In a further example shown in Model D, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a sequence or sequence-like data structure. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include a structure of Nodes 852 and/or Connections 853 organized as a sequence. In some aspects, Connections 853 may be optionally omitted from a

Autonomous Ex. 2004 p. 398 of 510
Tesla v. Autonomous
IPR2024-00745

sequence as the sequential order of Nodes 852 in a sequence may be implied in the structure. A sequence can be utilized as a predictive modeling approach in machine learning. Nodes 852, Connections 853, and/or other elements or operations of a sequence may include any features, functionalities, and embodiments of the aforementioned Nodes 852, Connections 853, and/or other elements or operations of a neural network, graph, and/or tree, and vice versa. An exemplary embodiment of a sequence (i.e. Collection of Sequences 530c, Sequence 533, etc.) is described later.

In yet another example, the disclosed artificially intelligent devices, systems, and methods for learning and/or using a device's circumstances for autonomous device operation may include a search-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities may include searching through a collection of possible solutions. For example, a search method can search through a neural network, graph, tree, sequence, or other data structure that includes data elements of interest. A search may use heuristics to limit the search for solutions by eliminating choices that are unlikely to lead to the goal. Heuristic techniques may provide a best guess solution. A search can also include optimization. For example, a search may begin with a guess and then refine the guess incrementally until no more refinements can be made. In a further example, the disclosed systems, devices, and methods may include logic-based model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can use formal or other type of logic. Logic based models may involve making inferences or deriving conclusions from a set of premises. As such, a logic based system can extend existing knowledge or create new knowledge automatically using inferences. Examples of the types of logic that can be utilized include propositional or sentential logic that comprises logic of statements which can be true or false; first-order logic that allows the use of quantifiers and predicates and that can express facts about objects, their properties, and their relations with each other; fuzzy logic that allows degrees of truth to be represented as a value between 0 and 1 rather than simply 0 (false) or 1 (true), which can be used for uncertain reasoning; subjective logic that comprises a type of probabilistic logic that may take uncertainty and belief into account, which can be suitable for modeling and analyzing situations involving uncertainty, incomplete knowledge and different world views; and/or other types of logic. In a further example, the disclosed systems, devices, and methods may include a probabilistic model and/or technique. As such, machine learning, knowledge structuring or representation, decision making, pattern recognition, and/or other artificial intelligence functionalities can be implemented to operate with incomplete or uncertain information where probabilities may affect outcomes. Bayesian network, among other models, is an example of a probabilistic tool used for purposes such as reasoning, learning, planning, perception, and/or others. One of ordinary skill in art will understand that the aforementioned artificial intelligence models and/or techniques are described merely as examples of a variety of possible implementations, and that while all possible artificial intelligence models and/or techniques are too voluminous to describe, other artificial intelligence models and/or techniques known in art are within the scope of this disclosure. One of ordinary skill in art will also recognize that an intelligent system may solve a specific problem by using any model and/or technique that works such as, for example, some systems can be symbolic and logical, some can be sub-symbolic neural networks, some can be deterministic or probabilistic, some can be hierarchical, some may include searching techniques, some may include optimization techniques, while

Autonomous Ex. 2004 p. 399 of 510
Tesla v. Autonomous
IPR2024-00745

others may use other or a combination of models and/or techniques. In general, any artificial intelligence model and/or technique can be utilized that can facilitate the functionalities described herein.

Referring to Figs. 18A-18C, embodiments of interconnected Knowledge Cells 800 and updating weights of Connections 853 are illustrated. As shown for example in Fig. 18A, Knowledge Cell 800za is connected to Knowledge Cell 800zb and Knowledge Cell 800zc by Connection 853z1 and Connection 853z2, respectively. Each of Connection 853z1 and Connection 853z2 may include or be associated with occurrence count, weight, and/or other parameter or data. The number of occurrences may track or store the number of observations that a Knowledge Cell 800 was followed by another Knowledge Cell 800 indicating a connection or relationship between them. For example, Knowledge Cell 800za was followed by Knowledge Cell 800zb 10 times as indicated by the number of occurrences of Connection 853z1. Also, Knowledge Cell 800za was followed by Knowledge Cell 800zc 15 times as indicated by the number of occurrences of Connection 853z2. The weight of Connection 853z1 can be calculated or determined as the number of occurrences of Connection 853z1 divided by the sum of occurrences of all connections (i.e. Connection 853z1 and Connection 853z2, etc.) originating from Knowledge Cell 800za. Therefore, the weight of Connection 853z1 can be calculated or determined as 10/(10+15)=0.4, for example. Also, the weight of Connection 853z2 can be calculated or determined as 15/(10+15)=0.6, for example. Therefore, the sum of weights of Connection 853z1, Connection 853z2, and/or any other Connections 853 originating from Knowledge Cell 800za may equal to 1 or 100%. As shown for example in Fig. 18B, in the case that Knowledge Cell 800zd is inserted and an observation is made that Knowledge Cell 800zd follows Knowledge Cell 800za, Connection 853z3 can be created between Knowledge Cell 800za and Knowledge Cell 800zd. The occurrence count of Connection 853z3 can be set to 1 and weight determined as 1/(10+15+1)=0.038. The weights of all other connections (i.e. Connection 853z1, Connection 853z2, etc.) originating from Knowledge Cell 800za may be updated to account for the creation of Connection 853z3. Therefore, the weight of Connection 853z1 can be updated as 10/(10+15+1)=0.385. The weight of Connection 853z2 can also be updated as 15/(10+15+1)=0.577. As shown for example in Fig. 18C, in the case that an additional occurrence of Connection 853z1 is observed (i.e. Knowledge Cell 800zb followed Knowledge Cell 800za, etc.), occurrence count of Connection 853z1 and weights of all connections (i.e. Connection 853z1, Connection 853z2, and Connection 853z3, etc.) originating from Knowledge Cell 800za may be updated to account for this observation. The occurrence count of Connection 853z1 can be increased by 1 and its weight updated as 11/(11+15+1)=0.407. The weight of Connection 853z2 can also be updated as 15/(11+15+1)=0.556. The weight of Connection 853z3 can also be updated as 1/(11+15+1)=0.037.

Referring to Fig. 19, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Knowledge Cells 530d is illustrated. Collection of Knowledge Cells 530d comprises the functionality for storing any number of Knowledge Cells 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Collection of Knowledge Cells 530d in a learning or training process. In effect, Collection of Knowledge Cells 530d may store Knowledge Cells 800 that can later be used to enable autonomous Device 98 operation. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 as previously described and the system applies them onto Collection of Knowledge Cells 530d, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The term apply or applying may refer to storing, copying, inserting, updating, or other similar action, therefore, these terms may be used

Autonomous Ex. 2004 p. 400 of 510
Tesla v. Autonomous
IPR2024-00745

interchangeably herein depending on context. The system can perform Similarity Comparisons 125 (later described) of a newly structured Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. If a substantially similar Knowledge Cell 800 is not found in Collection of Knowledge Cells 530d, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Collection of Knowledge Cells 530d, for example. On the other hand, if a substantially similar Knowledge Cell 800 is found in Collection of Knowledge Cells 530d, the system may optionally omit inserting the Knowledge Cell 800 from Knowledge Structuring Unit 520 as inserting a substantially similar Knowledge Cell 800 may not add much or any additional knowledge to the Collection of Knowledge Cells 530d, for example. Also, inserting a substantially similar Knowledge Cell 800 can optionally be omitted to save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. Any features, functionalities, and embodiments of Similarity Comparison 125, importance index (later described), similarity index (later described), and/or other disclosed elements can be utilized to facilitate determination of substantial or other similarity and whether to insert a newly structured Knowledge Cell 800 into Collection of Knowledge Cells 530d.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800ba and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bb into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is found between Knowledge Cell 800bc and any of the Knowledge Cells 800 in Collection of Knowledge Cells 530d, the system may perform no action. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800bd into the inserted new Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Collection of Knowledge Cells 530d. In the case that a substantially similar match is not found, the system may insert a new Knowledge Cell 800 into Collection of Knowledge Cells 530d and copy Knowledge Cell 800be into the inserted new Knowledge Cell 800. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Collection of Knowledge Cells 530d follows similar logic or process as the above-described.

Referring to Fig. 20, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a is illustrated. Neural Network 530a includes a number of neurons or Nodes 852 interconnected by Connections 853 as previously described. Knowledge Cells 800 are shown instead of Nodes 852 to simplify the illustration as Node 852 includes a Knowledge Cell 800, for example. Therefore, Knowledge Cells 800 and Nodes 852 can be used interchangeably herein depending on context. It should be noted that Node 852 may include other elements

Autonomous Ex. 2004 p. 401 of 510
Tesla v. Autonomous
IPR2024-00745

and/or functionalities instead of or in addition to Knowledge Cell 800. In some aspects, Knowledge Cells 800 may be stored into or applied onto Neural Network 530a individually or collectively in a learning or training process. In some designs, Neural Network 530a comprises a number of Layers 854 each of which may include one or more Knowledge Cells 800. Knowledge Cells 800 in successive Layers 854 can be connected by Connections 853. Connection 853 may include or be associated with occurrence count, weight, and/or other parameter or data as previously described. Neural Network 530a may include any number of Layers 854 comprising any number of Knowledge Cells 800. In some aspects, Neural Network 530a may store Knowledge Cells 800 interconnected by Connections 853 where following a path through the Neural Network 530a can later be used to enable autonomous Device 98 operation. It should be understood that, in some embodiments, Knowledge Cells 800 in one Layer 854 of Neural Network 530a need not be connected only with Knowledge Cells 800 in a successive Layer 854, but also in any other Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. A Knowledge Cell 800 can also be connected to itself such as, for example, in recurrent neural networks. In general, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 anywhere else in Neural Network 530a. In further embodiments, back-propagation of any data or information can be implemented. In one example, back-propagation of similarity (i.e. similarity index, etc.) of compared Knowledge Cells 800 in a path through Neural Network 530a can be implemented. In another example, back-propagation of errors can be implemented. Such back-propagations can then be used to adjust occurrence counts and/or weights of Connections 853 for better future predictions, for example. Any other back-propagation can be implemented for other purposes. Any combination of Nodes 852 (i.e. Nodes 852 comprising Knowledge Cells 800, etc.), Connections 853, Layers 854, and/or other elements or techniques can be implemented in alternate embodiments. Neural Network 530a may include any type or form of a neural network known in art such as a feed-forward neural network, a back-propagating neural network, a recurrent neural network, a convolutional neural network, deep neural network, and/or others including a custom neural network.

In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 (later described) of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in a Layer 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the Layer 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into the Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854. On the other hand, if a substantially similar Knowledge Cell 800 is found in the Layer 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a Knowledge Cell 800 in a prior Layer 854, and update any other Connections 853 originating from the Knowledge Cell 800 in the prior Layer 854.

For example, the system can perform Similarity Comparisons 125 (later described) of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854a of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800ba and Knowledge Cell 800ea, the

Autonomous Ex. 2004 p. 402 of 510
Tesla v. Autonomous
IPR2024-00745

system may perform no action since Knowledge Cell 800ea is the initial Knowledge Cell 800. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854b of Neural Network 530a. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800eb, the system may update occurrence count and weight of Connection 853e1 between Knowledge Cell 800ea and Knowledge Cell 800eb, and update weights of other Connections 853 originating from Knowledge Cell 800ea as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854c of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ec into Layer 854c and copy Knowledge Cell 800bc into the inserted Knowledge Cell 800ec. The system may also create Connection 853e2 between Knowledge Cell 800eb and Knowledge Cell 800ec with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other Connections 853 (one in this example) originating from Knowledge Cell 800eb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854d of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ed into Layer 854d and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800ed. The system may also create Connection 853e3 between Knowledge Cell 800ec and Knowledge Cell 800ed with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Layer 854e of Neural Network 530a. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ee into Layer 854e and copy Knowledge Cell 800be into the inserted Knowledge Cell 800ee. The system may also create Connection 853e4 between Knowledge Cell 800ed and Knowledge Cell 800ee with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Neural Network 530a follows similar logic or process as the above-described.

Referring now to Similarity Comparison 125, Similarity Comparison 125 comprises the functionality for comparing or matching Knowledge Cells 800 or portions thereof, and/or other functionalities. Similarity Comparison 125 comprises the functionality for comparing or matching Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching streams of Collections of Object Representations 525 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Representations 625 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Object Properties 630 or portions thereof. Similarity Comparison 125 comprises the functionality for comparing or matching Instruction Sets 526, Extra Info 527, text (i.e. characters, words, phrases, etc.), numbers, and/or other elements or portions thereof. Similarity Comparison 125 may include functions, rules, and/or logic for performing matching or comparisons and for determining that while a perfect match is not found, a partial or similar match has been found. In some aspects, a partial match may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. As such, Similarity Comparison 125 may include determining substantial similarity or substantial match of compared elements. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity.

Autonomous Ex. 2004 p. 403 of 510
Tesla v. Autonomous
IPR2024-00745

The rules for similarity or similar match can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In some designs, Similarity Comparison 125 comprises the functionality to automatically define appropriately strict rules for determining similarity of the compared elements. Similarity Comparison 125 can therefore set, reset, and/or adjust the strictness of the rules for finding or determining similarity of the compared elements, thereby fine tuning Similarity Comparison 125 so that the rules for determining similarity are appropriately strict. In some aspects, the rules for determining similarity may include a similarity threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their similarity exceeds a similarity threshold. In other aspects, the rules for determining similarity may include a difference threshold. As such, Similarity Comparison 125 can determine similarity of compared elements if their difference is lower than a difference threshold. In further aspects, the rules for determining similarity may include other thresholds. Similarity Comparison 125 enables comparing circumstances including objects with various properties and determining their similarity or match. In one example, a circumstance including an object detected at a distance of 8m and an angle/bearing of 64° relative to Device 98 may be found similar or matching by Similarity Comparison 125 to a circumstance including the same or similar object detected at a distance of 8.6m and an angle/bearing of 59° relative to Device 98. In another example, a circumstance including an object detected as a passenger vehicle may be found similar or matching by Similarity Comparison 125 to a circumstance including an object detected as a sport utility vehicle. In general, any one or more properties (i.e. existence, type, identity, distance, bearing/angle, location, shape/size, activity, etc.) of one or more objects can be utilized for determining similarity or match of circumstances including objects with various properties. Therefore, Similarity Comparison 125 provides flexibility in comparing and determining similarity of a variety of possible circumstances of Device 98.

In some embodiments where compared Knowledge Cells 800 include a single Collection of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform comparison of individual Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 with Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In some aspects, total equivalence is achieved when Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 matches Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800.

In some embodiments, in determining substantial similarity of individually compared Collections of Object Representations 525 (i.e. Collections of Object Representations 525 from the compared Knowledge Cells 800, etc.), Similarity Comparison 125 can compare one or more Object Representations 625 or portions (i.e. Object Properties 630, etc.) thereof from one Collection of Object Representations 525 with one or more Object Representations 625 or portions thereof from another Collection of Object Representations 525. In some aspects, total equivalence is found when all Object Representations 625 or portions thereof from one Collection of Object Representations 525 match all Object Representations 625 or portions thereof from another Collection of Object Representations 525. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Collections of Object Representations 525. In one example, substantial similarity can be

Autonomous Ex. 2004 p. 404 of 510
Tesla v. Autonomous
IPR2024-00745

achieved when most of the Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or a threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Representations 625 or portions thereof for determining substantial similarity of Collections of Object Representations 525. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Representations 625 or portions thereof such as Object Representations 625 representing near Objects 615, Object Representations 625 representing large Objects 615, etc., thereby tolerating mismatches in less important Object Representations 625 or portions thereof such as Object Representations 625 representing distant Objects 615, Object Representations 625 representing small Objects 615, etc. In general, any Object Representation 625 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Representations 625 or portions thereof from the comparison in determining substantial similarity of Collections of Object Representations 525. In one example, Object Representations 625 representing distant Objects 615 can be omitted from comparison. In another example, Object Representations 625 representing small Objects 615 can be omitted from comparison. In general, any Object Representation 625 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Collections of Object Representations 525. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Collections of Object Representations 525 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 81%, etc.) of Object Representations 625 or portions thereof from the compared Collections of Object Representations 525. If the comparison does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Representations 625 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Collections of Object Representations 525, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Representations 625 or

Autonomous Ex. 2004 p. 405 of 510
Tesla v. Autonomous
IPR2024-00745

portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Collections of Object Representations 525.

Where a reference to Object Representation 625 is used herein it should be understood that a portion of Object Representation 625 (i.e. Object Property 630, etc.) or a plurality of Object Representations 625 can be used instead of or in addition to the Object Representation 625. In one example, instead of or in addition to Object Representation 625, Object Properties 630 and/or other portions that constitute an Object Representation 625 can be compared. In another example, instead of or in addition to Object Representation 625, plurality of Object Representations 625 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Representation 625 may similarly apply to any portion of Object Representation 625 and/or a plurality of Object Representations 625 as applicable. In general, whole Object Representations 625, portions of Object Representations 625, and/or pluralities of Object Representations 625, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Representations 625 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments, in determining substantial similarity of Object Representations 625 (i.e. Object Representations 625 from the compared Collections of Object Representations 525, etc.), Similarity Comparison 125 can compare Object Properties 630 or portions (i.e. characters, words, numbers, etc.) thereof from one Object Representation 625 with Object Properties 630 or portions thereof from another Object Representation 625. In some aspects, total equivalence is found when all Object Properties 630 or portions thereof of one Object Representation 625 match all Object Properties 630 or portions thereof of another Object Representation 625. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Representations 625. In one example, substantial similarity can be achieved when most of the Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Object Properties 630 or portions thereof from the compared Object Representations 625 exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of Object Properties 630 or portions thereof from the compared Object Representations 625 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize Categories 635 associated with Object Properties 630 for determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof from the compared Object Representations 625 in a same Category 635 may be compared. This way, Object Properties 630 or portions thereof can be compared with their own peers. In one instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Type" may be compared. Any text comparison technique can be utilized in such comparing. In another

Autonomous Ex. 2004 p. 406 of 510
Tesla v. Autonomous
IPR2024-00745

instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Distance" or "Bearing" may be compared. Any number comparison technique can be utilized in such comparing. In a further instance, Object Properties 630 or portions thereof from the compared Object Representations 625 in Category 635 "Shape" may be compared. Any model, point cloud, or other computer construct comparison technique can be utilized in such comparing. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Object Properties 630 or portions thereof for determining substantial similarity of Object Representations 625. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Type", "Distance", "Bearing", etc., thereby tolerating mismatches in less important Object Properties 630 or portions thereof such as Object Properties 630 or portions thereof in Categories 635 "Identity", "Shape", etc. In general, any Object Property 630 or portion thereof can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can omit some of the Object Properties 630 or portions thereof from the comparison in determining substantial similarity of Object Representations 625. In one example, Object Properties 630 or portions thereof in Category 635 "Identity" can be omitted from comparison. In another example, Object Properties 630 or portions thereof in Category 635 "Shape" can be omitted from comparison. In general, any Object Property 630 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Object Representations 625. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Object Representations 625 had not been found. Similarity Comparison 125 can keep adjusting the strictness rules until a substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 87%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Object Properties 630 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Object Representations 625, Similarity Comparison 125 may further decrease the strictness (i.e. down to a certain minimum strictness or threshold, etc.) by requiring fewer Object Properties 630 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Object Representations 625. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Object Representations 625 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Object Representations 625 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 65%, etc.) of Object Properties 630 or portions thereof from the compared Object Representations 625. If the comparison determines a number of substantially similar Object Representations 625, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Object Representations 625. In response,

Autonomous Ex. 2004 p. 407 of 510
Tesla v. Autonomous
IPR2024-00745

Similarity Comparison 125 may attempt to find more matching or substantially matching Object Properties 630 or portions thereof in addition to the earlier found Object Properties 630 or portions thereof to limit the number of substantially similar Object Representations 625. If the comparison still provides more than one substantially similar Object Representation 625, Similarity Comparison 125 may further increase the strictness by requiring additional Object Properties 630 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Object Representations 625 until a best substantially similar Object Representation 625 is found.

Where a reference to Object Property 630 is used herein it should be understood that a portion of Object Property 630 or a plurality of Object Properties 630 can be used instead of or in addition to the Object Property 630. In one example, instead of or in addition to Object Property 630, characters, words, numbers, and/or other portions that constitute an Object Property 630 can be compared. In another example, instead of or in addition to Object Property 630, a plurality of Object Properties 630 can be compared. As such, any operations, rules, logic, and/or functions operating on Object Property 630 may similarly apply to any portion of Object Property 630 and/or a plurality of Object Properties 630 as applicable. In general, whole Object Properties 630, portions of Object Properties 630, and/or pluralities of Object Properties 630, including any operations thereon, can be combined to arrive at desired results. Some or all of the above-described rules, logic, and/or techniques can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Object Properties 630 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In some embodiments where compared Knowledge Cells 800 include a stream of Collections of Object Representations 525, in determining similarity of Knowledge Cells 800, Similarity Comparison 125 can perform collective comparison of Collections of Object Representations 525 or portions (i.e. Object Representations 625, Object Properties 630, etc.) thereof such as comparison of a stream of Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 with a stream of Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. Similarity Comparison 125 of collectively compared Collections of Object Representations 525 or portions thereof may include any features, functionalities, and embodiments of the previously described Similarity Comparison 125 of individually compared Collections of Object Representations 525 or portions thereof. In some aspects, total equivalence is found when all Collections of Object Representations 525 or portions thereof from one Knowledge Cell 800 match all Collections of Object Representations 525 or portions thereof from another Knowledge Cell 800. If total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial or other similarity of compared Knowledge Cells 800. In one example, substantial similarity can be achieved when most of the Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or a threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or

Autonomous Ex. 2004 p. 408 of 510
Tesla v. Autonomous
IPR2024-00745

percentage of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In some aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important Collections of Object Representations 525 or portions thereof such as more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.) or portions thereof, etc., thereby tolerating mismatches in less important Collections of Object Representations 525 or portions thereof such as less substantive or smaller Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a lower number of Object Representations 625, etc.) or portions thereof, etc. In general, any Collection of Object Representations 525 or portion thereof can be assigned higher or lower importance. In other aspects, Similarity Comparison 125 can utilize the order of Collections of Object Representations 525 or portions thereof for determining substantial similarity of Knowledge Cells 800. In one example, substantial similarity can be achieved when matches or substantial matches are found in earlier Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800, thereby tolerating mismatches in later Collections of Object Representations 525 or portions thereof. In another example, substantial similarity can be achieved when matches or substantial matches are found in corresponding (i.e. similarly ordered, temporally related, etc.) Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. In one instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a 94th Collection of Object Representations 525 or portions thereof from another Knowledge Cell 800. In another instance, a 94th Collection of Object Representations 525 or portions thereof from one Knowledge Cell 800 can be compared with a number of Collections of Object Representations 525 or portions thereof around (i.e. preceding and/or following) a 94th Collection of Object Representations 525 from another Knowledge Cell 800. This way, flexibility can be implemented in finding a substantially similar Collection of Object Representations 525 or portions thereof if the Collections of Object Representations 525 or portions thereof in the compared Knowledge Cells 800 are not perfectly aligned. In a further instance, Similarity Comparison 125 can utilize Dynamic Time Warping (DTW) and/or other techniques known in art for comparing and/or aligning temporal sequences (i.e. streams of Collections of Object Representations 525 or portions thereof, etc.) that may vary in time or speed. In further aspects, Similarity Comparison 125 can omit some of the Collections of Object Representations 525 or portions thereof from the comparison in determining substantial similarity of Knowledge Cells 800. In one example, less substantive or smaller Collections of Object Representations 525 or portions thereof can be omitted from comparison. In another example, some or all Collections of Object Representations 525 or portions thereof related to a specific time period can be omitted from comparison. In general, any Collection of Object Representations 525 or portion thereof can be omitted from comparison depending on implementation.

Similarity Comparison 125 can automatically adjust (i.e. increase or decrease) the strictness of the rules for determining substantial similarity of Knowledge Cells 800. In some aspects, such adjustment in strictness can be done by Similarity Comparison 125 in response to determining that total equivalence of compared Knowledge Cells

Autonomous Ex. 2004 p. 409 of 510
Tesla v. Autonomous
IPR2024-00745

800 had not been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until substantial similarity is found. All the rules or settings of substantial similarity can be set, reset, or adjusted by Similarity Comparison 125 in response to another strictness level determination. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 92%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may decide to decrease the strictness of the rules. In response, Similarity Comparison 125 may attempt to find fewer matching or substantially matching Collections of Object Representations 525 or portions thereof than in the previous attempt using stricter rules. If the comparison still does not determine substantial similarity of compared Knowledge Cells 800, Similarity Comparison 125 may further decrease (i.e. down to a certain minimum strictness or threshold, etc.) the strictness by requiring fewer Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further increasing a chance of finding substantial similarity in compared Knowledge Cells 800. In further aspects, an adjustment in strictness can be done by Similarity Comparison 125 in response to determining that multiple substantially similar Knowledge Cells 800 had been found. Similarity Comparison 125 can keep adjusting the strictness of the rules until a best of the substantially similar Knowledge Cells 800 is found. For example, Similarity Comparison 125 may attempt to find a match or substantial match in a certain percentage (i.e. 71%, etc.) of Collections of Object Representations 525 or portions thereof from the compared Knowledge Cells 800. If the comparison determines a number of substantially similar Knowledge Cells 800, Similarity Comparison 125 may decide to increase the strictness of the rules to decrease the number of substantially similar Knowledge Cells 800. In response, Similarity Comparison 125 may attempt to find more matching or substantially matching Collections of Object Representations 525 or portions thereof in addition to the earlier found Collections of Object Representations 525 or portions thereof to limit the number of substantially similar Knowledge Cells 800. If the comparison still provides more than one substantially similar Knowledge Cell 800, Similarity Comparison 125 may further increase the strictness by requiring additional Collections of Object Representations 525 or portions thereof to match or substantially match, thereby further narrowing the number of substantially similar Knowledge Cells 800 until a best substantially similar Knowledge Cell 800 is found.

Some or all of the aforementioned rules, logic, and/or techniques for determining substantial similarity of Knowledge Cells 800 can be utilized alone or in combination with each other or with other rules, logic, and/or techniques. One of ordinary skill in art will recognize that other techniques known in art for determining similarity of Knowledge Cells 800 and/or other data that would be too voluminous to describe are within the scope of this disclosure.

In any of the comparisons involving numbers such as, for example, Object Properties 630 including numbers (i.e. distances, bearings/angles, etc.), Similarity Comparison 125 can compare a number from one Object Property 630 with a number from another Object Property 630. In some aspects, total equivalence is found when the number from one Object Property 630 equals the number from another Object Property 630. In other aspects, if total equality is not found, Similarity Comparison 125 may attempt to determine substantial similarity of the compared numbers using a tolerance or threshold for determining a match. In some aspects, Similarity Comparison 125 can utilize a threshold for acceptable number difference in determining a match of compared numbers. For example, a threshold for acceptable number difference (i.e. absolute difference, etc.) can be set at 10. Therefore, 130 matches

Autonomous Ex. 2004 p. 410 of 510
Tesla v. Autonomous
IPR2024-00745

or is sufficiently similar to 135 because the number difference (i.e. 5 in this example) is lower than the threshold for acceptable number difference (i.e. 10 in this example, etc.). Furthermore, 130 does not match or is not sufficiently similar to 143 because the number difference (i.e. 13 in this example) is greater than the threshold for acceptable number difference. Any other threshold for acceptable number difference can be used such as 0.024, 1, 8, 15, 77, 197, 2438, 728322, and/or others. In other aspects, Similarity Comparison 125 can utilize a threshold for acceptable percentage difference in determining a match of compared numbers. For example, a threshold for acceptable percentage difference can be set at 10%. Therefore, 100 matches or is sufficiently similar to 106 because the percentage difference (i.e. 6% in this example) is lower than the threshold for acceptable percentage difference (i.e. 10% in this example). Furthermore, 100 does not match or is not sufficiently similar to 84 because the percentage difference (i.e. 16% in this example) is higher than the threshold for acceptable percentage difference. Any other threshold for acceptable percentage difference can be used such as 0.68%, 1%, 3%, 11%, 33%, 69%, 87%, and/or others. The aforementioned thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Other techniques known in art for comparing numbers can be utilized herein.

In any of the comparisons involving text such as, for example, Object Properties 630 including text (i.e. types, identities, etc.), Similarity Comparison 125 can compare words, characters, and/or other text from one Object Property 630 with words, characters, and/or other text from another Object Property 630. In some aspects, total equivalence is found when all words, characters, and/or other text from one Object Property 630 match all words, characters, and/or other text from another Object Property 630. In other aspects, if total equivalence is not found, Similarity Comparison 125 may attempt to determine substantial similarity of compared Object Properties 630. In one example, substantial similarity can be achieved when most of the words, characters, and/or other text from the compared Object Properties 630 match or substantially match. In another example, substantial similarity can be achieved when at least a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or percentage (i.e. 38%, 63%, 77%, 84%, 98%, etc.) of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Similarly, substantial similarity can be achieved when the number or percentage of matching or substantially matching words, characters, and/or other text from the compared Object Properties 630 exceeds a threshold number (i.e. 1, 2, 3, 4, 7, 11, etc.) or a threshold percentage (i.e. 48%, 63%, 77%, 84%, 98%, etc.). In a further example, substantial similarity can be achieved when all but a threshold number or percentage of words, characters, and/or other text from the compared Object Properties 630 match or substantially match. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In further aspects, Similarity Comparison 125 can utilize the importance (i.e. as indicated by importance index [later described], etc.) of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to more important words, characters, and/or other text such as longer words and/or other text, thereby tolerating mismatches in less important words, characters, and/or other text such as shorter words and/or other text. In general, any word, character, and/or other text can be assigned higher or lower importance. In further aspects, Similarity Comparison 125 can utilize the order of words, characters, and/or other text for determining substantial similarity of Object Properties 630. For example, substantial similarity can be achieved when matches or substantial matches are found with respect to front-most

Autonomous Ex. 2004 p. 411 of 510
Tesla v. Autonomous
IPR2024-00745

words, characters, and/or other text, thereby tolerating mismatches in later words, characters, and/or other text. In further aspects, Similarity Comparison 125 can utilize semantic conversion to account for variations of words and/or other text. In one example, Object Property 630 may include a word "house". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "home", "residence", "dwelling", "place", or other semantically similar variations of the word with a meaning "house". In another example, Object Property 630 may include a word "buy". In addition to searching for the exact word in a compared Object Property 630, Similarity Comparison 125 can employ semantic conversion and attempt to match "buying", "bought", or other semantically similar variations of the word with a meaning "buy" in different tenses. Any other grammatical analysis or transformation can be utilized to cover the full scope of word and/or other text variations. In some designs, semantic conversion can be implemented using a thesaurus or dictionary. In another example, semantic conversion can be implemented using a table where each row comprises semantically similar variations of a word and/or other text. In further aspects, Similarity Comparison 125 can utilize a language model for understanding or interpreting the concepts contained in the words and/or other text and compare the concepts instead of or in addition to the words and/or other text. Examples of language models include unigram model, n-gram model, neural network language model, bag of words model, and/or others. Any of the techniques for matching of words can similarly be used for matching of concepts. In further aspects, Similarity Comparison 125 can omit some of the words, characters, and/or other text from the comparison in determining substantial similarity of Object Properties 630. In one example, rear-most words, characters, and/or other text can be omitted from comparison. In another example, shorter words and/or other text can be omitted from comparison. In general, any word, character, and/or other text can be omitted from comparison depending on implementation. Other techniques known in art for comparing words, characters, and/or other text can be utilized herein.

In some embodiments, Similarity Comparison 125 can compare one or more Extra Info 527 (i.e. time information, location information, computed information, contextual information, and/or other information, etc.) in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. Extra Info 527 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, and/or other elements in the comparison. Since Extra Info 527 may include any contextual or other information that can be useful in determining similarity of any compared elements, Extra Info 527 can be used to enhance any of the aforementioned similarity determinations as applicable.

In some embodiments, Similarity Comparison 125 can also compare one or more Instruction Sets 526 in addition to or instead of comparing Collections of Object Representations 525 or portions thereof in determining substantial similarity of Knowledge Cells 800. In some aspects, Similarity Comparison 125 can compare portions of Instruction Sets 526 to determine substantial or other similarity of Instruction Sets 526. Similar to the above-described thresholds, thresholds for the number or percentage of matching portions of the compared Instruction Sets 526 can be utilized in determining substantial or other similarity of the compared Instruction Sets 526. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, and/or other techniques, knowledge, or input. In other aspects, Similarity Comparison 125 can compare text (i.e. characters, words, phrases, etc.), numbers, or other data (i.e. bits, etc.) to determine substantial or other similarity of Instruction Sets 526. Any other comparison technique can be

Autonomous Ex. 2004 p. 412 of 510
Tesla v. Autonomous
IPR2024-00745

utilized in comparing Instruction Sets 526 in alternate implementations. Instruction Sets 526 can be set to be less, equally, or more important (i.e. as indicated by importance index [later described], etc.) than Collections of Object Representations 525, Object Representations 625, Object Properties 630, Extra Info 527, and/or other elements in the comparison.

In some embodiments, an importance index (not shown) or other importance ranking technique can be used in any of the previously described comparisons or other processing involving elements of different importance. Importance index indicates importance of the element to or with which the index is assigned or associated. For example, importance index may indicate importance of a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element to or with which the index is assigned or associated. In some aspects, importance index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Importance index can be stored in or associated with the element to which the index pertains. Importance indexes of various elements can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. In one example, a higher Importance index can be assigned to more substantive or larger Collections of Object Representations 525 (i.e. Collections of Object Representations 525 comprising a higher number of Object Representations 625, etc.). In another example, a higher importance index can be assigned to Object Representations 625 representing closer, larger, and/or other Objects 615. Any importance index can be assigned to or associated with any element described herein depending on implementation. Any importance ranking technique can be utilized as or instead of importance index in alternate embodiments.

In some embodiments, Similarity Comparison 125 may generate a similarity index (not shown) for any of the compared elements. Similarity index indicates how well an element is matched with another element. For example, similarity index indicates how well a Knowledge Cell 800, Collection of Object Representations 525, Object Representation 625, Object Property 630, Instruction Set 526, Extra Info 527, and/or other element is matched with a compared element. In some aspects, similarity index on a scale from 0 to 1 can be utilized, although, any other range can also be utilized. Similarity index can be generated by Similarity Comparison 125 whether substantial or other similarity between the compared elements is achieved or not. In one example, similarity index can be determined for a Knowledge Cell 800 based on a ratio/percentage of matched or substantially matched Collections of Object Representations 525 relative to the number of Collections of Object Representations 525 in the compared Knowledge Cell 800. Specifically, similarity index of 0.91 is determined if 91% of Collections of Object Representations 525 of one Knowledge Cell 800 match or substantially match Collections of Object Representations 525 of another Knowledge Cell 800. In some designs, importance (i.e. as indicated by importance index, etc.) of one or more Collections of Object Representations 525 can be included in the calculation of a weighted similarity index. Similar determination of similarity index can be implemented with Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or other elements or portions thereof. Any combination of the aforementioned similarity index determinations or calculations can be utilized in alternate embodiments. Any similarity ranking technique can be utilized to determine or calculate similarity index in alternate embodiments.

Autonomous Ex. 2004 p. 413 of 510
Tesla v. Autonomous
IPR2024-00745

Referring to Fig. 21, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Neural Network 530a comprising shortcut Connections 853 is illustrated. In some designs, Knowledge Cells 800 in one Layer 854 of Neural Network 530a can be connected with Knowledge Cells 800 in any Layer 854, not only in a successive Layer 854, thereby creating shortcuts (i.e. shortcut Connections 853, etc.) through Neural Network 530a. In some aspects, creating a shortcut Connection 853 can be implemented by performing Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in any Layer 854 when applying (i.e. storing, copying, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 onto Neural Network 530a. Once created, shortcut Connections 853 enable a wider variety of Knowledge Cells 800 to be considered when selecting a path through Neural Network 530a. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Neural Network 530a, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in one or more Layers 854 of Neural Network 530a. If a substantially similar Knowledge Cell 800 is not found in the one or more Layers 854 of Neural Network 530a, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into a Layer 854 of Neural Network 530a, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in the one or more Layers 854 of Neural Network 530a, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, Layers 854, and/or other elements can similarly be utilized in Neural Network 530a that comprises shortcut Connections 853.

Referring to Fig. 22, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Graph 530b is illustrated. In some aspects, any Knowledge Cell 800 can be connected with any other Knowledge Cell 800 in Graph 530b. In other aspects, any Knowledge Cell 800 can be connected with itself and/or any other Knowledge Cell 800 in Graph 530b. In some embodiments, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies (i.e. store, copy, etc.) them onto Graph 530b, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform Similarity Comparisons 125 of a Knowledge Cell 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. If a substantially similar Knowledge Cell 800 is not found in Graph 530b, the system may insert (i.e. copy, store, etc.) the Knowledge Cell 800 from Knowledge Structuring Unit 520 into Graph 530b, and create a Connection 853 to the inserted Knowledge Cell 800 from a prior Knowledge Cell 800 including assigning an occurrence count to the new Connection 853, calculating a weight of the new Connection 853, and updating any other Connections 853 originating from the prior Knowledge Cell 800. On the other hand, if a substantially similar Knowledge Cell 800 is found in Graph 530b, the system may update occurrence count and weight of Connection 853 to that Knowledge Cell 800 from a prior Knowledge Cell 800, and update any other Connections 853 originating from the prior

Autonomous Ex. 2004 p. 414 of 510
Tesla v. Autonomous
IPR2024-00745

Knowledge Cell 800. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Graph 530b.

For example, the system can perform Similarity Comparisons 125 of Knowledge Cell 800ba from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800ha into Graph 530b and copy Knowledge Cell 800ba into the inserted Knowledge Cell 800ha. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bb from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bb and Knowledge Cell 800hb, the system may create Connection 853h1 between Knowledge Cell 800ha and Knowledge Cell 800hb with occurrence count of 1 and weight of 1. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bc from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is found between Knowledge Cell 800bc and Knowledge Cell 800hc, the system may update occurrence count and weight of Connection 853h2 between Knowledge Cell 800hb and Knowledge Cell 800hc, and update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hb as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800bd from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800hd into Graph 530b and copy Knowledge Cell 800bd into the inserted Knowledge Cell 800hd. The system may also create Connection 853h3 between Knowledge Cell 800hc and Knowledge Cell 800hd with occurrence count of 1 and weight calculated based on the occurrence count as previously described. The system may also update weights of other outgoing Connections 853 (one in this example) originating from Knowledge Cell 800hc as previously described. The system can then perform Similarity Comparisons 125 of Knowledge Cell 800be from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Graph 530b. In the case that a substantially similar match is not found, the system may insert Knowledge Cell 800he into Graph 530b and copy Knowledge Cell 800be into the inserted Knowledge Cell 800he. The system may also create Connection 853h4 between Knowledge Cell 800hd and Knowledge Cell 800he with occurrence count of 1 and weight of 1. Applying any additional Knowledge Cells 800 from Knowledge Structuring Unit 520 onto Graph 530b follows similar logic or process as the above-described.

Referring to Fig. 23, an embodiment of learning Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c comprises the functionality for storing one or more Sequences 533. Sequence 533 comprises the functionality for storing any number of Knowledge Cells 800. For example, Knowledge Structuring Unit 520 structures or generates Knowledge Cells 800 and the system applies them onto Collection of Sequences 530c, thereby implementing learning Device's 98 operation in circumstances including objects with various properties. The system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c to find a Sequence 533 comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. If Sequence 533 comprising such collectively substantially similar Knowledge Cells 800 is not found in Collection of Sequences 530c, the system

Autonomous Ex. 2004 p. 415 of 510
Tesla v. Autonomous
IPR2024-00745

may create a new Sequence 533 comprising the Knowledge Cells 800 from Knowledge Structuring Unit 520 and insert (i.e. copy, store, etc.) the new Sequence 533 into Collection of Sequences 530c. On the other hand, if Sequence 533 comprising collectively substantially similar Knowledge Cells 800 is found in Collection of Sequences 530c, the system may optionally omit inserting the Knowledge Cells 800 from Knowledge Structuring Unit 520 into Collection of Sequences 530c as inserting a similar Sequence 533 may not add much or any additional knowledge. This approach can save storage resources and limit the number of Knowledge Cells 800 that may later need to be processed or compared. In some aspects, a Sequence 533 may include Knowledge Cells 800 relating to a single operation of Device 98. In other aspects, a Sequence 533 may include Knowledge Cells 800 relating to a part of an operation of Device 98. In further aspects, one or more long Sequences 533 each including Knowledge Cells 800 of multiple operations of Device 98 can be utilized. In one example, Knowledge Cells 800 of all operations can be stored in a single long Sequence 533 in which case Collection of Sequences 530c as a separate element can be omitted. In another example, Knowledge Cells 800 of multiple operations can be included in a plurality of long Sequences 533 such as hourly, daily, weekly, monthly, yearly, or other periodic or other Sequences 533. Similarity Comparisons 125 can be performed by traversing the one or more long Sequences 533 to find a match or substantially similar match. For instance, the system can perform collective Similarity Comparisons 125 of Knowledge Cells 800 from Knowledge Structuring Unit 520 with Knowledge Cells 800 in subsequences of a long Sequence 533 in incremental or other traversing pattern to find a subsequence comprising Knowledge Cells 800 that are collectively substantially similar to the Knowledge Cells 800 from Knowledge Structuring Unit 520. The incremental traversing pattern may start from one end of a long Sequence 533 and move the comparison subsequence up or down one or any number of incremental Knowledge Cells 800 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Sequence 533 and subdividing the resulting sub-sequences in a recursive pattern, or any other traversing pattern or method. If a subsequence comprising collectively substantially similar Knowledge Cells 800 is not found in the long Sequence 533, the system may concatenate or append the Knowledge Cells 800 from Knowledge Structuring Unit 520 to the long Sequence 533. In further aspects, Connections 853 can optionally be used in Sequence 533 to connect Knowledge Cells 800. For example, a Knowledge Cell 800 can be connected not only with a next Knowledge Cell 800 in the Sequence 533, but also with any other Knowledge Cell 800 in the Sequence 533, thereby creating alternate routes or shortcuts through the Sequence 533. Any number of Connections 853 connecting any Knowledge Cells 800 can be utilized. Any of the previously described and/or other techniques for comparing, inserting, updating, and/or other operations on Knowledge Cells 800, Connections 853, and/or other elements can similarly be utilized in Sequences 533 and/or Collection of Sequences 530c.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned substantial similarity determinations with respect to collectively compared Knowledge Cells 800 and/or other elements. In some aspects, substantial similarity of collectively compared Knowledge Cells 800 can be determined based on similarities or similarity indexes of the individually compared Knowledge Cells 800. In one example, an average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. In another example, a weighted average of similarities or similarity indexes of individually compared Knowledge Cells 800 can be used to determine similarity of collectively compared Knowledge Cells 800. For instance, to affect the weighting of collective similarity, a higher weight or

Autonomous Ex. 2004 p. 416 of 510
Tesla v. Autonomous
IPR2024-00745

importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some Knowledge Cells 800 and lower for other Knowledge Cells 800. Any higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can similarly be utilized for collectively compared elements. In one example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when at least a threshold number or percentage of Knowledge Cells 800 from the collectively compared Knowledge Cells 800 match or substantially match. Similarly, substantial similarity of collectively compared Knowledge Cells 800 can be achieved when a number or percentage of matching or substantially matching Knowledge Cells 800 from the collectively compared Knowledge Cells 800 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Collections of Object Representations 525, Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

Any of the previously described data structures or arrangements of Knowledge Cells 800 such as Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or others can be used alone, or in combination with each other or with other elements, in alternate embodiments. In one example, a path in Neural Network 530a or Graph 530b may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. In another example, a part of a path in Neural Network 530a or Graph 530b may include a sequence of Knowledge Cells 800 interconnected with Knowledge Cells 800 in other paths, whereas, another part of the path may include its own separate sequence of Knowledge Cells 800 that are not interconnected with Knowledge Cells 800 in other paths. Any other combinations or arrangements of Knowledge Cells 800 can be implemented.

Referring to Fig. 24, an embodiment of determining anticipatory Instruction Sets 526 from a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. Decision-making Unit 540 comprises the functionality for anticipating or determining a device's operation in circumstances including objects with various properties. Decision-making Unit 540 comprises the functionality for anticipating or determining Instruction Sets 526 to be used or executed in Device's 98 autonomous operation. In some aspects, Instruction Sets 526 anticipated or determined to be used or executed in Device's 98 autonomous operation may be referred to as anticipatory Instruction Sets 526, alternate Instruction Sets 526, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Decision-making Unit 540 also comprises other disclosed functionalities.

In some aspects, Decision-making Unit 540 may anticipate or determine Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation by performing Similarity Comparisons 125 of incoming Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Knowledgebase 530 (i.e.

Autonomous Ex. 2004 p. 417 of 510
Tesla v. Autonomous
IPR2024-00745

Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). A Knowledge Cell 800 includes knowledge (i.e. one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in a circumstance including objects with various properties as previously described. When one or more Collections of Object Representations 525 representing objects with similar properties are received in the future, Decision-making Unit 540 can anticipate the Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) previously learned in a similar circumstance, thereby enabling autonomous Device 98 operation. In some aspects, Decision-making Unit 540 can perform Similarity Comparisons 125 of incoming Collections of Object Representations 525 from Object Processing Unit 93 with Collections of Object Representations 525 from Knowledge Cells 800 in Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). If one or more substantially similar Collections of Object Representations 525 or portions thereof are found in a Knowledge Cell 800 from Knowledgebase 530, Instruction Sets 526 (i.e. anticipatory Instruction Sets 526, etc.) for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with the one or more Collections of Object Representations 525 from the Knowledge Cell 800. In some designs, subsequent one or more Instruction Sets 526 for autonomous Device 98 operation can be anticipated in Instruction Sets 526 correlated with subsequent Collections of Object Representations 525 from the Knowledge Cell 800 or other Knowledge Cells 800, thereby anticipating not only current, but also additional future Instruction Sets 526. Although, Extra Info 527 is not shown in this and/or other figures for clarity of illustration, it should be noted that any Collection of Object Representations 525, Instruction Set 526, and/or other element may include or be associated with Extra Info 527 and that Decision-making Unit 540 can utilize Extra Info 527 for enhanced decision making.

For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525I1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a1-526a3 correlated with Collection of Object Representations 525a1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525I2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Set 526a4 correlated with Collection of Object Representations 525a2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525I3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525I4 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then

Autonomous Ex. 2004 p. 418 of 510
Tesla v. Autonomous
IPR2024-00745

perform Similarity Comparison 125 of Collection of Object Representations 525l5 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, the described comparisons in a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 25, an embodiment of determining anticipatory Instruction Sets 526 by traversing a single Knowledge Cell 800 is illustrated. Knowledge Cell 800 may be part of a Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) such as Collection of Knowledge Cells 530d. For example, Decision-making Unit 540 can perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a1 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a2 or portions thereof from Knowledge Cell 800oa may not be found substantially similar. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a3 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a3. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l2 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a4 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 can anticipate Instruction Sets 526a5-

Autonomous Ex. 2004 p. 419 of 510
Tesla v. Autonomous
IPR2024-00745

526a6 correlated with Collection of Object Representations 525a4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparison 125 of Collection of Object Representations 525l3 or portions thereof from Object Processing Unit 93 with Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa. Collection of Object Representations 525a5 or portions thereof from Knowledge Cell 800oa may be found substantially similar. Decision-making Unit 540 may not anticipate any Instruction Sets 526 since none are correlated with Collection of Object Representations 525a5. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. In one example, Extra Info 527 can be included in the Similarity Comparisons 125 as previously described. In another example, as history of incoming Collections of Object Representations 525 becomes available, Decision-making Unit 540 can perform collective Similarity Comparisons 125 of the history of Collections of Object Representations 525 or portions thereof from Object Processing Unit 93 with subsequences of Collections of Object Representations 525 or portions thereof from Knowledge Cell 800. In a further example, traversing may be performed in incremental traversing pattern such as starting from one end of Knowledge Cell 800 and moving the comparison subsequence up or down the list one or any number of incremental Collections of Object Representations 525 at a time. Other traversing patterns or methods can be employed such as starting from the middle of the Knowledge Cell 800 and subdividing the resulting subsequence in a recursive pattern, or any other traversing pattern or method. In a further example, the described traversing of a single Knowledge Cell 800 may be performed on any number of Knowledge Cells 800 sequentially or in parallel. Parallel processors such as a plurality of Processors 11 or cores thereof can be utilized for such parallel processing. In a further example, various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800 can be utilized as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 26, an embodiment of determining anticipatory Instruction Sets 526 using collective similarity comparisons is illustrated. For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collection of Object Representations 525l1 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collection of Object Representations 525c1 or portions thereof from Knowledge Cell 800rc may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c1, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l2 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525c1-525c2 or portions thereof from Knowledge Cell 800rc may be found

Autonomous Ex. 2004 p. 420 of 510
Tesla v. Autonomous
IPR2024-00745

substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525c2, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l3 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d3 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d3, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l4 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d4 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d4, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform collective Similarity Comparisons 125 of Collections of Object Representations 525l1-525l5 or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Collection of Knowledge Cells 530d. Collections of Object Representations 525d1-525d5 or portions thereof from Knowledge Cell 800rd may be found substantially similar with highest similarity. Decision-making Unit 540 can anticipate any Instruction Sets 526 (not shown) correlated with Collection of Object Representations 525d5, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

In some embodiments, various elements and/or techniques can be utilized in the aforementioned similarity determinations with respect to collectively compared Collections of Object Representations 525 and/or other elements. In some aspects, similarity of collectively compared Collections of Object Representations 525 can be determined based on similarities or similarity indexes of the individually compared Collections of Object Representations 525. In one example, an average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. In another example, a weighted average of similarities or similarity indexes of individually compared Collections of Object Representations 525 can be used to determine similarity of collectively compared Collections of Object Representations 525. For instance, to affect the weighting of collective similarity, a higher weight or importance (i.e. importance index, etc.) can be assigned to the similarities or similarity indexes of some (i.e. more substantive or larger, etc.) Collections of Object Representations 525 and lower for other (i.e. less substantive or smaller, etc.) Collections of Object Representations 525. Any other higher or lower weight or importance assignment can be implemented. In other aspects, any of the previously described or other thresholds for substantial similarity of individually compared elements can be similarly utilized for collectively compared elements. In one example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when their collective similarity or similarity index exceeds a similarity threshold. In another example, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when at

Autonomous Ex. 2004 p. 421 of 510
Tesla v. Autonomous
IPR2024-00745

least a threshold number or percentage of Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 match or substantially match. Similarly, substantial similarity of collectively compared Collections of Object Representations 525 can be achieved when a number or percentage of matching or substantially matching Collections of Object Representations 525 or portions thereof from the collectively compared Collections of Object Representations 525 exceeds a threshold. Such thresholds can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Similar elements and/or techniques as the aforementioned can be used for similarity determinations of other collectively compared elements such as Object Representations 625, Object Properties 630, Instruction Sets 526, Extra Info 527, Knowledge Cells 800, and/or others. Similarity determinations of collectively compared elements may include any features, functionalities, and embodiments of Similarity Comparison 125, and vice versa.

It should be understood that any of the described elements and/or techniques in the foregoing example can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 27, an embodiment of determining anticipatory Instruction Sets 526 using Neural Network 530a is illustrated. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or elements (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Decision-making Unit 540 can utilize various elements and/or techniques for selecting a path through Neural Network 530a. Although, these elements and/or techniques are described with respect to Neural Network 530a below, they can similarly be used in any Knowledgebase 530 (i.e. Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.) as applicable.

In some embodiments, Decision-making Unit 540 can utilize similarity index in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. For instance, similarity index may indicate how well one Knowledge Cell 800 or portions thereof are matched with another Knowledge Cell 800 or portions thereof as previously described. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 with highest similarity index even if Connection 853 pointing to that Knowledge Cell 800 has less than the highest weight. Therefore, similarity index or other such element or parameter can override or disregard the weight of a Connection 853 or other element. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity

Autonomous Ex. 2004 p. 422 of 510
Tesla v. Autonomous
IPR2024-00745

index is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. Similarity index can be set to be more, less, or equally important than a weight of a Connection 853.

In some embodiments, Decision-making Unit 540 can utilize Connections 853 in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In some aspects, Decision-making Unit 540 can take into account weights of Connections 853 among the interconnected Knowledge Cells 800 in choosing from which Knowledge Cell 800 to compare one or more Collections of Object Representations 525 first, second, third, and so on. Specifically, for instance, Decision-making Unit 540 can perform Similarity Comparisons 125 with one or more Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the highest weight Connection 853 first, Collections of Object Representations 525 from Knowledge Cell 800 pointed to by the second highest weight Connection 853 second, and so on. In other aspects, Decision-making Unit 540 can stop performing Similarity Comparisons 125 as soon as it finds one or more substantially similar Collections of Object Representations 525 in an interconnected Knowledge Cell 800. In further aspects, Decision-making Unit 540 may only follow the highest weight Connection 853 to arrive at a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 to be compared, thereby disregarding Connections 853 with less than the highest weight. In further aspects, Decision-making Unit 540 may ignore weights and/or other parameters of Connections 853. In further aspects, Decision-making Unit 540 may ignore Connections 853.

In some embodiments, Decision-making Unit 540 can utilize a bias to adjust similarity index, weight of a Connection 853, and/or other element or parameter used in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a. In one example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is higher than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In another example, Decision-making Unit 540 may select a Knowledge Cell 800 comprising one or more Collections of Object Representations 525 whose similarity index multiplied by or adjusted for a bias is lower than or equal to a weight of Connection 853 pointing to that Knowledge Cell 800. In a further example, bias can be used to resolve deadlock situations where similarity index is equal to a weight of a Connection 853. In some aspects, bias can be expressed in percentages such as 0.3 percent, 1.2 percent, 25.7 percent, 79.8 percent, 99.9 percent, 100.1 percent, 155.4 percent, 298.6 percent, 1105.5 percent, and so on. For example, a bias below 100 percent decreases an element or parameter to which it is applied, a bias equal to 100 percent does not change the element or parameter to which it is applied, and a bias higher than 100 percent increases the element or parameter to which it is applied. In general, any amount of bias can be utilized depending on implementation. Bias can be applied to one or more of a weight of a Connection 853, similarity index, any other element or parameter, and/or all or any combination of them. Also, different biases can be applied to each of a weight of a Connection 853, similarity index, or any other element or parameter. For example, 30 percent bias can be applied to similarity index and 15 percent bias can be applied to a weight of a Connection 853. Also, different biases can be applied to various Layers 854 of Neural Network 530a, and/or other

Autonomous Ex. 2004 p. 423 of 510
Tesla v. Autonomous
IPR2024-00745

disclosed elements. Bias can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input.

Any other element and/or technique can be utilized in selecting Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a.

In some embodiments, Neural Network 530a may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Neural Network 530a may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Neural Network 530a. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Neural Network 530a, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854a (or any other one or more Layers 854, etc.). Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ta may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854b interconnected with Knowledge Cell 800ta. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tb may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853t2 is the only connection from Knowledge Cell 800tb, Decision-making Unit 540 may follow Connection 853t2 and perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc in Layer 854c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800tc may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object

Autonomous Ex. 2004 p. 424 of 510
Tesla v. Autonomous
IPR2024-00745

Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854d interconnected with Knowledge Cell 800tc. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800td may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t3. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Layer 854e interconnected with Knowledge Cell 800td. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800te may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853t4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Neural Network 530a would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 for collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate instruction Sets 526 correlated with substantially similar streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Neural Network 530a such as in any Layer 854 subsequent to a current Layer 854, in the first Layer 854, in the entire Neural Network 530a, and/or others, even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object

Autonomous Ex. 2004 p. 425 of 510
Tesla v. Autonomous
IPR2024-00745

Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 28, an embodiment of determining anticipatory Instruction Sets 526 using Graph 530b is illustrated. Graph 530b may include knowledge (i.e. interconnected Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 using Graph 530b may include selecting a path of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof through Graph 530b. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof. Substantial similarity may be used primarily for selecting a path through Graph 530b, whereas, weight of any Connection 853 may be used secondarily or not at all.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ua may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ua by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ub may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u1 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ub by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800uc may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u2 disregarding its less than highest weight. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

Autonomous Ex. 2004 p. 426 of 510
Tesla v. Autonomous
IPR2024-00745

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Since Connection 853u3 is the only connection from Knowledge Cell 800uc, Decision-making Unit 540 may follow Connection 853u3 and perform Similarity Comparisons 125 of Collections of Object Representations 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud in Graph 530b. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ud may be found collectively substantially similar. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from one or more Knowledge Cells 800 in Graph 530b interconnected with Knowledge Cell 800ud by outgoing Connections 853. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ue may be found collectively substantially similar with highest similarity, thus, Decision-making Unit 540 may follow Connection 853u4. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, Connections 853, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof in a path through Graph 530b would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof of any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Graph 530b even if such Knowledge Cell 800 may be unconnected with a prior Knowledge Cell 800. It should be

Autonomous Ex. 2004 p. 427 of 510
Tesla v. Autonomous
IPR2024-00745

noted that any of Collections of Object Representations 525a1-525an, Collections of Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring to Fig. 29, an embodiment of determining anticipatory Instruction Sets 526 using Collection of Sequences 530c is illustrated. Collection of Sequences 530c may include knowledge (i.e. sequences of Knowledge Cells 800 comprising one or more Collections of Object Representations 525 correlated with any Instruction Sets 526 and/or Extra Info 527, etc.) of how Device 98 operated in circumstances including objects with various properties. In some aspects, determining anticipatory Instruction Sets 526 for autonomous Device 98 operation using Collection of Sequences 530c may include selecting a Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof from Collection of Sequences 530c. Individual and/or collective Similarity Comparisons 125 can be used to determine substantial similarity of the individually and/or collectively compared Collections of Object Representations 525 or portions thereof.

For example, Decision-making Unit 540 can perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in one or more Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cell 800ca in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an and 525b1-525bn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800ca-800cb in Sequence 533wc may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, and 525c1-525cn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dc in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-

Autonomous Ex. 2004 p. 428 of 510
Tesla v. Autonomous
IPR2024-00745

making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, and 525d1-525dn or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800dd in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can then perform Similarity Comparisons 125 of Collections of Object Representations 525a1-525an, 525b1-525bn, 525c1-525cn, 525d1-525dn, and 525e1-525en or portions thereof from Object Processing Unit 93 with Collections of Object Representations 525 or portions thereof from Knowledge Cells 800 in Sequences 533 of Collection of Sequences 530c. Collections of Object Representations 525 or portions thereof from Knowledge Cells 800da-800de in Sequence 533wd may be found collectively substantially similar with highest similarity. As the comparisons of individual Collections of Object Representations 525 are performed to determine collective similarity, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525 as previously described, thereby enabling autonomous Device 98 operation. Decision-making Unit 540 can implement similar logic or process for any additional Collections of Object Representations 525 from Object Processing Unit 93, and so on.

The foregoing exemplary embodiment provides an example of utilizing a combination of collective Similarity Comparisons 125, individual Similarity Comparisons 125, and/or other elements or techniques. It should be understood that any of these elements and/or techniques can be omitted, used in a different combination, or used in combination with other elements and/or techniques, in which case the selection of Sequence 533 of Knowledge Cells 800 or portions (i.e. Collections of Object Representations 525, Instruction Sets 526, etc.) thereof would be affected accordingly. Any of the elements and/or techniques utilized in other examples or embodiments described herein such as using Extra Info 527 in Similarity Comparisons 125, traversing of Knowledge Cells 800 or other elements, using history of Collections of Object Representations 525 or Knowledge Cells 800 in collective Similarity Comparisons 125, using various arrangements of Collections of Object Representations 525 and/or other elements in a Knowledge Cell 800, and/or others can similarly be utilized in this example. These elements and/or techniques can similarly be utilized in Neural Network 530a, Graph 530b, Collection of Knowledge Cells 530d, and/or other data structures or arrangements. In some aspects, instead of anticipating Instruction Sets 526 correlated with substantially similar individual Collections of Object Representations 525, Decision-making Unit 540 can anticipate Instruction Sets 526 correlated with substantially matching streams of Collections of Object Representations 525. In other aspects, any time that substantial similarity or other similarity threshold is not achieved in compared Collections of Object Representations 525 or portions thereof from any of the Knowledge Cells 800, Decision-making Unit 540 can decide to look for a substantially or otherwise similar Collections of Object Representations 525 or portions thereof in Knowledge Cells 800 elsewhere in Collection of Sequences 530c such as in different Sequences 533. It should be noted that any of Collections of Object Representations 525a1-525an, Collections of

Autonomous Ex. 2004 p. 429 of 510
Tesla v. Autonomous
IPR2024-00745

Object Representations 525b1-525bn, Collections of Object Representations 525c1-525cn, Collections of Object Representations 525d1-525dn, Collections of Object Representations 525e1-525en, etc. may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525. It should also be noted that any Knowledge Cell 800 may include one Collection of Object Representations 525 or a stream of Collections of Object Representations 525 as previously described. One of ordinary skill in art will understand that the foregoing exemplary embodiment is described merely as an example of a variety of possible implementations, and that while all of its variations are too voluminous to describe, they are within the scope of this disclosure.

Referring now to Modification Interface 130. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element at runtime. Modification Interface 130 comprises the functionality for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element based on anticipatory Instruction Sets 526. In one example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on runtime engine/environment, virtual machine, operating system, compiler, just-in-time (JIT) compiler, interpreter, translator, execution stack, file, object, data structure, and/or other computing system elements. In another example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on memory, storage, bus, interfaces, and/or other computing system elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on Processor 11 registers and/or other Processor 11 elements. In a further example, Modification Interface 130 comprises the functionality to access, modify, and/or perform other manipulations on inputs and/or outputs of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on functions, methods, procedures, routines, subroutines, and/or other elements of Application Program 18. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on source code, bytecode, compiled, interpreted, or otherwise translated code, machine code, and/or other code. In a further example, Modification Interface 130 comprises the functionality to access, create, delete, modify, and/or perform other manipulations on values, variables, parameters, and/or other data or information. Modification Interface 130 comprises any features, functionalities, and embodiments of Acquisition Interface 120, and vice versa. Modification Interface 130 also comprises other disclosed functionalities.

Modification Interface 130 can employ various techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. In some aspects, some of the previously described techniques and/or tools can be utilized. Code instrumentation, for instance, may involve inserting additional code, overwriting or rewriting existing code, and/or branching to a separate segment of code in Application Program 18 as previously described. For example, instrumented code may include the following:

```
Device1.moveLeft(23);
modifyApplication();
```

In the above sample code, instrumented call to Modification Interface's 130 function (i.e. modifyApplication(), etc.) can be placed after a function (i.e. Device1.moveLeft(23), etc.) of Application Program 18. Similar call to an

Autonomous Ex. 2004 p. 430 of 510
Tesla v. Autonomous
IPR2024-00745

application modifying function can be placed after or before some or all functions/routines/subroutines, some or all lines of code, some or all statements, some or all instructions or instruction sets, some or all basic blocks, and/or some or all other code segments of Application Program 18. One or more application modifying function calls can be placed anywhere in Application Program's 18 code and can be executed at any points in Application Program's 18 execution. The application modifying function (i.e. modifyApplication(), etc.) may include Artificial Intelligence Unit 110-determined anticipatory Instruction Sets 526 that can modify execution and/or functionality of Application Program 18. In some embodiments, the previously described obtaining Application Program's 18 instruction sets, data, and/or other information as well as modifying execution and/or functionality of Application Program 18 can be implemented in a single function that performs both tasks (i.e. traceAndModifyApplication(), etc.).

In some embodiments, various computing systems and/or platforms may provide native tools for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element. Independent vendors may provide tools with similar functionalities that can be utilized across different platforms. These tools enable a wide range of techniques or capabilities such as instrumentation, self-modifying code capabilities, dynamic code capabilities, branching, code rewriting, code overwriting, hot swapping, accessing and/or modifying objects or data structures, accessing and/or modifying functions/routines/subroutines, accessing and/or modifying variable or parameter values, accessing and/or modifying processor registers, accessing and/or modifying inputs and/or outputs, providing runtime memory access, and/or other capabilities. One of ordinary skill in art will understand that, while all possible variations of the techniques for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing element are too voluminous to describe, these techniques are within the scope of this disclosure.

In one example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing metaprogramming techniques, which include applications that can self-modify or that can create, modify, and/or manipulate other applications. Self-modifying code, dynamic code, reflection, and/or other techniques can be used to facilitate metaprogramming. In some aspects, metaprogramming is facilitated through a programming language's ability to access and manipulate the internals of the runtime engine directly or via an API. In other aspects, metaprogramming is facilitated through dynamic execution of expressions (i.e. anticipatory Instruction Sets 526, etc.) that can be created and/or executed at runtime. In yet other aspects, metaprogramming is facilitated through application modification tools, which can perform modifications on an application regardless of whether the application's programming language enables metaprogramming capabilities. Some operating systems may protect an application loaded into memory by restricting access to the loaded application. This protection mechanism can be circumvented by utilizing operating system's, processor's, and/or other low level features or commands to unprotect the loaded application. For example, a self-modifying application may modify the in-memory image of itself. To do so, the application can obtain the in-memory address of its code. The application may then change the operating system's or platform's protection on this memory range allowing it to modify the code (i.e. insert anticipatory Instruction Sets 526, etc.). In addition to a self-modifying application, one application can utilize similar technique to modify another application. Linux mprotect command or similar commends of other operating systems can be used to change protection (i.e. unprotect, etc.) for a region of memory, for example. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be

Autonomous Ex. 2004 p. 431 of 510
Tesla v. Autonomous
IPR2024-00745

implemented through native capabilities of dynamic, interpreted, and/or scripting programming languages and/or platforms. Most of these languages and/or platforms can perform functionalities at runtime that static programming languages may perform during compilation. Dynamic, interpreted, and/or scripting languages provide native functionalities such as self-modification of code, dynamic code, extending the application, adding new code, extending objects and definitions, and/or other functionalities that can modify an application's execution and/or functionality at runtime. Examples of dynamic, interpreted, and/or scripting languages include Lisp, Perl, PHP, JavaScript, Ruby, Python, Smalltalk, Tcl, VBScript, and/or others. Similar functionalities can also be provided in languages such as Java, C, and/or others using reflection. Reflection includes the ability of an application to examine and modify the structure and behavior of the application at runtime. For example, JavaScript can modify its own code as it runs by utilizing Function object constructor as follows:

```
myFunc=new Function(arg1, arg2, argN, functionBody);
```

The sample code above causes a new function object to be created with the specified arguments and body. The body and/or arguments of the new function object may include new instruction sets (i.e. anticipatory Instruction Sets 526, etc.). The new function can be invoked as any other function in the original code. In another example, JavaScript can utilize eval method that accepts a string of JavaScript statements (i.e. anticipatory Instruction Sets 526, etc.) and execute them as if they were within the original code. An example of how eval method can be used to modify an application includes the following JavaScript code:

```
anticipatoryInstr = 'Device1.moveForward(27);';
if (anticipatoryInstr != "" && anticipatoryInstr != null)
{
        eval(anticipatoryInstr);
}
```

In the sample code above, Artificial Intelligence Unit 110 may generate anticipatory Instruction Set 526 (i.e. 'Device1.moveForward(27)' for moving a Device1 forward 27 units, etc.) and save it in anticipatoryInstr variable, which eval method can then execute. Lisp is another example of dynamic, interpreted, and/or scripting language that includes similar capabilities as previously described JavaScript. For example, Lisp's compile command can create a function at runtime, eval command may parse and evaluate an expression at runtime, and exec command may execute a given instruction set (i.e. string, etc.) at runtime. In another example, dynamic as well as some non-dynamic languages may provide macros, which combine code introspection and/or eval capabilities. In some aspects, macros can access inner workings of the compiler, interpreter, virtual machine, runtime environment/engine, and/or other components of the computing platform enabling the definition of language-like constructs and/or generation of a complete program or sections thereof. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic code, dynamic class loading, reflection, and/or other native functionalities of a programming language or platform. In static applications or static programming, a class can be defined and/or loaded at compile time. Conversely, in dynamic applications or dynamic programming, a class can be loaded into a running environment at runtime. For example, Java Runtime Environment (JRE) may not require that all classes be loaded at compile time and class loading can occur when a class is first referenced at runtime. Dynamic class

Autonomous Ex. 2004 p. 432 of 510
Tesla v. Autonomous
IPR2024-00745

loading enables inclusion or injection of on-demand code and/or functionalities at runtime. System provided or custom class loaders may enable loading of classes into the running environment. Custom class loaders can be created to enable custom functionalities such as, for example, specifying a remote location from which a class can be loaded. In addition to dynamic loading of a pre-defined class, a class can also be created at runtime. In some aspects, a class source code can be created at runtime. A compiler such as javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages can then be utilized to compile the source code. Javac, com.sun.tools.javac.Main, javax.tools, javax.tools.JavaCompiler, and/or other packages may include an interface to invoke Java compiler from within a running application. A Java compiler may accept source code in a file, string, object (i.e. Java String, StringBuffer, CharSequence, etc.) and/or other source, and may generate Java bytecode (i.e. class file, etc.). Once compiled, a class loader can then load the compiled class into the running environment. In other aspects, a tool such as Javaassist (i.e. Java programming assistant) can be utilized to enable an application to create or modify a class at runtime. Javassist may include a Java library that provides functionalities to create and/or manipulate Java bytecode of an application as well as reflection capabilities. Javassist may provide source-level and bytecode-level APIs. Using the source-level API, a class can be created and/or modified using only source code, which Javassist may compile seamlessly on the fly. Javassist source-level API can therefore be used without knowledge of Java bytecode specification. Bytecode-level API enables creating and/or editing a class bytecode directly. In yet other aspects, similar functionalities to the aforementioned ones may be provided in tools such as Apache Commons BCEL (Byte Code Engineering Library), ObjectWeb ASM, CGLIB (Byte Code Generation Library), and/or others. Once a dynamic code or class is created and loaded, reflection in high-level programming languages such as Java and/or others can be used to manipulate or change the runtime behavior of an application. Examples of reflective programming languages and/or platforms include Java, JavaScript, Smalltalk, Lisp, Python, .NET Common Language Runtime (CLR), Tcl, Ruby, Perl, PHP, Scheme, PL/SQL, and/or others. Reflection can be used in an application to access, examine, modify, and/or manipulate a loaded class and/or its elements. Reflection in Java can be implemented by utilizing a reflection API such as java.lang.Reflect package. The reflection API provides functionalities such as, for example, loading or reloading a class, instantiating a new instance of a class, determining class and instance methods, invoking class and instance methods, accessing and manipulating a class, fields, methods and constructors, determining the modifiers for fields, methods, classes, and interfaces, and/or other functionalities. The above described dynamic code, dynamic class loading, reflection, and/or other functionalities are similarly provided in the .NET platform through its tools such as, for example, System.CodeDom.Compiler namespace, System.Reflection.Emit namespace, and/or other native or other .NET tools. Other platforms in addition to Java and .NET may provide similar tools and/or functionalities. In some designs, dynamic code, dynamic class loading, reflection, and/or other functionalities can be used to facilitate modification of an application by inserting or injecting instruction sets (i.e. anticipatory Instruction Sets 526, etc.) into a running application. For example, an existing or dynamically created class comprising DCADO Unit 100 functionalities can be loaded into a running application through manual, automatic, or dynamic instrumentation. Once the class is created and loaded, an instance of DCADO Unit 100 class may be constructed. The instance of DCADO Unit 100 can then take or exert control of the application and/or implement alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at any point in the application's execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Autonomous Ex. 2004 p. 433 of 510
Tesla v. Autonomous
IPR2024-00745

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through independent tools that can be utilized across different platforms. Such tools provide instrumentation and/or other capabilities on more than one platform or computing system and may facilitate application modification or insertion of instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Examples of these tools include Pin, DynamoRIO, DynInst, Kprobes, KernInst, OpenPAT, DTrace, SystemTap, and/or others. In some aspects, Pin and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. Pin can perform instrumentation by taking control of an application after it loads into memory. Pin may insert itself into the address space of an executing application enabling it to take control. Pin JIT compiler can then compile and implement alternate code (i.e. anticipatory Instruction Sets 526, etc.). Pin provides an extensive API for instrumentation at several abstraction levels. Pin supports two modes of instrumentation, JIT mode and probe mode. JIT mode uses a just-in-time compiler to insert instrumentation and recompile program code while probe mode uses code trampolines for instrumentation. Pin was designed for architecture and operating system independence. In other aspects, KernInst and/or any of its elements, methods, and/or techniques can be utilized for dynamic instrumentation. KernInst includes an instrumentation framework designed for dynamically inserting code into a running kernel of an operating system. KernInst implements probe-based dynamic instrumentation where code can be inserted, changed, and/or removed at will. Kerninst API enables client tools to construct their own tools for dynamic kernel instrumentation to suit variety of purposes such as insertion of alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Client tools can communicate with KernInst over a network (i.e. internet, wireless network, LAW, WAN, etc). Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing operating system's native tools or capabilities such as Unix ptrace command. Ptrace includes a system call that may enable one process to control another allowing the controller to inspect and manipulate the internal state of its target. Ptrace can be used to modify a running application such as modify an application with alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). By attaching to an application using the ptrace call, the controlling application can gain extensive control over the operation of its target. This may include manipulation of its instruction sets, execution path, file descriptors, memory, registers, and/or other components. Ptrace can single-step through the target's code, observe and intercept system calls and their results, manipulate the target's signal handlers, receive and send signals on the target's behalf, and/or perform other operations within the target application. Ptrace's ability to write into the target application's memory space enables the controller to modify the running code of the target application. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through utilizing just-in-time (JIT) compiling. JIT compilation (also known as dynamic translation, dynamic compilation, etc.) includes compilation performed during an application's execution (i.e. runtime, etc.). A code can be compiled when it is about to be executed, and it may be cached and reused later without the need for additional compilation. In some aspects, a JIT compiler can convert source code or byte code into machine code. In other aspects, a JIT compiler can convert source code into byte code. JIT compiling may be performed directly in memory. For example, JIT compiler can output machine code directly into memory and immediately execute it.

Autonomous Ex. 2004 p. 434 of 510
Tesla v. Autonomous
IPR2024-00745

Platforms such as Java, .NET, and/or others may implement JIT compilation as their native functionality. Platform independent tools for custom system design may include JIT compilation functionalities as well. In some aspects, JIT compilation includes redirecting application's execution to a JIT compiler from a specific entry point. For example, Pin can insert its JIT compiler into the address space of an application. Once execution is redirected to it, JIT compiler may receive alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) immediately before their compilation. The JIT compiled instruction sets can be stored in memory or another repository from where they may be retrieved and executed. Alternatively, for example, JIT compiler can create a copy of the original application code or a segment thereof, and insert alternate code (i.e. anticipatory Instruction Sets 526, etc.) before compiling the modified code copy. In some aspects, JIT compiler may include a specialized memory such as fast cache memory dedicated to JIT compiler functionalities from which the modified code can be fetched rapidly. JIT compilation and/or any compilation in general may include compilation, interpretation, or other translation into machine code, bytecode, and/or other formats or types of code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through dynamic recompilation. Dynamic recompilation includes recompiling an application or part thereof during execution. An application can be modified with alternate features or instruction sets that may take effect after recompilation. Dynamic recompilation may be practical in various types of applications including object oriented, event driven, forms based, and/or other applications. In a typical windows-based application, most of the action after initial startup occurs in response to user or system events such as moving the mouse, selecting a menu option, typing text, running a scheduled task, making a network connection, and/or other events when an event handler is called to perform an operation appropriate for the event. Generally, when no events are being generated, the application is idle. For example, when an event occurs and an appropriate event handler is called, instrumentation can be implemented in the application's source code to insert alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.) at which point the modified source code can be recompiled and/or executed. In some aspects, the state of the application can be saved before recompiling its modified source code so that the application may continue from its prior state. Saving the application's state can be achieved by saving its variables, data structures, objects, location of its current instruction, and/or other necessary information in environmental variables, memory, or other repositories where they can be accessed once the application is recompiled. In other aspects, application's variables, data structures, objects, address of its current instruction, and/or other necessary information can be saved in a repository such as file, database, or other repository accessible to the application after recompilation of its source code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through modifying or redirecting Application Program's 18 execution path. Generally, an application can be loaded into memory and the flow of execution proceeds from one instruction set to the next until the end of the application. An application may include a branching mechanism that can be driven by keyboard or other input devices, system events, and/or other computing system components or events that may impact the execution path. The execution path can also be altered by an external application through acquiring control of execution and/or redirecting execution to a function, routine/subroutine, or an alternate code segment at any point in the application's

Autonomous Ex. 2004 p. 435 of 510
Tesla v. Autonomous
IPR2024-00745

execution. A branch, jump, or other mechanism can be utilized to implement the redirected execution. For example, a jump instruction can be inserted at a specific point in an application's execution to redirect execution to an alternate code segment. A jump instruction set may include, for example, an unconditional branch, which always results in branching, or a conditional branch, which may or may not result in branching depending on a condition. When executing an application, a computer may fetch and execute instruction sets in sequence until it encounters a branch instruction set. If the instruction set is an unconditional branch, or it is conditional and the condition is satisfied, the computer may fetch its next instruction set from a different instruction set sequence or code segment as specified by the branch instruction set. After the execution of the alternate code segment, control may be redirected back to the original jump point or to another point in the application. For example, modifying an application can be implemented by redirecting execution of an application to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). Alternate instruction sets can be pre-compiled, pre-interpreted, or otherwise pre-translated and ready for execution. Alternate instruction sets can also be JIT compiled, JIT interpreted, or otherwise JIT translated before execution. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through assembly language. Assembly language instructions may be directly related with the architecture's machine instructions as previously described. Assembly language can, therefore, be a powerful tool for implementing direct hardware (i.e. processor registers, memory, etc.) access and manipulations as well as access and manipulations of specialized processor features or instructions. Assembly language can also be a powerful tool for implementing low-level embedded systems, real-time systems, interrupt handlers, self or dynamically modifying code, and/or other applications. Specifically, for instance, self or dynamically modifying code that can be used to facilitate modifying of an application can be seamlessly implemented using assembly language. For example, using assembly language, instruction sets can be dynamically created and loaded into memory similar to the ones that a compiler may generate. Furthermore, using assembly language, memory space of a loaded application can be accessed to modify (including rewrite, overwrite, etc.) original instruction sets or to insert jumps or branches to alternate code elsewhere in memory. Some operating systems may implement protection from changes to applications loaded into memory. Operating system's, processor's, or other low level features or commands can be used to unprotect the protected locations in memory before the change as previously described. Alternatively, a pointer that may reside in a memory location where it could be readily altered can be utilized where the pointer may reference alternate code. In one example, assembly language can be utilized to write alternate code (i.e. anticipatory Instruction Sets 526, etc.) into a location in memory outside a running application's memory space. Assembly language can then be utilized to redirect the application's execution to the alternate code by inserting a jump or branch into the application's in-memory code, by redirecting program counter, or by other technique. In another example, assembly language can be utilized to overwrite or rewrite the entire or part of an application's in-memory code with alternate code. In some aspects, high-level programming languages can call an external assembly language program to facilitate application modification as previously described. In yet other aspects, relatively low-level programming languages such as C may allow embedding assembly language directly in their source code such as, for example, using asm keyword of C. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Autonomous Ex. 2004 p. 436 of 510
Tesla v. Autonomous
IPR2024-00745

In a further example, modifying execution and/or functionality of Application Program 18 can be implemented through binary rewriting. Binary rewriting tools and/or techniques may modify an application's executable. In some aspects, modification can be minor such as in the case of optimization where the original executable's functionality is kept. In other aspects, modification may change the application's functionality such as by inserting alternate code (i.e. anticipatory Instruction Sets 526, etc.). Examples of binary rewriting tools include SecondWrite, ATOM, DynamoRIO, Purify, Pin, EEL, DynInst, PLTO, and/or others. Binary rewriting may include disassembly, analysis, and/or modification of target application. Since binary rewriting works directly on machine code executable, it is independent of source language, compiler, virtual machine (if one is utilized), and/or other higher level abstraction layers. Also, binary rewriting tools can perform application modifications without access to original source code. Binary rewriting tools include static rewriters, dynamic rewriters, minimally-invasive rewriters, and/or others. Static binary rewriters can modify an executable when the executable is not in use (i.e. not running). The rewritten executable may then be executed including any new or modified functionality. Dynamic binary rewriters can modify an executable during its execution, thereby enabling modification of an application's functionality at runtime. In some aspects, dynamic rewriters can be used for instrumentation or selective modifications such as insertion of alternate code (i.e. anticipatory Instruction Sets 526, etc.), and/or for other runtime transformations or modifications. For example, some dynamic rewriters can be configured to intercept an application's execution at indirect control transfers and insert instrumentation or other application modifying code. Minimally-invasive rewriters may keep the original machine code to the greatest extent possible. They support limited modifications such as insertion of jumps into and out of instrumented code. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Fig. 30, in a further example, modifying execution and/or functionality of Processor 11 can be implemented through modification of processor registers, memory, or other computing system components. In some aspects, modifying execution and/or functionality of Processor 11 can be implemented by redirecting Processor's 11 execution to alternate instruction sets (i.e. anticipatory Instruction Sets 526, etc.). In one example, Program Counter 211 may hold or point to a memory address of the next instruction set that will be executed by Processor 11. Artificial Intelligence Unit 110 may generate anticipatory Instruction Sets 526 and store them in Memory 12 as previously described. Modification Interface 130 may then change Program Counter 211 to point to the location in Memory 12 where anticipatory Instruction Sets 526 are stored. The anticipatory Instruction Sets 526 can then be fetched from the location in Memory 12 pointed to by the modified Program Counter 211 and loaded into Instruction Register 212 for decoding and execution. Once anticipatory Instruction Sets 526 are executed, Modification Interface 130 may change Program Counter 211 to point to the last instruction set before the redirection or to any other instruction set. In other aspects, anticipatory Instruction Sets 526 can be loaded directly into Instruction Register 212. As previously described, examples of other processor or computing system components that can be used during an instruction cycle include memory address register (MAR), memory data register (MDR), data registers, address registers, general purpose registers (GPRs), conditional registers, floating point registers (FPRs), constant registers, special purpose registers, machine-specific registers, Register Array 214, Arithmetic Logic Unit 215, control unit, and/or other circuits or components. Any of the aforementioned processor registers, memory, or other computing system components can be accessed and/or modified to facilitate the disclosed functionalities. In some embodiments, processor interrupt may be issued to facilitate such access and/or modification. In some

Autonomous Ex. 2004 p. 437 of 510
Tesla v. Autonomous
IPR2024-00745

designs, modifying execution and/or functionality of Processor 11 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Processor 11 with marginal or no impact to computing overhead. Other platforms, tools, and/or techniques may provide equivalent or similar functionalities as the above described ones.

Referring to Figs. 31A-31B, in a further example, modifying execution and/or functionality of Logic Circuit 250 can be implemented through modification of inputs and/or outputs of Logic Circuit 250. While Processor 11 includes any type of logic circuit, Logic Circuit 250 is described separately herein to offer additional detail on its functioning. Logic Circuit 250 comprises the functionality for performing logic operations using the circuit's inputs and producing outputs based on the logic operations performed as previously described. In one example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to Logic Circuit 250 through the four hardwired connections as shown in Fig. 31A. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to Logic Circuit 250 from its usual input source. As such, DCADO Unit 100 may cause Logic Circuit 250 to perform its logic operations using the four anticipatory input values, thereby implementing autonomous Device 98 operation. In another example, Logic Circuit 250 may perform some logic operations using four input values and produce two output values. Modifying execution and/or functionality of Logic Circuit 250 can be implemented by replacing its output values with anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.). Artificial Intelligence Unit 110 may generate anticipatory output values (i.e. anticipatory Instruction Sets 526, etc.) as previously described. Modification Interface 130 can then transmit the anticipatory output values through the two hardwired connections as shown in Fig. 31B. Modification Interface 130 may use Switches 251 to prevent delivery of any output values that may be sent by Logic Circuit 250. As such, DCADO Unit 100 may bypass Logic Circuit 250 and transmit the two anticipatory output values to downstream elements, thereby implementing autonomous Device 98 operation. In a further example, instead of or in addition to modifying input and/or output values of Logic Circuit 250, the execution and/or functionality of Logic Circuit 250 may be modified by modifying values or signals in one or more Logic Circuit's 250 internal components such as registers, memories, buses, and/or others (i.e. similar to the previously described modifying of Processor 11 components, etc.). In some designs, modifying execution and/or functionality of Logic Circuit 250 can be implemented in a program, combination of programs and hardware, or purely hardware system. Dedicated hardware may be built to perform modifying execution and/or functionality of Logic Circuit 250 with marginal or no impact to computing overhead. Any of the elements and/or techniques for modifying execution and/or functionality of Logic Circuit 250 can similarly be implemented with Processor 11 and/or other processing elements.

In some embodiments, DCADO Unit 100 may directly modify the functionality of an actuator (previously described, not shown). For example, Logic Circuit 250 or other processing element may control an actuator that enables Device 98 to perform mechanical, physical, and/or other operations. An actuator may receive one or more input values or control signals from Logic Circuit 250 or other processing element directing the actuator to perform specific operations. Modifying functionality of an actuator can be implemented by replacing its input values with anticipatory input values (i.e. anticipatory Instruction Sets 526, etc.) as previously described with respect to replacing

Autonomous Ex. 2004 p. 438 of 510
Tesla v. Autonomous
IPR2024-00745

input values of Logic Circuit 250. Specifically, for instance, Artificial Intelligence Unit 110 may generate anticipatory input values as previously described. Modification Interface 130 can then transmit the anticipatory input values to the actuator. Modification Interface 130 may use Switches 251 to prevent delivery of any input values that may be sent to the actuator from its usual input source. As such, DCADO Unit 100 may cause the actuator to perform its operations using the anticipatory input values, thereby implementing autonomous Device 98 operation.

One of ordinary skill in art will recognize that Figs. 31A-31B depict one of many implementations of Logic Circuit 250 and that any number of input and/or output values can be utilized in alternate implementations. One of ordinary skill in art will also recognize that Logic Circuit 250 may include any number and/or combination of logic components to implement any logic operations.

Other additional techniques or elements can be utilized as needed for modifying execution and/or functionality of Application Program 18, Processor 11, Logic Circuit 250, and/or other processing elements, or some of the disclosed techniques or elements can be excluded, or a combination thereof can be utilized in alternate embodiments.

Referring to Fig. 32, the illustration shows an embodiment of a method 9100 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9100 may include any action or operation of any of the disclosed methods such as method 9200, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9100.

At step 9105, a first collection of object representations is received. A collection of object representations (i.e. Collection of Object Representations 525, etc.) may include one or more object representations (i.e. Object Representations 625, etc.), object properties (i.e. Object Properties 630, etc.), and/or other elements or information. An object representation may include an electronic representation of an object (i.e. Object 615, etc.) detected in a device's surrounding. In some aspects, a collection of object representations may include one or more object representations, object properties, and/or other elements or information detected in a device's (i.e. Device's 98, etc.) surrounding at a particular time. A collection of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties at a particular time. In some designs, a collection of object representations may include or be associated with a time stamp (not shown), order (not shown), or other time related information. In some embodiments, a collection of object representations may include or be substituted with a stream of collections of object representations, and vice versa. Therefore, the terms collection of object representations and stream of collections of object representations may be used interchangeably herein depending on context. A stream of collections of object representations may include one collection of object representations or a group, sequence, or other plurality of collections of object representations. In some aspects, a stream of collections of object representations may include one or more collections of object representations, and/or other elements or information detected in a device's surrounding over time. A stream of collections of object representations may, therefore, include knowledge (i.e. unit of knowledge, etc.) of a device's circumstances including objects with various properties over time. As circumstances including objects with various properties in a device's surrounding change (i.e. objects and/or their properties change, move, act, transform, etc.) over time, this

Autonomous Ex. 2004 p. 439 of 510
Tesla v. Autonomous
IPR2024-00745

change may be captured in a stream of collections of object representations. In some designs, each collection of object representations in a stream may include or be associated with the aforementioned time stamp, order, or other time related information. Examples of objects include biological objects (i.e. persons, animals, vegetation, etc.), nature objects (i.e. rocks, bodies of water, etc.), manmade objects (i.e. buildings, streets, ground/aerial/aquatic vehicles, etc.), and/or others. In some aspects, any part of an object may be detected as an object itself. For instance, instead of or in addition to detecting a vehicle as an object, a wheel and/or other parts of the vehicle may be detected as objects. In general, an object may include any object or part thereof that can be detected. Examples of object properties include existence of an object, type of an object (i.e. person, cat, vehicle, building, street, tree, rock, etc.), identity of an object (i.e. name, identifier, etc.), distance of an object, bearing/angle of an object, location of an object (i.e. distance and bearing/angle from a known point, object coordinates, etc.), shape/size of an object (i.e. height, width, depth, computer model, point cloud, etc.), activity of an object (i.e. motion, gestures, etc.), and/or other properties of an object. In general, an object property may include any attribute of an object (i.e. existence of an object, type of an object, identity of an object, shape/size of an object, etc.), any relationship of an object with the device, other objects, or the environment (i.e. distance of an object, bearing/angle of an object, friend/foe relationship, etc.), and/or other information related to an object. Objects and/or their properties can be detected by one or more sensors (i.e. Sensors 92, etc.) and/or an object processing unit (i.e. Object Processing Unit 93, etc.). A sensor may obtain or detect information about its environment. As such, one or more sensors can be used to detect objects and/or their properties in a device's surrounding. In some designs, a sensor may be part of a device whose circumstances are being used for DCADO functionalities. In other designs, a sensor may be part of a remote device whose circumstances are being used for DCADO functionalities. Examples of a sensor include a camera (i.e. Camera 92a, etc.), a microphone (i.e. Microphone 92b, etc.), a lidar (i.e. Lidar 92c, etc.), a radar (i.e. Radar 92d, etc.), a sonar (i.e. Sonar 92e, etc.), and/or others. An object processing unit may process output from a sensor to obtain information of interest. As such, an object processing unit can be used to process output from a sensor to detect objects and/or their properties in a device's surrounding. In some aspects, an object processing unit may create or generate a collection of object representations. In other aspects, an object processing unit may create or generate a stream of collections of object representations. An object processing unit may include a picture recognizer (i.e. Picture Recognizer 94a, etc.), a sound recognizer (i.e. Sound Recognizer 94b, etc.), a lidar processing unit (i.e. Lidar Processing Unit 94c, etc.), a radar processing unit (i.e. Radar Processing Unit 94d, etc.), a sonar processing unit (i.e. Sonar Processing Unit 94e, etc.), and/or other elements or functionalities. In general, an object processing unit may include any signal processing element or technique known in art as applicable. In some implementations, an object processing unit and/or any of its elements or functionalities can be included in sensor and/or other elements. Receiving comprises any action or operation by or for a Collection of Object Representations 525, stream of Collections of Object Representations 525, Object Representation 625, Object Property 630, Sensor 92, Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, Object Processing Unit 93, Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other disclosed elements.

At step 9110, a first one or more instruction sets for operating a device are received. In some embodiments, an instruction set (i.e. Instruction Set 526, etc.) may be used or executed by a processor (i.e. Processor 11, etc.) in operating a device. In other embodiments, an instruction set may be part of an application program (i.e. Application

Autonomous Ex. 2004 p. 440 of 510
Tesla v. Autonomous
IPR2024-00745

Program 18, etc.) used in operating a device. For example, the application can run or execute on one or more processors or other processing elements. In further embodiments, an instruction set may be used or executed by a logic circuit (i.e. Logic Circuit 250, etc.) in operating a device. For example, such instruction set may be or include one or more inputs into or outputs from a logic circuit. In further embodiments, an instruction set may be used by an actuator in operating a device. For example, such instruction set may be or include one or more inputs into an actuator. Operating a device includes performing or causing any operations on/by/with the device. In some designs, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element as the instruction set is being used or executed. In other aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element after the instruction set is used or executed. In further aspects, an instruction set can be received from a processor, application program, logic circuit, and/or other processing element before the instruction set has been used or executed. In further aspects, an instruction set can be received from a running processor, running application program, running logic circuit, and/or other running processing element. As such, an instruction set can be received at runtime. In other designs, an instruction set can be received from an actuator. In further designs, an instruction set can be received from memory (i.e. Memory 12, etc.), hard drive, or any other storage element or repository. In further designs, an instruction set can be received over a network such as Internet, local area network, wireless network, and/or other network. In further designs, an instruction set can be received by an interface (i.e. Acquisition Interface 120, etc.) configured to obtain instruction sets from a processor, application program, logic circuit, actuator, and/or other element. In general, an instruction set can be received by any element of the system. One or more instruction sets may temporally correspond to a collection of object representations. In some aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed at the time of generating the collection of object representations. In other aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed within a certain time period before and/or after generating the collection of object representations. Any time period can be utilized depending on implementation. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating the collection of object representations to the time of generating a next collection of object representations. In further aspects, an instruction set that temporally corresponds to a collection of object representations includes an instruction set used or executed from the time of generating a preceding collection of object representations to the time of generating the collection of object representations. Any other temporal relationship or correspondence between collections of object representations and correlated instruction sets can be implemented. In general, one or more instruction sets that temporally correspond to a collection of object representations enable structuring knowledge of a device's operation at or around the time of generating the collection of object representations. Such functionality enables spontaneous or seamless learning of a device's operation in circumstances including objects with various properties as the device is operated in real life situations. In some embodiments, an instruction set may include one or more commands, keywords, symbols (i.e. parentheses, brackets, commas, semicolons, etc.), instructions, operators (i.e. =, <, >, etc.), variables, values, objects, data structures, functions (i.e. Function1(), FIRST(), MIN(), SQRT(), etc.), parameters, states, signals, inputs, outputs, references thereto, and/or other components. In other embodiments, an instruction set may include source code, bytecode, intermediate code, compiled, interpreted, or otherwise translated code,

Autonomous Ex. 2004 p. 441 of 510
Tesla v. Autonomous
IPR2024-00745

runtime code, assembly code, machine code, and/or any other computer code. In further embodiments, an instruction set may include one or more inputs into and/or outputs from a logic circuit. In further embodiments, an instruction set may include one or more inputs into an actuator. Receiving comprises any action or operation by or for an Acquisition Interface 120, Instruction Set 526, and/or other disclosed elements.

At step 9115, the first collection of object representations is correlated with the first one or more instruction sets for operating the device. In some aspects, individual collections of object representations can be correlated with one or more instruction sets. In other aspects, streams of collections of object representations can be correlated with one or more instruction sets. In further aspects, individual collections of object representations or streams of collections of object representations can be correlated with the aforementioned temporally corresponding instruction sets. In further aspects, a collection of object representations or stream of collections of object representations may not be correlated with any instruction sets. Correlating may include structuring or generating a knowledge cell (i.e. Knowledge Cell 800, etc.) and storing one or more collections of object representations correlated with any instruction sets into the knowledge cell. Therefore, a knowledge cell may include any data structure or arrangement that can facilitate such storing. A knowledge cell includes knowledge (i.e. unit of knowledge, etc.) of how a device operated in a circumstance including objects with various properties. In some designs, extra information (i.e. Extra Info 527, etc.) may optionally be used to facilitate enhanced comparisons or decision making in autonomous device operation where applicable. Therefore, any collection of object representations, instruction set, and/or other element may include or be correlated with extra information. Extra information may include any information useful in comparisons or decision making performed in autonomous device operation. Examples of extra information include time information, location information, computed information, contextual information, and/or other information. Correlating may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Correlating comprises any action or operation by or for a Knowledge Structuring Unit 520, Knowledge Cell 800, and/or other disclosed elements.

At step 9120, the first collection of object representations correlated with the first one or more instruction sets for operating the device are stored. A collection of object representations correlated with one or more instruction sets may be part of a stored plurality of collections of object representations correlated with one or more instruction sets. Collections of object representations correlated with any instruction sets can be stored in a memory unit or other repository. The aforementioned knowledge cells comprising collections of object representations correlated with any instruction sets can be used in/as neurons, nodes, vertices, or other elements in any of the data structures or arrangements (i.e. neural networks, graphs, sequences, collection of knowledge cells, etc.) used for storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledge cells may be connected, interrelated, or interlinked into knowledge structures using statistical, artificial intelligence, machine learning, and/or other models or techniques. Such interconnected or interrelated knowledge cells can be used for enabling autonomous device operation. The interconnected or interrelated knowledge cells may be stored or organized into a knowledgebase (i.e. Knowledgebase 530, etc.). In some embodiments, knowledgebase may be or include a neural network (i.e. Neural Network 530a, etc.). In other embodiments, knowledgebase may be or include a graph (i.e. Graph 530b, etc.). In further embodiments, knowledgebase may be or include a collection of sequences (i.e. Collection of Sequences 530c, etc.). In further embodiments, knowledgebase may be or include a sequence (i.e. Sequence 533, etc.). In further embodiments, knowledgebase may be or include a collection of knowledge cells

Autonomous Ex. 2004 p. 442 of 510
Tesla v. Autonomous
IPR2024-00745

(i.e. Collection of Knowledge Cells 530d, etc.). In general, knowledgebase may be or include any data structure or arrangement, and/or repository capable of storing the knowledge of a device's operation in circumstances including objects with various properties. Knowledgebase may also include or be substituted with various artificial intelligence methods, systems, and/or models for knowledge structuring, storing, and/or representation such as deep learning, supervised learning, unsupervised learning, neural networks (i.e. convolutional neural network, recurrent neural network, deep neural network, etc.), search-based, logic and/or fuzzy logic-based, optimization-based, tree/graph/other data structure-based, hierarchical, symbolic and/or sub-symbolic, evolutionary, genetic, multi-agent, deterministic, probabilistic, statistical, and/or other methods, systems, and/or models. Storing may be omitted where learning of a device's operation in circumstances including objects with various properties is not implemented. Storing comprises any action or operation by or for a Knowledgebase 530, Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, Knowledge Cell 800, Node 852, Layer 854, Connection 853, Similarity Comparison 125, and/or other disclosed elements.

At step 9125, a new collection of object representations is received. Step 9125 may include any action or operation described in Step 9105 as applicable.

At step 9130, the new collection of object representations is compared with the first collection of object representations. Comparing one collection of object representations with another collection of object representations may include comparing at least a portion of one collection of object representations with at least a portion of the other collection of object representations. In some embodiments, collections of object representations may be compared individually. In some aspects, comparing of individual collections of object representations may include comparing one or more object representations of one collection of object representations with one or more object representations of another collection of object representations. In other aspects, comparing of object representations may include comparing one or more object properties of one object representation with one or more object properties of another object representation. In some designs, one or more object properties in the same category (i.e. Category 635, etc.) can be compared. Comparing may include any techniques for comparing text, numbers, and/or other data. In further aspects, some object representations, object properties, and/or other elements of a collection of object representations can be omitted from comparison depending on implementation. In other embodiments, collections of object representations may be compared collectively as part of streams of collections of object representations. Collective comparing of collections of object representations may include any features, functionalities, and embodiments of the aforementioned individual comparing of collections of object representations. In some aspects, collective comparing of collections of object representations may include comparing one or more collections of object representations of one stream of collections of object representations with one or more collections of object representations of another stream of collections of object representations. In some designs, one or more corresponding (i.e. similarly ordered, temporally related, etc.) collections of object representations from the compared streams of collections of object representations can be compared. In other designs, Dynamic Time Warping (DTW) and/or other techniques can be utilized for comparison and/or aligning temporal sequences (i.e. streams of collections of object representations, etc.) that may vary in time or speed. In further aspects, some collections of object representations can be omitted from comparison depending on implementation. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments of the comparing. Comparing may be omitted where anticipating of a device's operation in

Autonomous Ex. 2004 p. 443 of 510
Tesla v. Autonomous
IPR2024-00745

circumstances including objects with various properties is not implemented. Comparing comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9135, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. In some embodiments, determining at least a partial match between individually compared collections of object representations includes determining that a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other embodiments, determining at least a partial match between individually compared collections of object representations includes determining at least a partial match between one or more portions of one collection of object representations and one or more portions of another collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining substantial similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations. A portion of a collection of object representations may include an object representation, an object property, and/or other portion or element of the collection of object representations. In further embodiments, determining at least a partial match between individually compared collections of object representations includes determining that the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In some aspects, type of object representations, importance of object representations, and/or other elements or techniques relating to object representations can be utilized for determining similarity using object representations. In further aspects, some of the object representations can be omitted in determining similarity using object representations depending on implementation. In further embodiments, determining a match or substantial match between compared object representations includes determining that the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some aspects, categories of object properties, importance of object properties, and/or other elements or techniques relating to object properties can be utilized for determining similarity using object properties. In further aspects, some of the object properties can be omitted in determining similarity using object properties depending on implementation. In some designs, substantial similarity of individually compared collections of object representations can be achieved when a similarity between one or more portions of one collection of object representations and one or more portions of another collection of object representations exceeds a similarity threshold. In other designs, substantial similarity of individually compared collections of object representations can be achieved when the number or percentage of matching or substantially matching object representations of the compared collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 7, 18, etc.) or threshold percentage (i.e. 41%, 62%, 79%, 85%, 93%, etc.). In further aspects, substantial similarity of compared object representations can be achieved when the number or percentage of matching or substantially matching object properties of the compared object representations exceeds a threshold number (i.e. 1, 2, 3, 6, 11, etc.) or a threshold percentage (i.e. 55%, 61%, 78%, 82%, 99%, etc.). In some embodiments, determining at least a partial match between collectively compared collections of object representations (i.e. streams of collections of object representations, etc.) includes determining that the number or percentage of matching or

Autonomous Ex. 2004 p. 444 of 510
Tesla v. Autonomous
IPR2024-00745

substantially matching collections of object representations of the compared streams of collections of object representations exceeds a threshold number (i.e. 1, 2, 4, 9, 33, 138, etc.) or threshold percentage (i.e. 39%, 58%, 77%, 88%, 94%, etc.). In some aspects, importance of collections of object representations, order of collections of object representations, and/or other elements or techniques relating to collections of object representations can be utilized for determining similarity of collectively compared collections of object representations or streams of collections of object representations. In further aspects, some of the collections of object representations can be omitted in determining similarity of collectively compared collections of object representations or streams of collections of object representations depending on implementation. In some designs, a threshold for a number or percentage similarity can be used to determine a match or substantial match between any of the aforementioned elements. Any text, number, and/or other data similarity determination techniques can be used in any of the aforementioned similarity determinations. A partial match of any of the compared elements may include a substantially or otherwise similar match, and vice versa. Therefore, these terms may be used interchangeably herein depending on context. Although, substantial similarity or substantial match is frequently used herein, it should be understood that any level of similarity, however high or low, may be utilized as defined by the rules (i.e. thresholds, etc.) for similarity. Any combination of the aforementioned and/or other elements or techniques can be utilized in alternate embodiments. Determining may be omitted where anticipating of a device's operation in circumstances including objects with various properties is not implemented. Determining comprises any action or operation by or for a Decision-making Unit 540, Similarity Comparison 125, and/or other disclosed elements.

At step 9140, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Executing may be performed in response to the aforementioned determining. Executing may be caused by DCADO Unit 100, Artificial Intelligence Unit 110, Modification Interface 130, and/or other disclosed elements. An instruction set may be executed by a processor (i.e. Processor 11, etc.), application program (i.e. Application Program 18, etc.), logic circuit (i.e. Logic Circuit 250, etc.), and/or other processing element. An instruction set may be executed or acted upon by an actuator. In some aspects, instruction sets (i.e. the one or more instruction sets for operating the device correlated with the first collection of object representations, etc.) anticipated or determined to be used or executed in a device's autonomous operation may be referred to as anticipatory instruction sets, alternate instruction sets, and/or other suitable name or reference. Therefore, these terms can be used interchangeably herein depending on context. Executing may include executing one or more alternate instruction sets (i.e. anticipatory instruction sets, etc.) instead of or prior to an instruction set that would have been executed in a regular course of execution. In some embodiments, executing may include modifying a register or other element of a processor with one or more alternate instruction sets. Executing may also include redirecting a processor to one or more alternate instruction sets. In other embodiments, processor may be or comprises a logic circuit. Executing may further include modifying an element of a logic circuit with one or more alternate instruction sets, redirecting the logic circuit to one or more alternate instruction sets, replacing the inputs into the logic circuit with one or more alternate inputs or instruction sets, and/or replacing the outputs from the logic circuit with one or more alternate outputs or instruction sets. Executing may further include replacing the inputs into an actuator with one or more alternate inputs or instruction sets. In further embodiments, a processor may run an application including instruction sets for operating a device. In some aspects, executing includes executing one or more alternate instruction sets as part of the application. In other aspects, executing includes modifying the

Autonomous Ex. 2004 p. 445 of 510
Tesla v. Autonomous
IPR2024-00745

application. In further aspects, executing includes redirecting the application to one or more alternate instruction sets. In further aspects, executing includes modifying one or more instruction sets of the application. In further aspects, executing includes modifying the application's source code, bytecode, intermediate code, compiled code, interpreted code, translated code, runtime code, assembly code, machine code, or other code. In further aspects, executing includes modifying memory, processor register, storage, repository or other element where the application's instruction sets are stored or used. In further aspects, executing includes modifying instruction sets used for operating an object of the application. In further aspects, executing includes modifying an element of a processor, an element of a device, a virtual machine, a runtime engine, an operating system, an execution stack, a program counter, or a user input used in running the application. In further aspects, executing includes modifying the application at source code write time, compile time, interpretation time, translation time, linking time, loading time, runtime, or other time. In further aspects, executing includes modifying one or more of the application's lines of code, statements, instructions, functions, routines, subroutines, basic blocks, or other code segments. In further aspects, executing includes a manual, automatic, dynamic, just in time (JIT), or other instrumentation of the application. In further aspects, executing includes utilizing one or more of a .NET tool, .NET application programming interface (API), Java tool, Java API, operating system tool, independent tool, or other tool for modifying the application. In further aspects, executing includes utilizing a dynamic, interpreted, scripting, or other programming language. In further aspects, executing includes utilizing dynamic code, dynamic class loading, or reflection. In further aspects, executing includes utilizing assembly language. In further aspects, executing includes utilizing metaprogramming, self-modifying code, or an application modification tool. In further aspects, executing includes utilizing just in time (JIT) compiling, JIT interpretation, JIT translation, dynamic recompiling, or binary rewriting. In further aspects, executing includes utilizing dynamic expression creation, dynamic expression execution, dynamic function creation, or dynamic function execution. In further aspects, executing includes adding or inserting additional code into the application's code. In further aspects, executing includes modifying, removing, rewriting, or overwriting the application's code. In further aspects, executing includes branching, redirecting, extending, or hot swapping the application's code. Branching or redirecting an application's code may include inserting a branch, jump, or other means for redirecting the application's execution. Executing comprises any action or operation by or for a Processor 11, Application Program 18, Logic Circuit 250, Modification Interface 130, and/or other disclosed elements.

At step 9145, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. The one or more operations may be performed in response to the aforementioned executing. In some aspects, an operation includes any operation that can be performed by/with/on a computing enabled device. In other aspects, an operation includes any operation that can be performed by/with/on an actuator. In further aspects, an operation includes any operation that can be performed by/with/on a computer. In general, an operation includes any operation that can be performed by/with/on a device or element thereof. One of ordinary skill in art will recognize that, while all possible variations of operations by/with/on a device are too voluminous to describe and limited only by the device's design and/or user's utilization, all operations are within the scope of this disclosure in various implementations.

Referring to Fig. 33, the illustration shows an embodiment of a method 9200 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various

Autonomous Ex. 2004 p. 446 of 510
Tesla v. Autonomous
IPR2024-00745

properties and enable autonomous device operation in similar circumstances. Method 9200 may include any action or operation of any of the disclosed methods such as method 9100, 9300, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9200.

At step 9205, a first collection of object representations is received. Step 9205 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9210, a first one or more instruction sets for operating a device are received. Step 9210 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9215, the first collection of object representations correlated with the first one or more instruction sets for operating the device are learned. Step 9215 may include any action or operation described in Step 9115 and/or Step 9120 of method 9100 as applicable.

At step 9220, a new collection of object representations is received. Step 9220 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9225, the first one or more instruction sets for operating the device correlated with the first collection of object representations are anticipated based on at least a partial match between the new collection of object representations and the first collection of object representations. Step 9225 may include any action or operation described in Step 9130 and/or Step 9135 of method 9100 as applicable.

At step 9230, the first one or more instruction sets for operating the device correlated with the first collection of object representations are executed. Step 9230 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9235, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first collection of object representations are performed by the device. Step 9235 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 34, the illustration shows an embodiment of a method 9300 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9300 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9400, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9300.

At step 9305, a first stream of collections of object representations is received. Step 9305 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9310, a first one or more instruction sets for operating a device are received. Step 9310 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9315, the first stream of collections of object representations is correlated with the first one or more instruction sets for operating the device. Step 9315 may include any action or operation described in Step 9115 of method 9100 as applicable.

Autonomous Ex. 2004 p. 447 of 510
Tesla v. Autonomous
IPR2024-00745

At step 9320, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device are stored. Step 9320 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9325, a new stream of collections of object representations is received. Step 9325 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9330, the new stream of collections of object representations is compared with the first stream of collections of object representations. Step 9330 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9335, a determination is made that there is at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations. Step 9335 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9340, the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are executed. Step 9340 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9345, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations are performed by the device. Step 9345 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 35, the illustration shows an embodiment of a method 9400 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9400 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9500, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9400.

At step 9405, a first collection of object representations is received. Step 9405 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9410, a first one or more inputs are received, wherein the first one or more inputs are also received by a logic circuit, and wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9410 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9415, the first collection of object representations is correlated with the first one or more inputs. Step 9415 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9420, the first collection of object representations correlated with the first one or more inputs are stored. Step 9420 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9425, a new collection of object representations is received. Step 9425 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9430, the new collection of object representations is compared with the first collection of object representations. Step 9430 may include any action or operation described in Step 9130 of method 9100 as applicable.

Autonomous Ex. 2004 p. 448 of 510
Tesla v. Autonomous
IPR2024-00745

At step 9435, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9435 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9440, the first one or more inputs correlated with the first collection of object representations are received by the logic circuit. Step 9440 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9445, one or more operations defined by one or more outputs for operating the device produced by the logic circuit are performed by the device. Step 9445 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 36, the illustration shows an embodiment of a method 9500 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9500 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9600, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9500.

At step 9505, a first collection of object representations is received. Step 9505 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9510, a first one or more outputs are received, the first one or more outputs transmitted from a logic circuit, wherein the logic circuit is configured to receive inputs and produce outputs, and wherein the outputs are used for operating a device. Step 9510 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9515, the first collection of object representations is correlated with the first one or more outputs. Step 9515 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9520, the first collection of object representations correlated with the first one or more outputs are stored. Step 9520 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9525, a new collection of object representations is received. Step 9525 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9530, the new collection of object representations is compared with the first collection of object representations. Step 9530 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9535, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9535 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9540, one or more operations defined by the first one or more outputs correlated with the first collection of object representations are performed by the device. Step 9540 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 37, the illustration shows an embodiment of a method 9600 for learning and/or using a device's circumstances for autonomous device operation. In some aspects, the method can be used on a computing

Autonomous Ex. 2004 p. 449 of 510
Tesla v. Autonomous
IPR2024-00745

enabled device or system to enable learning of a device's operation in circumstances including objects with various properties and enable autonomous device operation in similar circumstances. Method 9600 may include any action or operation of any of the disclosed methods such as method 9100, 9200, 9300, 9400, 9500, and/or others. Additional steps, actions, or operations can be included as needed, or some of the disclosed ones can be optionally omitted, or a different combination or order thereof can be implemented in alternate embodiments of method 9600.

At step 9605, a first collection of object representations is received. Step 9605 may include any action or operation described in Step 9105 of method 9100 as applicable.

At step 9610, a first one or more inputs are received, wherein the first one or more inputs are also received by an actuator, and wherein the actuator is configured to receive inputs and perform motions. Step 9610 may include any action or operation described in Step 9110 of method 9100 as applicable.

At step 9615, the first collection of object representations is correlated with the first one or more inputs. Step 9615 may include any action or operation described in Step 9115 of method 9100 as applicable.

At step 9620, the first collection of object representations correlated with the first one or more inputs are stored. Step 9620 may include any action or operation described in Step 9120 of method 9100 as applicable.

At step 9625, a new collection of object representations is received. Step 9625 may include any action or operation described in Step 9125 of method 9100 as applicable.

At step 9630, the new collection of object representations is compared with the first collection of object representations. Step 9630 may include any action or operation described in Step 9130 of method 9100 as applicable.

At step 9635, a determination is made that there is at least a partial match between the new collection of object representations and the first collection of object representations. Step 9635 may include any action or operation described in Step 9135 of method 9100 as applicable.

At step 9640, the first one or more inputs correlated with the first collection of object representations are received by the actuator. Step 9640 may include any action or operation described in Step 9140 of method 9100 as applicable.

At step 9645, one or more motions defined by the first one or more inputs correlated with the first collection of object representations are performed by the actuator. Step 9645 may include any action or operation described in Step 9145 of method 9100 as applicable.

Referring to Fig. 38, in some exemplary embodiments, Device 98 may be or include Loader 98a. Loader 98a may be operated by User 50 in person or remotely. Loader 98a may include or be coupled to one or more Sensors 92 (i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615aa-615ad, and/or other elements or information in Loader's 98a surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Loader's 98a surrounding. Loader 98a may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e.

Autonomous Ex. 2004 p. 450 of 510
Tesla v. Autonomous
IPR2024-00745

operator's, etc.) operating directions and causes desired operations with Loader 98a such as moving, maneuvering, collecting, lifting, unloading, and/or others. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause Loader's 98a arm with bucket to collect a load, one or more motors or other actuators to move or maneuver Loader 98a, lifting system (i.e. hydraulic, pneumatic, mechanical, electrical, etc.) to lift a load, and/or arm with bucket to unload a load. Loader 98a may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Loader's 98a Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Loader's 98a Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Loader's 98a Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Loader's 98a operation. As User 50 operates Loader 98a in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Loader's 98a operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Loader's 98a surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Loader 98a in similar circumstances as in previously learned ones. For instance, Loader 98a comprising DCADO Unit 100 may learn User 50-directed collecting, moving, maneuvering, lifting, unloading, and/or other operations in a circumstance that includes Rock 615aa, Pile of Material 615ab, Person 615ac, Truck 615ad, and/or other Objects 615 among which Loader 98a may need to maneuver and/or with which Loader 98a may need to interact. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Loader 98a may implement collecting, moving, maneuvering, lifting, and/or unloading operations autonomously.

In some embodiments, DCADO Unit 100 may reside on Server 96 accessible over Network 95 as previously described. In such embodiments, any number of Loaders 98a may connect to such remote DCADO Unit

Autonomous Ex. 2004 p. 451 of 510
Tesla v. Autonomous
IPR2024-00745

100 and the remote DCADO Unit 100 may learn their operations in circumstances including objects with various properties. In turn, any number of Loaders 98a can utilize the remote DCADO Unit 100 for autonomous operation in circumstances including objects with various properties. For example, multiple operators (i.e. Users 50, etc.) may operate their Loaders 98a that are configured to transmit their operations in circumstances including objects with various properties to a remote DCADO Unit 100. Such remote DCADO Unit 100 enables learning of the operators' collective knowledge of operating Loaders 98 in circumstances including objects with various properties. Any number of Loaders 98 can utilize such collective knowledge comprised in the remote DCADO Unit 100 for their autonomous operation. Any of the disclosed elements such as Artificial Intelligence Unit 110, Knowledgebase 530, and/or others may reside on Server 96, and any combination of local and remote elements can be implemented in alternate embodiments.

In some embodiments, Loader 98a may include or be coupled to a plurality of Sensors 92 and/or their corresponding Object Processing Units 93. In one example, multiple Sensors 92 may detect objects and/or their properties from different angles or on different sides of Loader 98a. In another example, one or more Sensors 92 may be placed on different sub-devices, sub-systems, or elements of Loader 98a. For instance, one Sensor 92 may be placed on the roof of Loader 98a, another Sensor 92 may be placed on the arm of Loader 98a, and an additional Sensor 92 may be placed on the bucket of Loader 98a. In some designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, multiple DCADO Units 100 can be utilized (i.e. one DCADO Unit 100 for each Sensor 92 or group of Sensors 92 and/or their corresponding Object Processing Units 93, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn operations of Loader's 98a sub-device, sub-system, or element in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected by Sensor 92 on the sub-device, sub-system, or element assigned to the DCADO Unit 100 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual sub-device, sub-system, or element. In other designs where multiple Sensors 92 are placed on different sub-devices, sub-systems, or elements of Loader 98a, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating collective Collections of Object Representations 525 representing Objects 615 detected by Sensors 92 on the sub-devices, sub-systems, or elements with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element.

In some embodiments, Loader 98a may include a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements. In some aspects, one or more sub-devices, sub-systems, or elements of Loader 98a may be controlled by different processing elements. For example, one Processor 11 (i.e. including any Application Programs 18 running thereon, etc.) may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, one Logic Circuit 250 may control an arm of Loader 98a, and an additional Logic Circuit 250 may control a bucket of Loader 98a. In some designs where multiple processing elements are utilized, multiple DCADO Units 100 can also be utilized (i.e. one DCADO Unit 100 for each processing element, etc.). In such designs, as User 50 operates Loader 98a in circumstances including objects with various properties, a particular DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object

Autonomous Ex. 2004 p. 452 of 510
Tesla v. Autonomous
IPR2024-00745

Representations 525 with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element assigned to the DCADO Unit 100. The learning and/or decision making in Loader's 98a operation can, therefore, be performed per individual processing element. In other designs where multiple processing elements are utilized, as User 50 operates Loader 98a in circumstances including objects with various properties, a single DCADO Unit 100 may learn Loader's 98a operations in these circumstances by correlating Collections of Object Representations 525 with collective Instruction Sets 526 used or executed by a plurality of Logic Circuits 250, Processors 11, Application Programs 18, and/or other processing elements.

In some embodiments, a combination of DCADO Unit 100 and other systems and/or techniques can be utilized to implement Loader's 98a operation. In one example, DCADO Unit 100 may be a primary or preferred system for implementing Loader's 98a operation. While operating autonomously under the control of DCADO Unit 100, Loader 98a may encounter a circumstance including objects with various properties that has not been encountered or learned before. In such situations, User 50 and/or non-DCADO system may take control of Loader's 98a operation. DCADO Unit 100 may take control again when Loader 98a encounters a previously learned circumstance including objects with various properties. Naturally, DCADO Unit 100 can learn Loader's 98a operation in the circumstances while User 50 and/or non-DCADO system is in control of Loader 98a, thereby reducing or eliminating the need for future involvement of User 50 and/or non-DCADO system. For instance, one User 50 can control or assist in controlling multiple Loaders 98a comprising DCADO Units 100. In such instances, User 50 can control or assist in controlling a Loader 98a that may encounter a circumstance including objects with various properties that has not been encountered or learned before while the Loaders 98a operating in previously learned circumstances can operate autonomously. In another example, User 50 and/or non-DCADO system may be a primary or preferred system for implementing Loader's 98a operation. While operating under the control of User 50 and/or non-DCADO system, User 50 and/or non-DCADO system may release control to DCADO Unit 100 for any reason (i.e. User 50 gets tired or distracted, non-DCADO system gets stuck or cannot make a decision, etc.), at which point Loader 98a can be controlled by DCADO Unit 100. In some designs, DCADO Unit 100 may take control in certain special circumstances including objects with various properties where DCADO Unit 100 may offer superior performance even though User 50 and/or non-DCADO system may generally be preferred. Once Loader 98a leaves such special circumstances, DCADO Unit 100 may release control to User 50 and/or non-DCADO system. In general, DCADO Unit 100 can take control from, share control with, or release control to User 50, non-DCADO system, and/or other system or process at any time, in any circumstances, and remain in control for any period of time as needed.

In some embodiments, DCADO Unit 100 may control one or more sub-devices, sub-systems, or elements of Loader 98a while User 50 and/or non-DCADO system may control other one or more sub-devices, sub-systems, or elements of Loader 98a. For example, User 50 and/or non-DCADO system may control the moving system (i.e. drivetrain, powertrain, etc.) of Loader 98a, while DCADO Unit 100 may control an arm and bucket of Loader 98a. Any other combination of controlling various sub-devices, sub-systems, or elements of Loader 98a by DCADO Unit 100 and User 50 and/or non-DCADO system can be implemented.

Referring to Fig. 39, in some exemplary embodiments, Device 98 may be or include Boat 98b. Boat 98b may be operated by User 50 in person or remotely. Boat 98b may include or be coupled to one or more Sensors 92

Autonomous Ex. 2004 p. 453 of 510
Tesla v. Autonomous
IPR2024-00745

(i.e. collectively referred to as Sensor 92, etc.) such as Camera 92a, Microphone 92b, Lidar 92c, Radar 92d, Sonar 92e, etc. and/or Object Processing Unit 93 that can detect Objects 615ba-615bd, and/or other elements or information in Boat's 98b surrounding. Object Processing Unit 93 may include Picture Recognizer 94a, Sound Recognizer 94b, Lidar Processing Unit 94c, Radar Processing Unit 94d, Sonar Processing Unit 94e, and/or other elements or functionalities as applicable. Object Processing Unit 93 may create or generate one or more (i.e. stream, etc.) Collections of Object Representations 525 comprising Object Representations 625, Object Properties 630, and/or other elements or information representing Objects 615 detected in Boat's 98b surrounding. Boat 98b may also include or be controlled by Logic Circuit 250 (i.e. microcontroller, etc.), Processor 11 (i.e. including any Application Program 18 running thereon, etc.), and/or other processing element that receives User's 50 (i.e. operator's, etc.) operating directions and causes desired operations with Boat 98b such as moving, maneuvering, and/or other operations. User 50 can interact with Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element through inputting operating directions via Human-machine Interface 23 such as one or more steering wheels, levers, pedals, buttons, or other input devices. For instance, responsive to User's 50 manipulating a steering wheel and one or more levers, Logic Circuit 250 or Processor 11 may cause one or more motors or other actuators to move or maneuver Boat 98b. Boat 98b may also include or be coupled to DCADO Unit 100. DCADO Unit 100 may be embedded (i.e. integrated, etc.) into or coupled to Boat's 98b Logic Circuit 250, Processor 11, and/or other processing element. DCADO Unit 100 may also be a program embedded (i.e. integrated, etc.) into or interfaced with Application Program 18 running on Processor 11 and/or other processing element. DCADO Unit 100 can obtain Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. In some aspects, Instruction Sets 526 may include one or more inputs into or outputs from Boat's 98b Logic Circuit 250 (i.e. microcontroller, etc.). In other aspects, Instruction Sets 526 may include one or more instruction sets from Boat's 98b Processor's 11 registers or other components. In further aspects, Instruction Sets 526 may include one or more instruction sets used or executed in Application Program 18. DCADO Unit 100 may also optionally obtain any Extra Info 527 (i.e. time, location, computed, contextual, and/or other information, etc.) related to Boat's 98b operation. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Boat's 98b surrounding with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar circumstances as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in

Autonomous Ex. 2004 p. 454 of 510
Tesla v. Autonomous
IPR2024-00745

a circumstance that includes Fishing Boat 615ba, Lighthouse 615bb, Sailboat 615bc, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when a circumstance that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously. In some aspects, the shore (not enumerated) or any part thereof (i.e. cliff, ridge, beach, etc.) may be detected as an Object 615 itself, which may then be learned and used in autonomous operation of Boat 98b.

Referring to Fig. 40, in some exemplary embodiments, an Area of Interest 450 can be utilized. In one example, Area of Interest 450 may include a radial, circular, elliptical, or other such area around Boat 98b. In another example, Area of Interest 450 may include a triangular, rectangular, octagonal, or other such area around Boat 98b. In a further example, Area of Interest 450 may include a spherical, cubical, pyramid-like, or other such area around Boat 98b as applicable to 3D space. Any other Area of Interest 450 shape can be utilized depending on implementation. The shape and/or size of Area of Interest 450 can be defined by a user, by DCADO system administrator, or automatically by the system based on experience, testing, inquiry, analysis, synthesis, or other techniques, knowledge, or input. Utilizing Area of Interest 450 enables DCADO Unit 100 to focus on Boat's 98b immediate surrounding, thereby avoiding extraneous detail in the rest of the surrounding. In some aspects, Area of Interest 450 can be subdivided into sub-areas (i.e. sub-circles, sub-rectangles, sub-spheres, etc.). Sub-areas can be used to classify the surrounding by distance from Boat 98b. For example, the surrounding closer to Boat 98b may be more important and may be assigned higher importance index or weight. As User 50 operates Boat 98b in circumstances including objects with various properties as shown, DCADO Unit 100 may learn Boat's 98b operations in these circumstances by correlating Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with one or more Instruction Sets 526 used or executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element. Any Extra Info 527 related to Boat's 98b operation may also optionally be correlated with Collections of Object Representations 525. DCADO Unit 100 may store this knowledge into Knowledgebase 530 (i.e. Neural Network 530a, Graph 530b, Collection of Sequences 530c, Sequence 533, Collection of Knowledge Cells 530d, etc.). In the future, DCADO Unit 110 may compare incoming Collections of Object Representations 525 representing Objects 615 detected in Area of Interest 450 around Boat 98b with previously learned Collections of Object Representations 525 including optionally using any Extra Info 527 for enhanced decision making. If substantially similar or at least a partial match is found or determined, the Instruction Sets 526 correlated with the previously learned Collections of Object Representations 525 can be autonomously executed by Logic Circuit 250, Processor 11, Application Program 18, and/or other processing element, thereby enabling autonomous operation of Boat 98b in similar Areas of Interest 450 as in previously learned ones. For instance, Boat 98b comprising DCADO Unit 100 may learn User 50-directed moving, maneuvering, and/or other operations in an Area of Interest 450 that includes Fishing Boat 615ba, Lighthouse 615bb, Cruise Ship 615bd, and/or other Objects 615 among which Boat 98b may need to maneuver. In the future, when an Area of Interest 450 that includes Objects 615 with similar Object Properties 630 is encountered, Boat 98b may implement moving, maneuvering, and/or other operations autonomously.

The features, functionalities, and embodiments described with respect to Loader 98a and Boat 98b can be implemented in any situation where Device 98 may need to autonomously maneuver among, interact with, or perform other operations relative to objects in its surrounding. Therefore, the features, functionalities, and

Autonomous Ex. 2004 p. 455 of 510
Tesla v. Autonomous
IPR2024-00745

embodiments described with respect to Loader 98a and Boat 98b can similarly be implemented on any computing enabled machine such as a bulldozer, an excavator, a crane, a forklift, a truck, a construction machine, an assembly machine, an object handling machine, an object dispensing machine, a sorting machine, a restocking machine, an industrial machine, an agricultural machine, a harvesting machine, a building control system, a home or other appliance, a toy, a robot, a tank, an aircraft, a vessel, a submarine, a ground vehicle, an aerial vehicle, an aquatic vehicle, and/or other computing-enabled machine or system.

In yet some exemplary embodiments, Device 98 may be or include a control device such as a thermostat, control panel, remote or other controller, and/or other control device. For instance, a thermostat comprising DCADO Unit 100 may learn User's 50 setting temperature of an air conditioning system controlled by the thermostat in a circumstance that includes User 50 and/or other persons entering or being present in a room. In the future, when a circumstance that includes User 50 and/or other persons entering or being present in the room is encountered, thermostat may implement setting temperature of the air conditioning system autonomously. In some aspects, a control device may be included in the device being controlled (i.e. control panel of an oven, refrigerator, fixture, etc.). In other aspects, a control device may be separate from the device being controlled (i.e. remote controller of a television device, etc.). In yet further exemplary embodiments, Device 98 may be or include a mobile computer such as a smartphone, tablet, and/or other mobile computer. For instance, a smartphone comprising DCADO Unit 100 may learn User 50-directed playing a music file, setting a vibrate mode, and/or other operations in a circumstance that includes objects with various properties. In the future, when a circumstance that includes objects with similar properties is encountered, smartphone may implement playing music file, setting vibrate mode, and/or other operations autonomously. In general, Device 98 may be or include any movable, stationary, or other device. One of ordinary skill in art will understand that Device 98 may be or include any device that can implement and/or benefit from the functionalities described herein.

It must be noted that as used herein and in the appended claims, the singular forms "a", "an", and "the" include plural referents unless the context clearly dictates otherwise.

A number of embodiments have been described herein. While this disclosure contains many specific implementation details, these should not be construed as limitations on the scope of the disclosure or of what may be claimed, but rather as descriptions of features specific to particular embodiments. It should be understood that various modifications can be made without departing from the spirit and scope of the disclosure. The logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other or additional steps, elements, or connections can be included, or some of the steps, elements, or connections can be eliminated, or a combination thereof can be utilized in the described flows, illustrations, or descriptions. Further, the various aspects of the disclosed devices, apparatuses, systems, and/or methods can be combined in whole or in part with each other to produce additional implementations. Moreover, separation of various components in the embodiments described herein should not be understood as requiring such separation in all embodiments, and it should be understood that the described components can generally be integrated together in a single product or packaged into multiple products. Accordingly, other embodiments are within the scope of the following claims.

Autonomous Ex. 2004 p. 456 of 510
Tesla v. Autonomous
IPR2024-00745

CLAIMS

Claim 1. A system for learning and using a device's circumstances for autonomous device operating, the system implemented at least in part on one or more computing devices, the system comprising:

a processor circuit configured to execute instruction sets for operating a device;

a memory unit configured to store data;

a sensor configured to detect objects; and

an artificial intelligence unit configured to:

receive a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

receive a first one or more instruction sets for operating the device;

learn the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receive a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipate the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

Autonomous Ex. 2004 p. 457 of 510
Tesla v. Autonomous
IPR2024-00745

cause the processor circuit to execute the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating of the artificial intelligence unit, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations performed in response to the executing by the processor circuit.

Claim 2. The system of Claim 1, wherein the receiving the first one or more instruction sets for operating the device includes a tracing of the processor circuit or a tracing of a component of the processor circuit.

Claim 3. The system of Claim 1, further comprising:

an application including the instruction sets for operating the device, the application running on the processor circuit, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from the application.

Claim 4. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes at least one of: a manual, an automatic, a dynamic, or a just in time (JIT) instrumentation of the application.

Autonomous Ex. 2004 p. 458 of 510
Tesla v. Autonomous
IPR2024-00745

Claim 5. The system of Claim 3, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 6. The system of Claim 1, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 7. The system of Claim 1, wherein the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device include a knowledge of how the device operated in a circumstance.

Claim 8. The system of Claim 1, wherein the learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device includes storing the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device into the memory unit, the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device being part of a plurality of streams of collections of object representations correlated with one or more instruction sets for operating the device stored in the memory unit.

Autonomous Ex. 2004 p. 459 of 510
Tesla v. Autonomous
IPR2024-00745

Claim 9. The system of Claim 1, wherein at least one of: the processor circuit, the memory unit, the sensor, or the artificial intelligence unit are part of the device.

Claim 10. The system of Claim 1, wherein the artificial intelligence unit is part of, operating on, or coupled to the processor circuit.

Claim 11. A non-transitory computer storage medium having a computer program stored thereon, the program including instructions that when executed by one or more processor circuits cause the one or more processor circuits to perform operations comprising:

receiving a first stream of collections of object representations, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

receiving a first one or more instruction sets for operating a device;

learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device;

receiving a new stream of collections of object representations, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on

Autonomous Ex. 2004 p. 460 of 510
Tesla v. Autonomous
IPR2024-00745

at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations; and

causing an execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the causing performed in response to the anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, wherein the device performs one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the execution.

Claim 12. The non-transitory computer storage medium of Claim 11, wherein the execution of the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations is performed by the one or more processor circuits or by another one or more processor circuits.

Claim 13. The non-transitory computer storage medium of Claim 11, wherein the receiving the first one or more instruction sets for operating the device includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Autonomous Ex. 2004 p. 461 of 510
Tesla v. Autonomous
IPR2024-00745

Claim 14. The non-transitory computer storage medium of Claim 13, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Claim 15. The non-transitory computer storage medium of Claim 11, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Claim 16. A method comprising:

(a) receiving a first stream of collections of object representations by a processor circuit, the first stream of collections of object representations including one or more collections of representations of objects detected by a sensor;

(b) receiving a first one or more instruction sets for operating a device by the processor circuit;

(c) learning the first stream of collections of object representations correlated with the first one or more instruction sets for operating the device, the learning of (c) performed by the processor circuit;

(d) receiving a new stream of collections of object representations by the processor circuit, the new stream of collections of object representations including one or more collections of representations of objects detected by the sensor;

Autonomous Ex. 2004 p. 462 of 510
Tesla v. Autonomous
IPR2024-00745

(e) anticipating the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations based on at least a partial match between the new stream of collections of object representations and the first stream of collections of object representations, the anticipating of (e) performed by the processor circuit;

(f) executing the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the executing of (f) performed in response to the anticipating of (e); and

(g) performing, by the device, one or more operations defined by the first one or more instruction sets for operating the device correlated with the first stream of collections of object representations, the one or more operations by the device performed in response to the executing of (f).

Claim 17. The method of Claim 16, wherein the executing of (f) is performed by the processor circuit or by another processor circuit.

Claim 18. The method of Claim 16, wherein the receiving of (b) includes receiving the first one or more instruction sets for operating the device from an application, the application comprising instruction sets for operating the device.

Claim 19. The method of Claim 18, wherein the receiving the first one or more instruction sets for operating the device from the application includes a tracing of the application.

Autonomous Ex. 2004 p. 463 of 510
Tesla v. Autonomous
IPR2024-00745

Claim 20. The method of Claim 16, wherein the first one or more instruction sets for operating the device include one or more instruction sets that temporally correspond to the first stream of collections of object representations.

Autonomous Ex. 2004 p. 464 of 510
Tesla v. Autonomous
IPR2024-00745

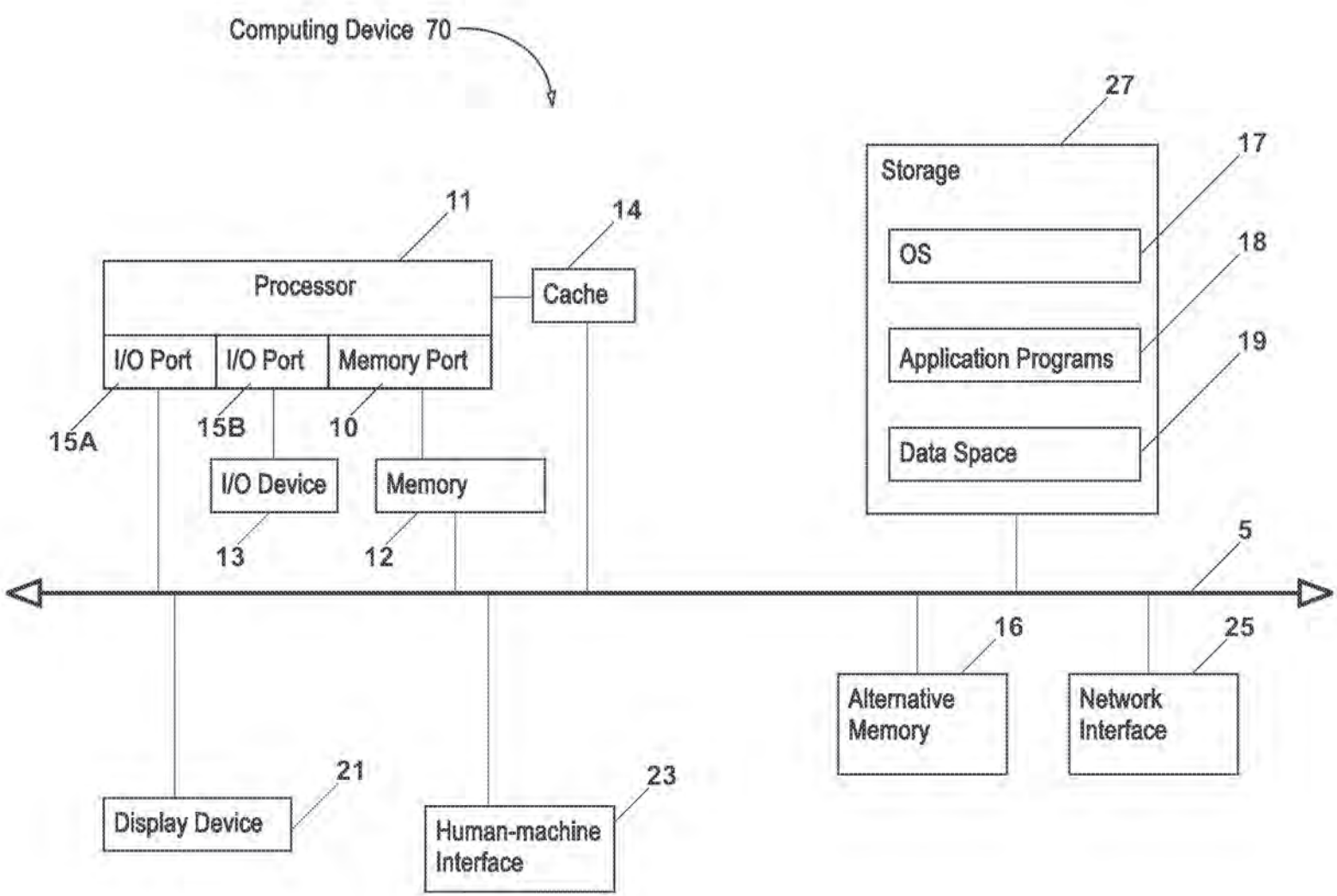


FIG. 1

Autonomous Ex. 2004 p. 465 of 510
Tesla v. Autonomous
IPR2024-00745

98
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
12
Memory
27
Storage
92
Sensor
93
Object Processing Unit
23
Human-machine Interface
Device
Processor
Application Program
11
18
50

FIG. 2

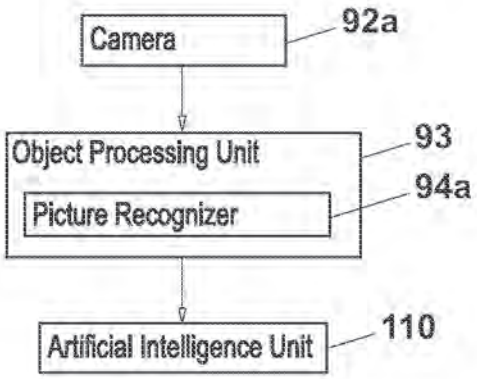


FIG. 3A

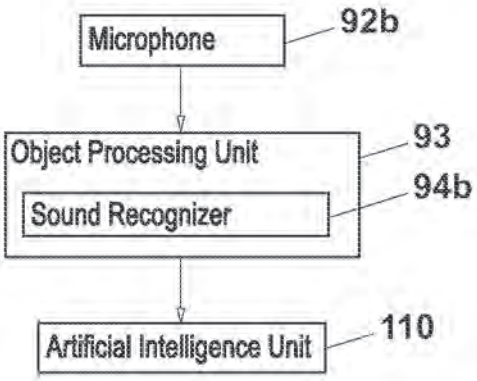


FIG. 3B

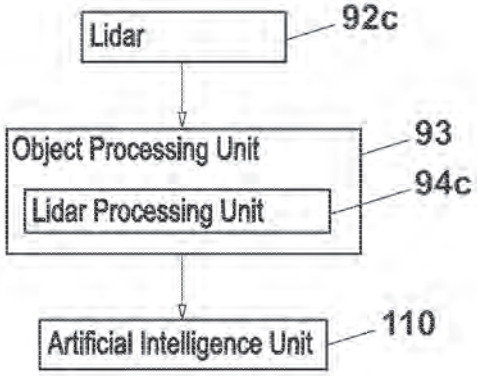


FIG. 3C

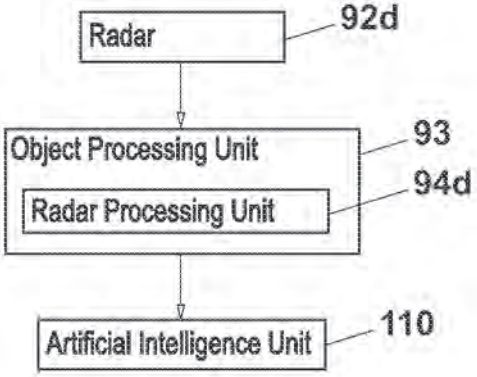


FIG. 3D

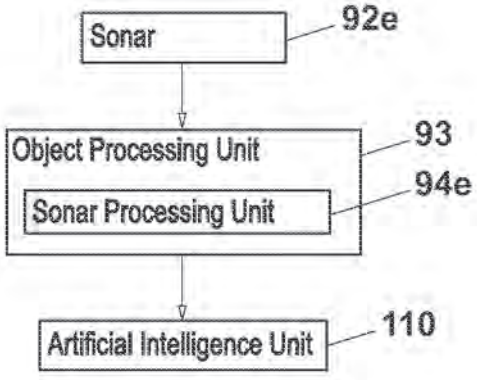


FIG. 3E

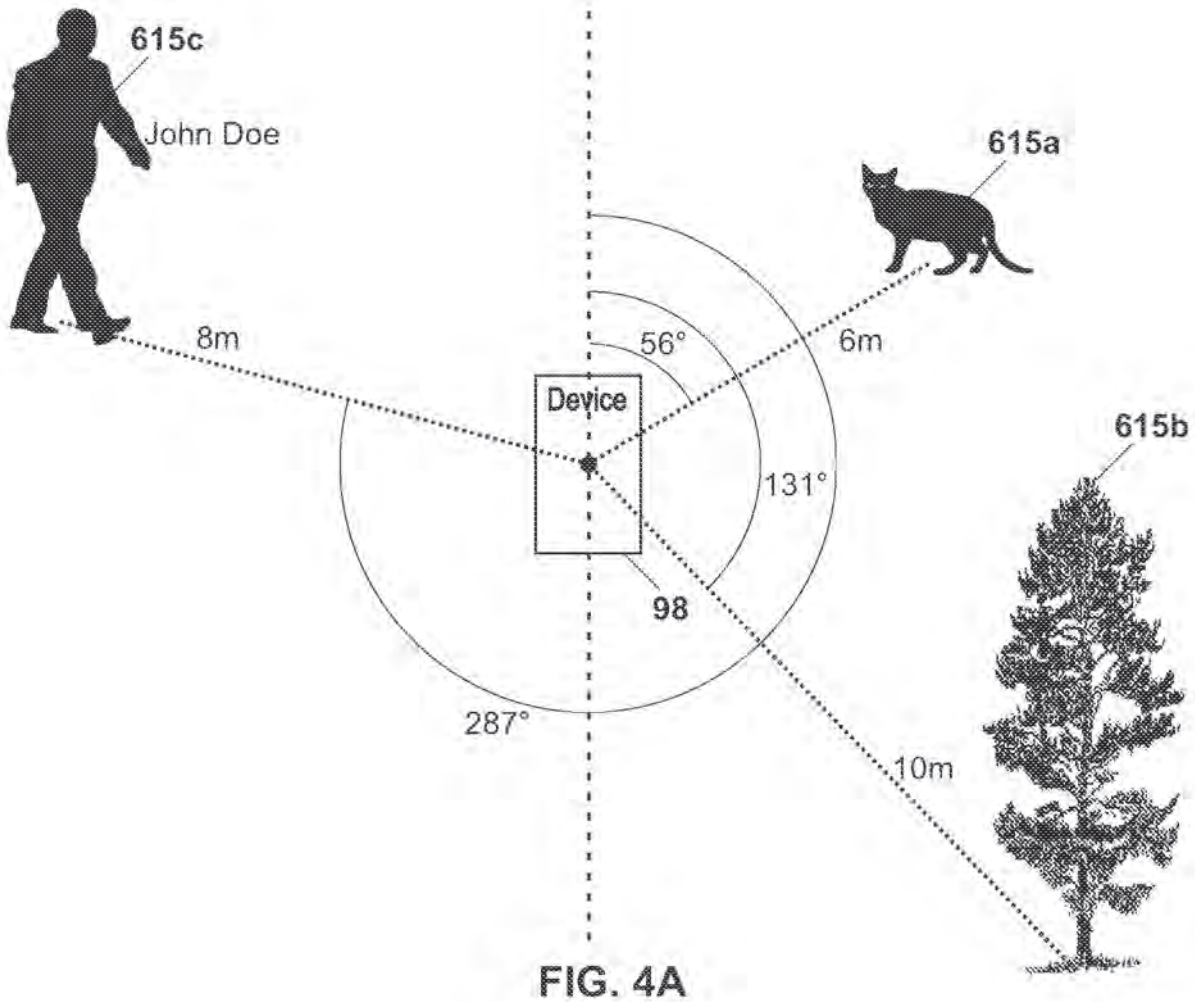

615c
John Doe
615a
8m
56°
6m
Device
615b
131°
98
287°
10m

FIG. 4A

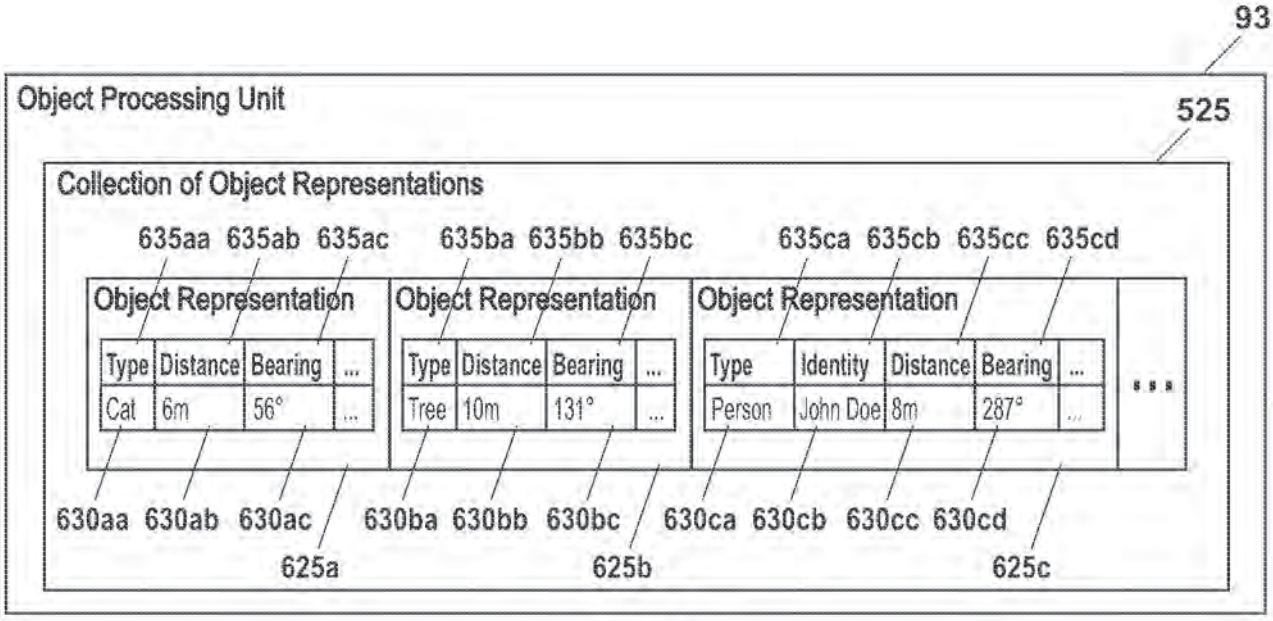

93
Object Processing Unit
525
Collection of Object Representations
635aa 635ab 635ac
635ba 635bb 635bc
635ca 635cb 635cc 635cd
Object Representation
Type | Distance | Bearing | ...
Cat | 6m | 56° | ...
Object Representation
Type | Distance | Bearing | ...
Tree | 10m | 131° | ...
Object Representation
Type | Identity | Distance | Bearing | ...
Person | John Doe | 8m | 287° | ...
630aa 630ab 630ac
625a
630ba 630bb 630bc
625b
630ca 630cb 630cc 630cd
625c

FIG. 4B

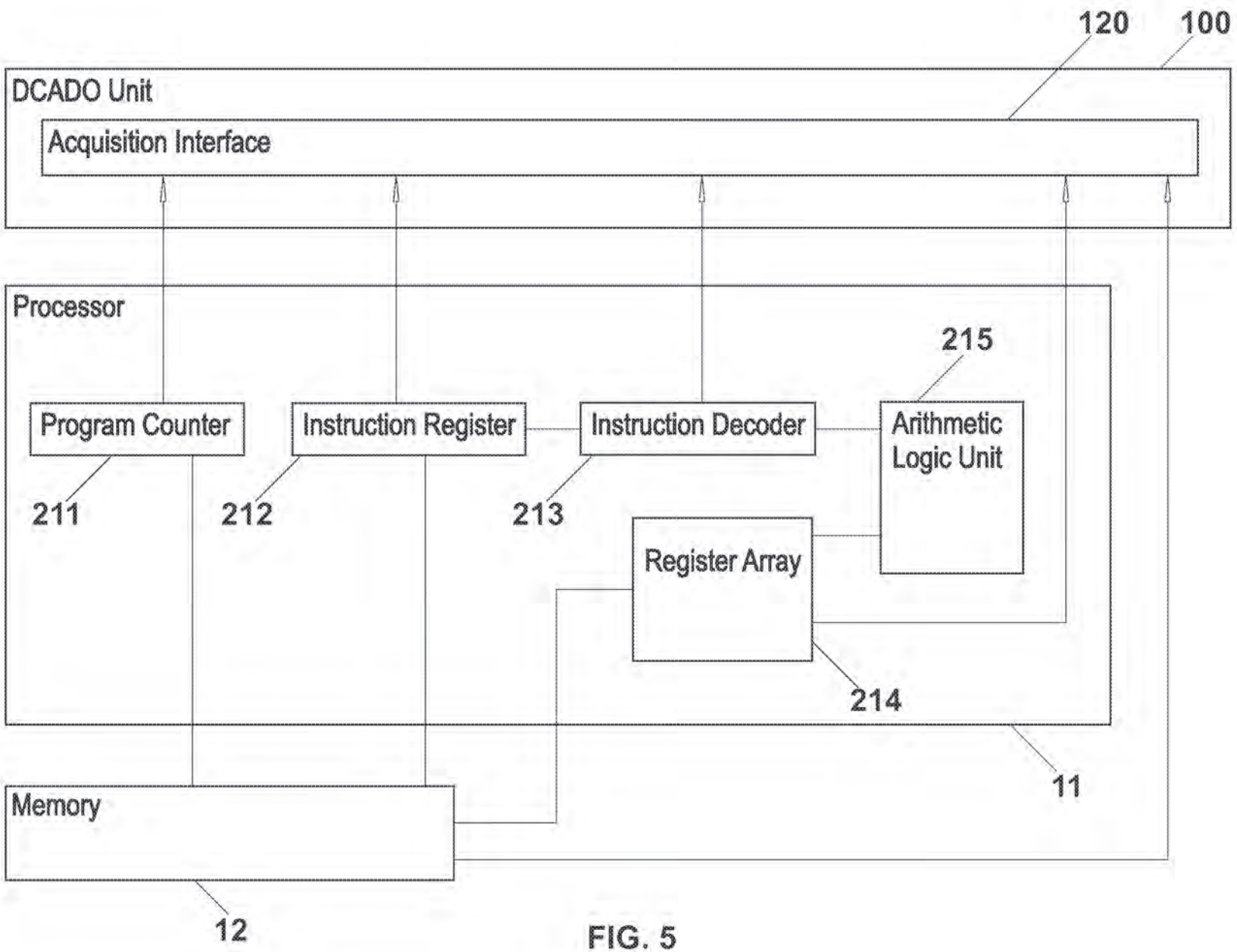


FIG. 5

Autonomous Ex. 2004 p. 469 of 510
Tesla v. Autonomous
IPR2024-00745

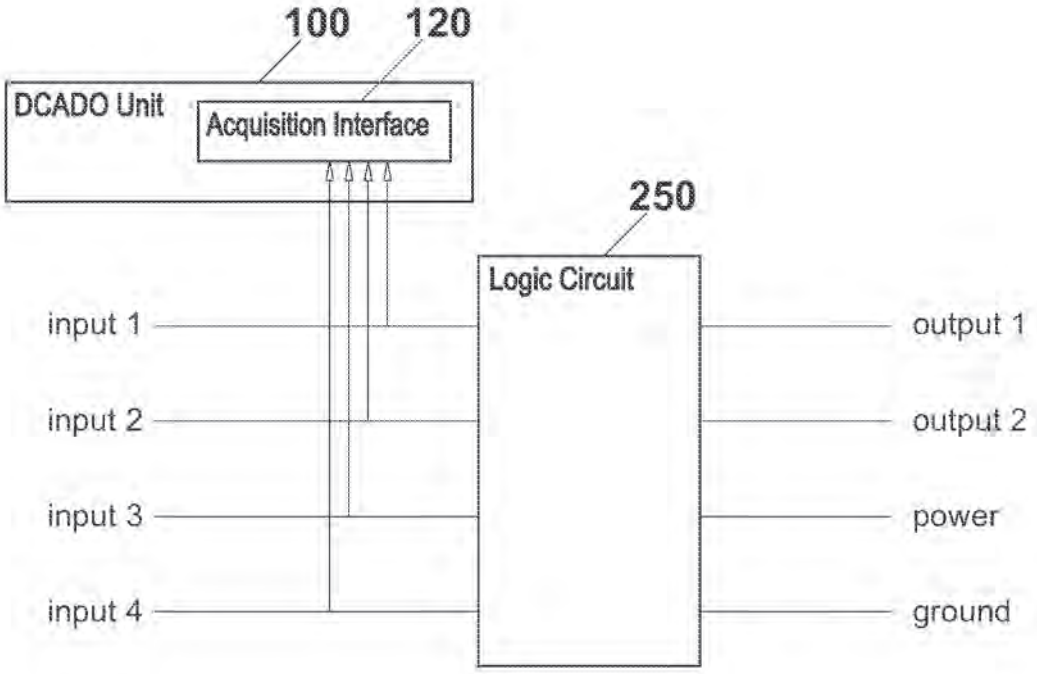

100
120
DCADO Unit
Acquisition Interface
250
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground

FIG. 6A

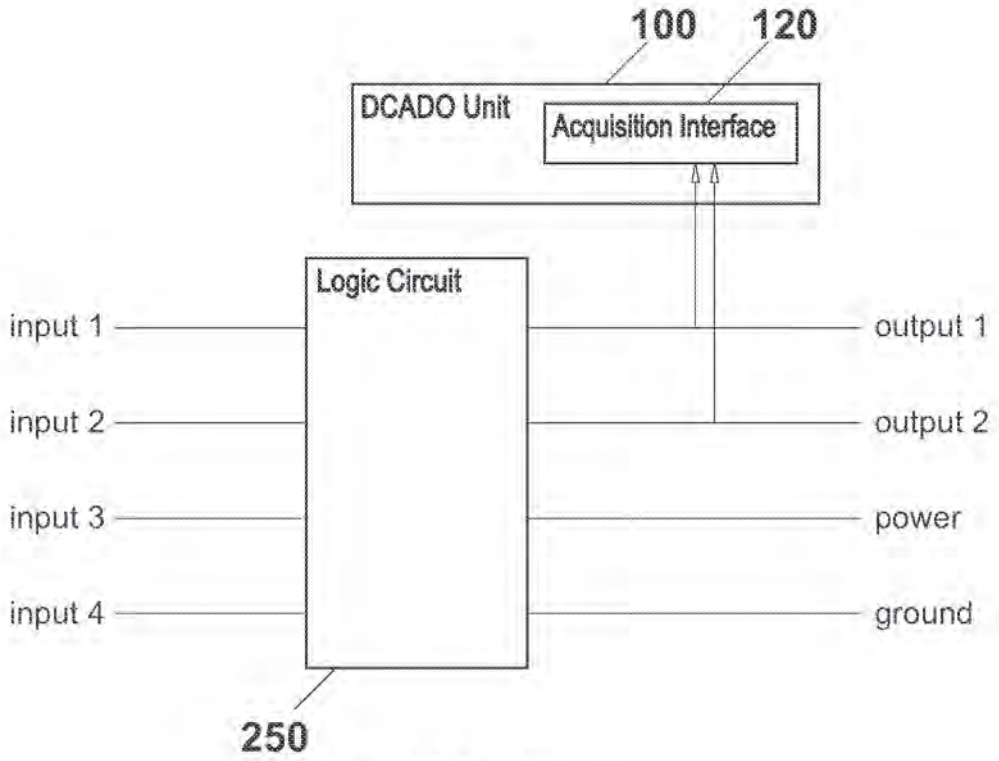

100
120
DCADO Unit
Acquisition Interface
Logic Circuit
input 1
input 2
input 3
input 4
output 1
output 2
power
ground
250

FIG. 6B

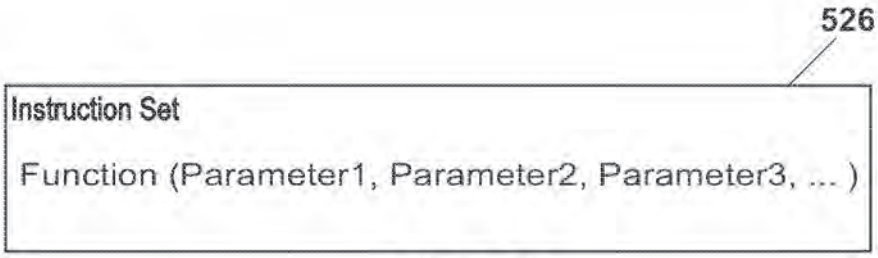


FIG. 7A

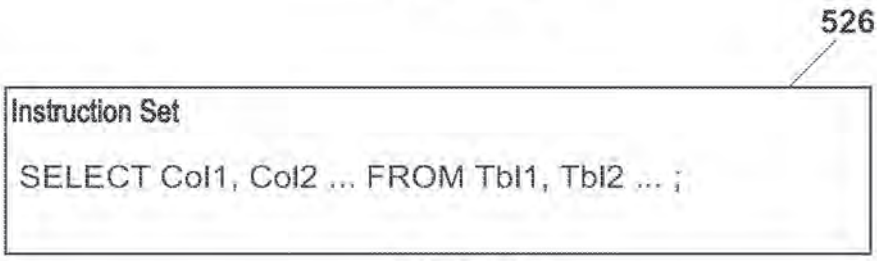


FIG. 7B

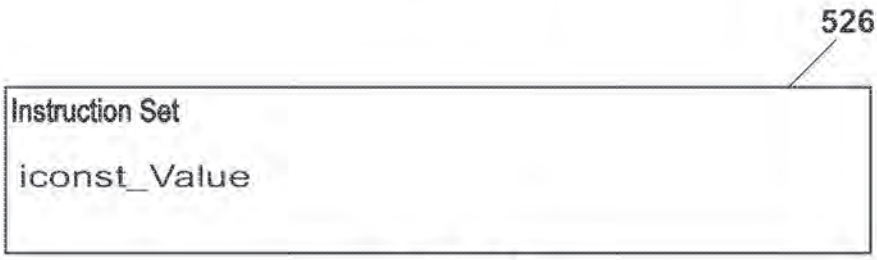


FIG. 7C

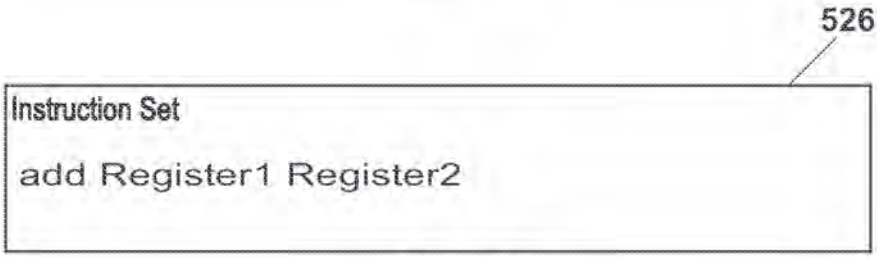


FIG. 7D

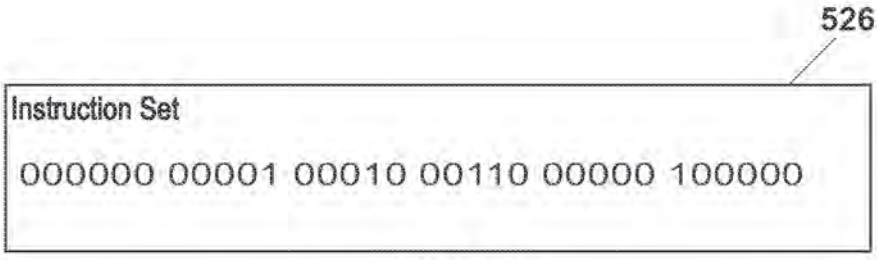


FIG. 7E

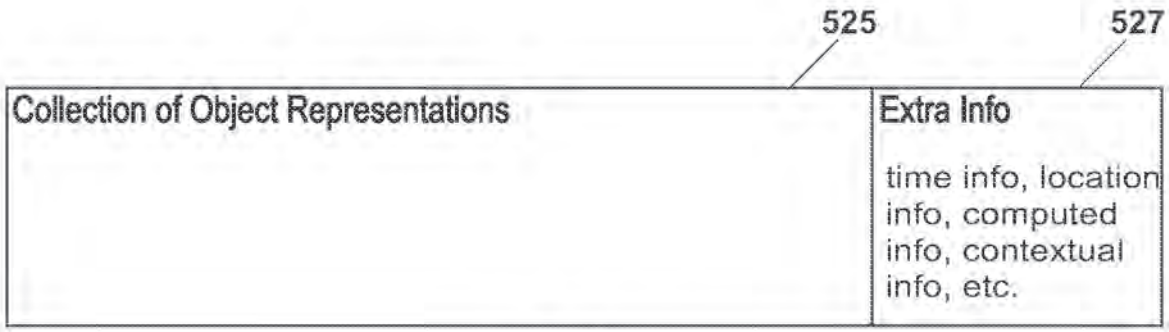

525
527
Collection of Object Representations
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8A

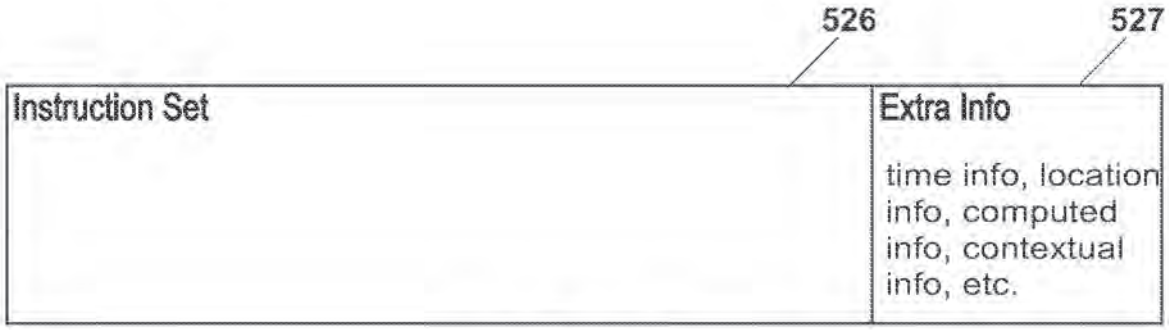

526
527
Instruction Set
Extra Info
time info, location info, computed info, contextual info, etc.

FIG. 8B

98
Device
11
Processor
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
18
Application Program
12
Memory
27
Storage
23
Human-machine Interface
93
Object Processing Unit
92
Sensor
50

FIG. 9

95
Network
96
Server
100
Unit for Learning and/or Using a Device's Circumstances for Autonomous Device Operation (DCADO Unit)
110
Artificial Intelligence Unit
120
Acquisition Interface
130
Modification Interface
98
Device
92
Sensor
93
Object Processing Unit
11
Processor
18
Application Program
12
Memory
27
Storage
23
Human-machine Interface
50

FIG. 10

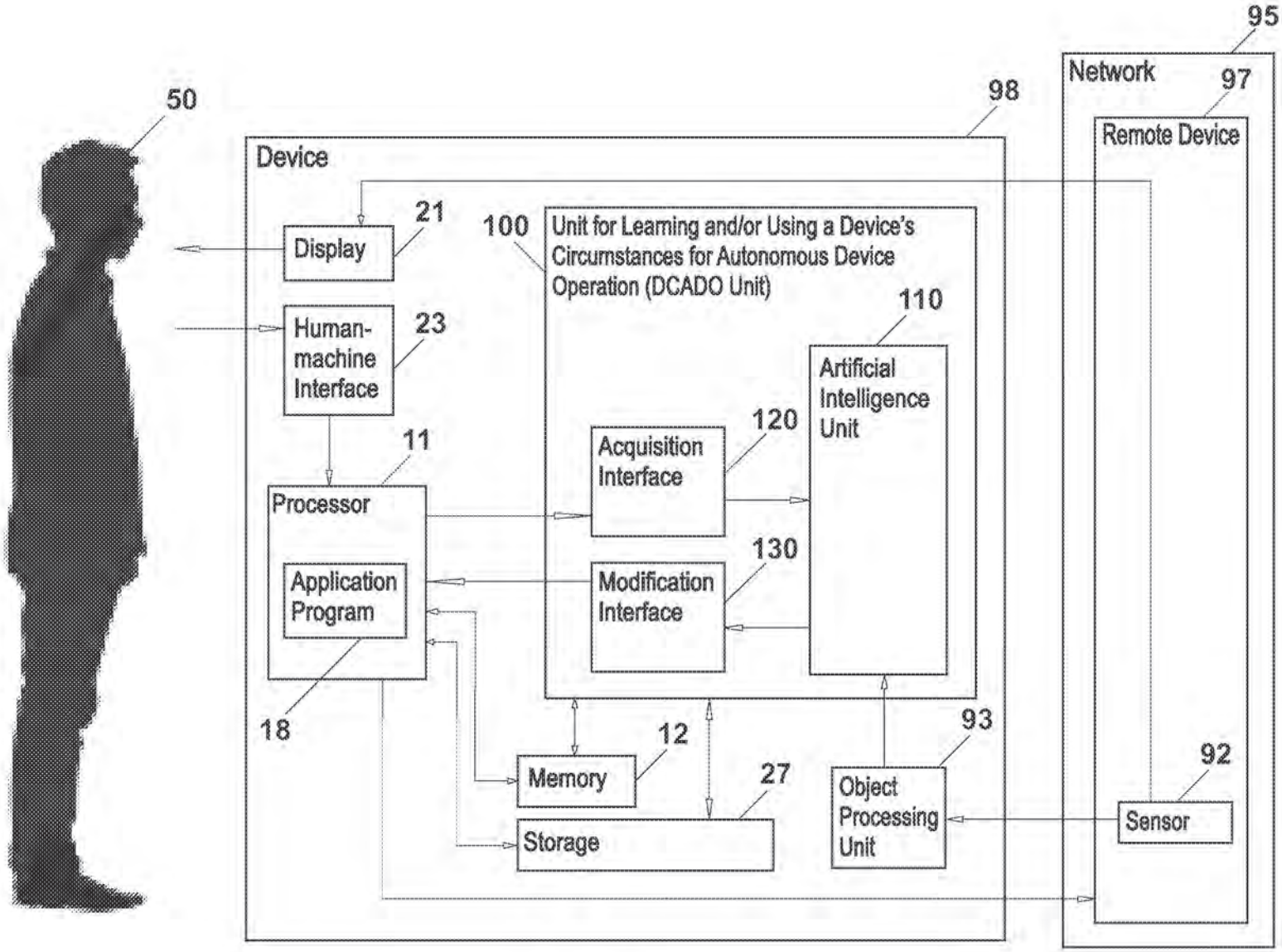


FIG. 11

Autonomous Ex. 2004 p. 475 of 510
Tesla v. Autonomous
IPR2024-00745

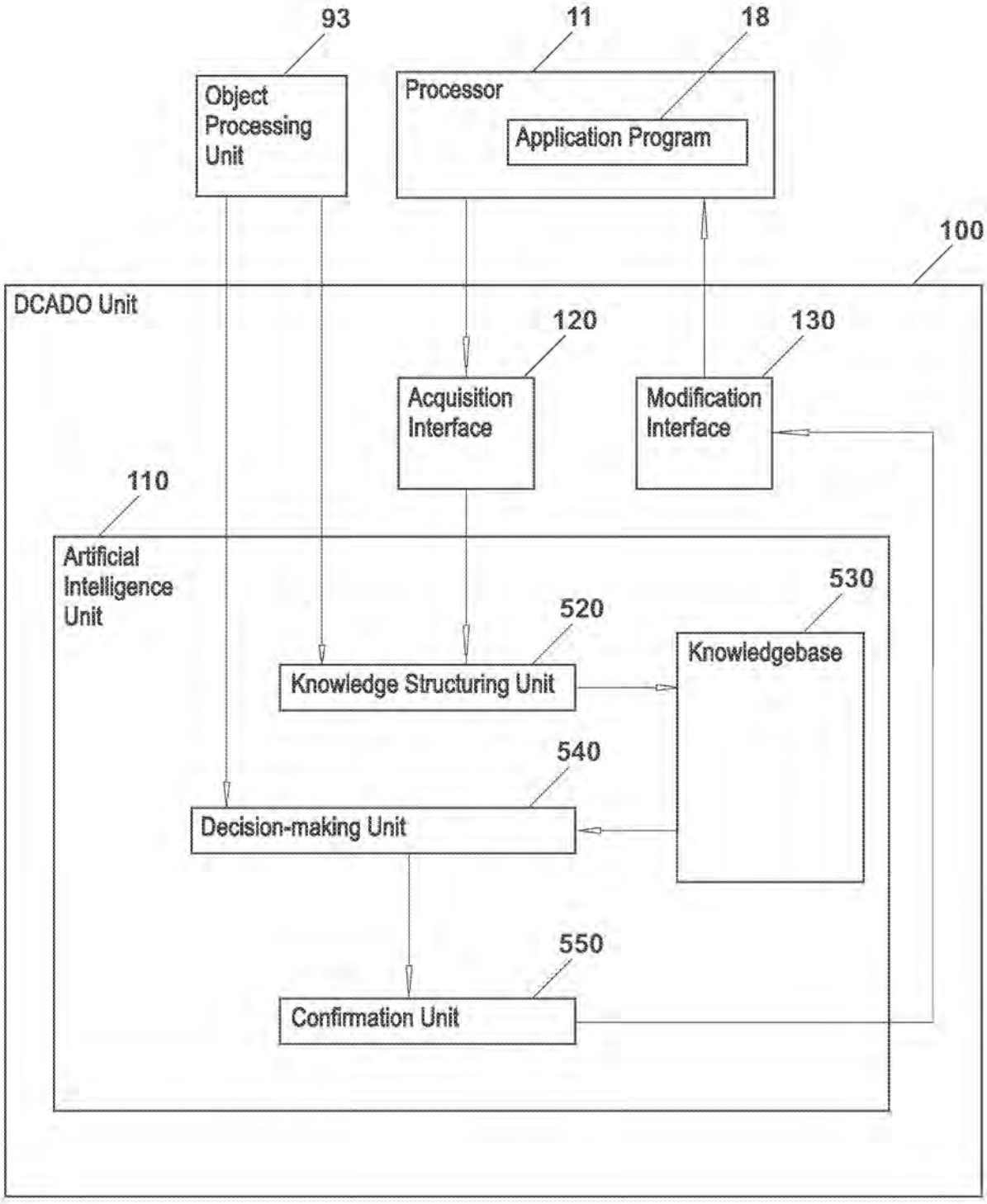

93
11
18
Object Processing Unit
Processor
Application Program
100
DCADO Unit
120
130
Acquisition Interface
Modification Interface
110
Artificial Intelligence Unit
530
520
Knowledgebase
Knowledge Structuring Unit
540
Decision-making Unit
550
Confirmation Unit

FIG. 12

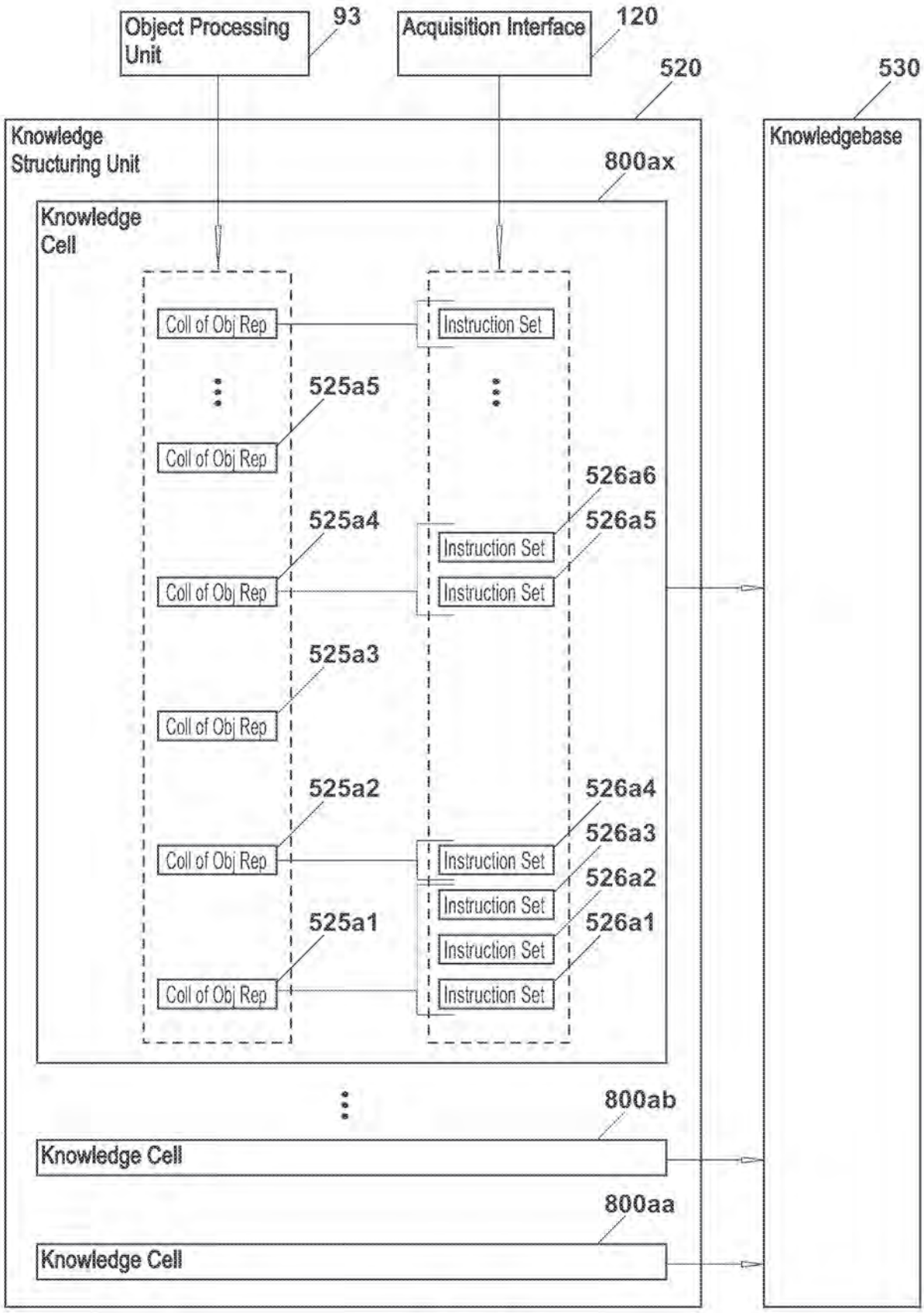

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
800ax
Knowledgebase
Knowledge Cell
Coll of Obj Rep
Instruction Set
525a5
526a6
525a4
526a5
525a3
525a2
526a4
526a3
526a2
525a1
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 13

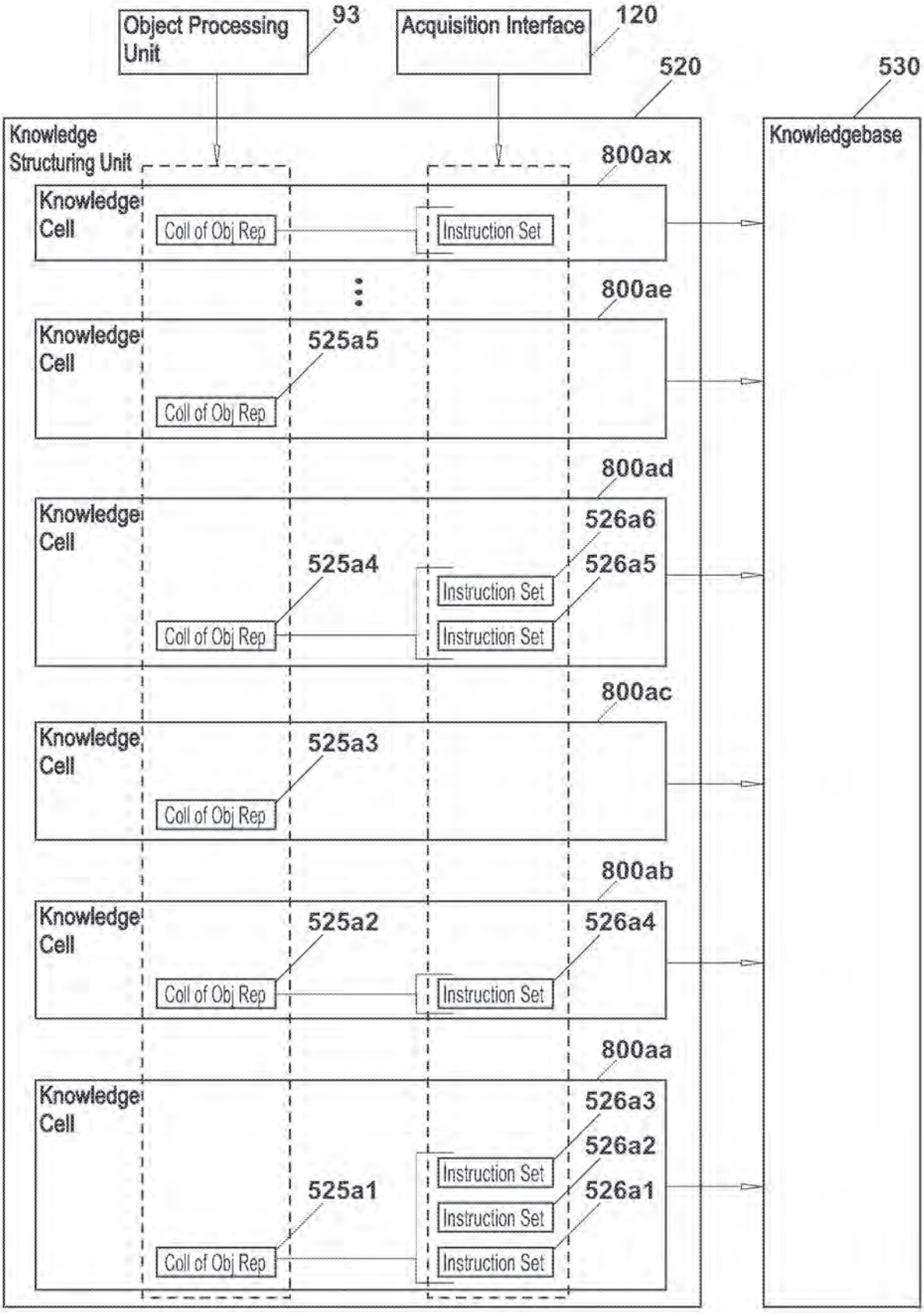

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
800ae
525a5
800ad
526a6
525a4
526a5
800ac
525a3
800ab
525a2
526a4
800aa
526a3
526a2
525a1
526a1

FIG. 14

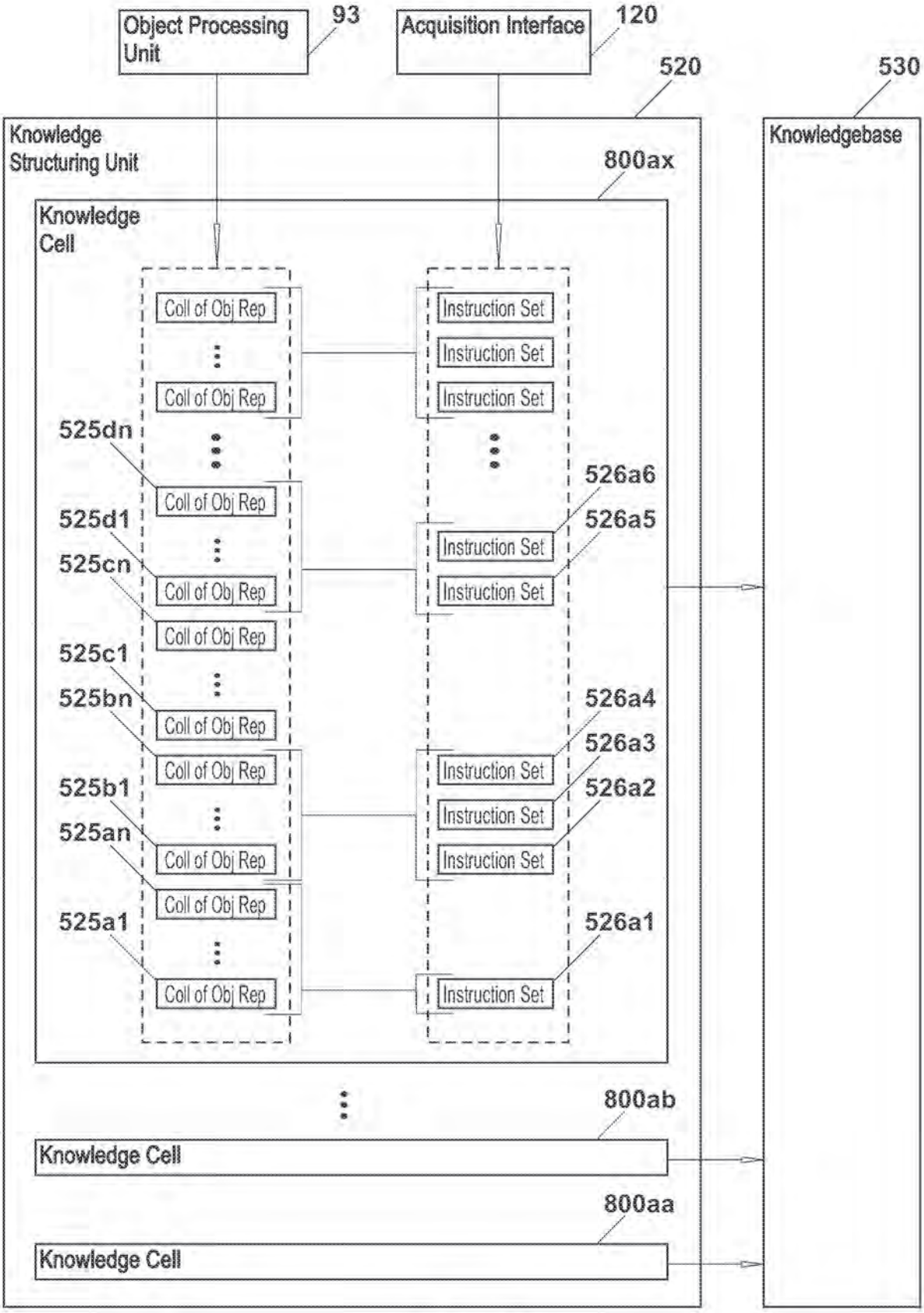

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Instruction Set
525dn
525d1
525cn
525c1
525bn
525b1
525an
525a1
526a6
526a5
526a4
526a3
526a2
526a1
800ab
Knowledge Cell
800aa
Knowledge Cell

FIG. 15

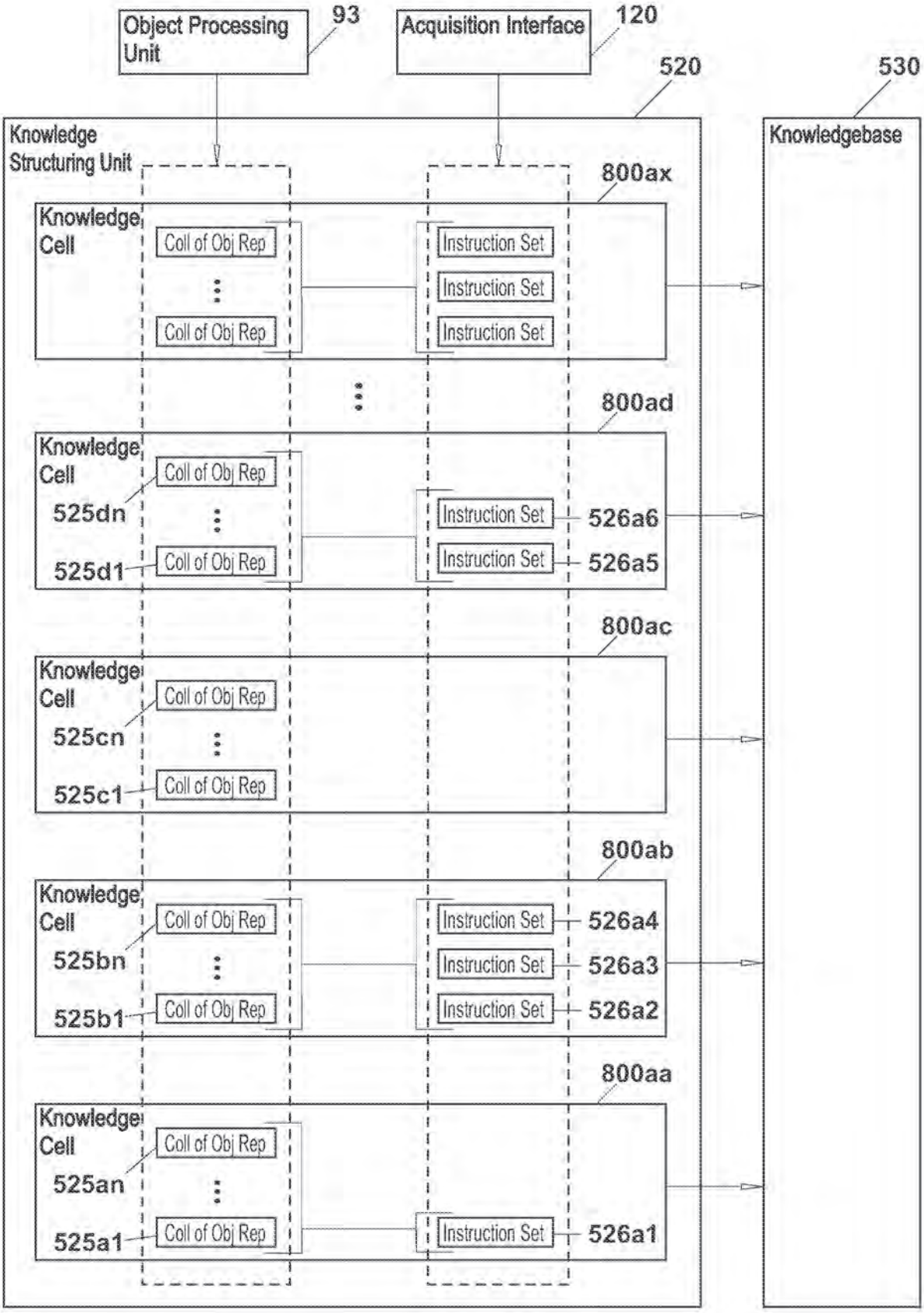

Object Processing Unit
93
Acquisition Interface
120
520
530
Knowledge Structuring Unit
Knowledgebase
800ax
Knowledge Cell
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
800ad
Knowledge Cell
525dn
525d1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
526a6
526a5
800ac
Knowledge Cell
525cn
525c1
Coll of Obj Rep
Coll of Obj Rep
800ab
Knowledge Cell
525bn
525b1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
Instruction Set
Instruction Set
526a4
526a3
526a2
800aa
Knowledge Cell
525an
525a1
Coll of Obj Rep
Coll of Obj Rep
Instruction Set
526a1

FIG. 16

FIG. 17

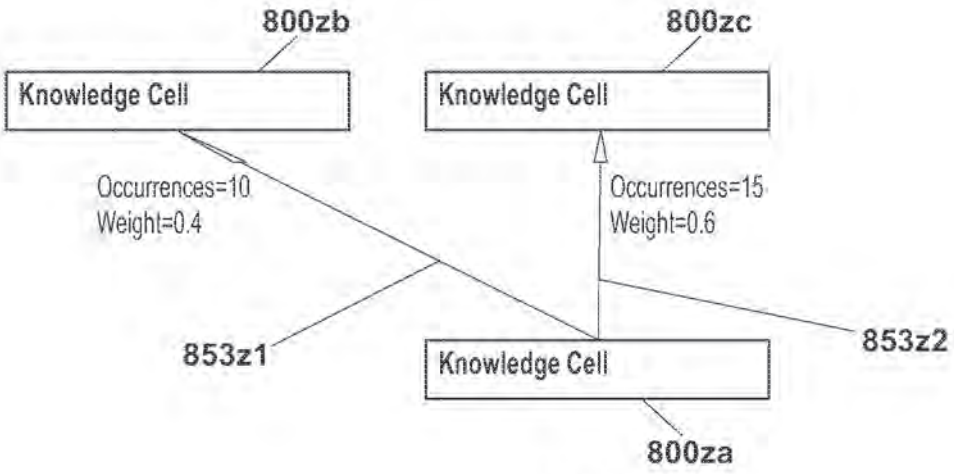


FIG. 18A

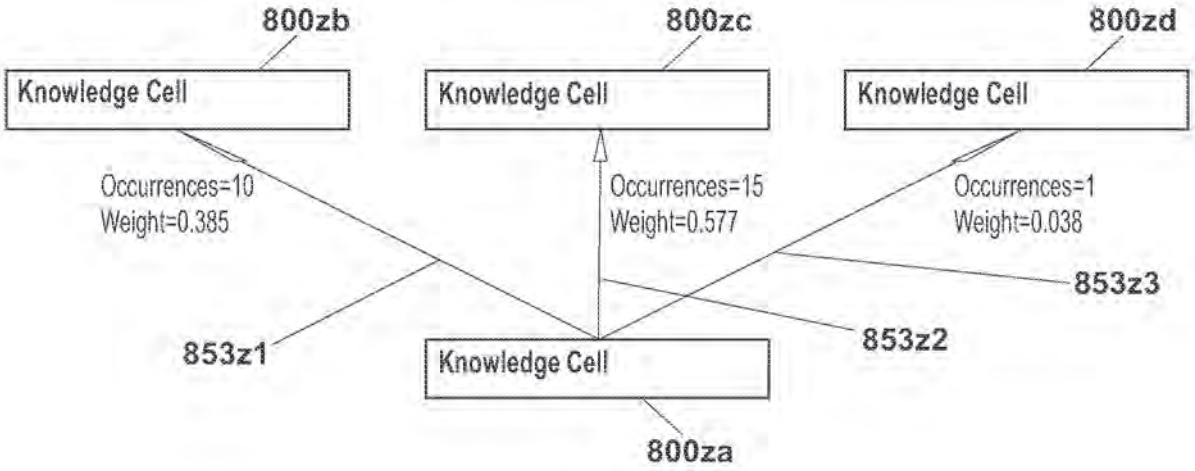


FIG. 18B

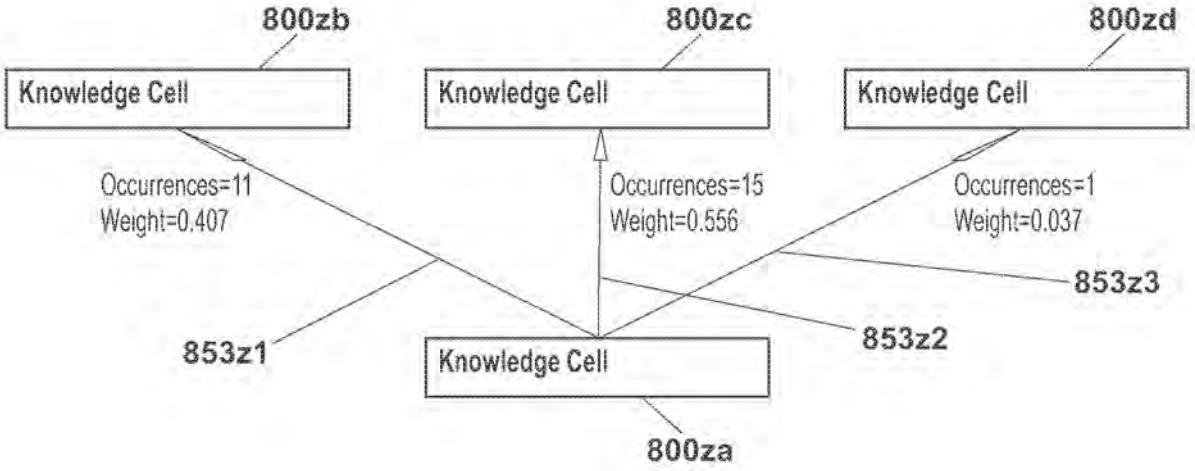


FIG. 18C

Knowledge Structuring Unit
520
Knowledge Cell
800be
800bd
800bc
800bb
800ba
Collection of Knowledge Cells
530d
800ca
800cb
800cc
800cd
800ce

FIG. 19

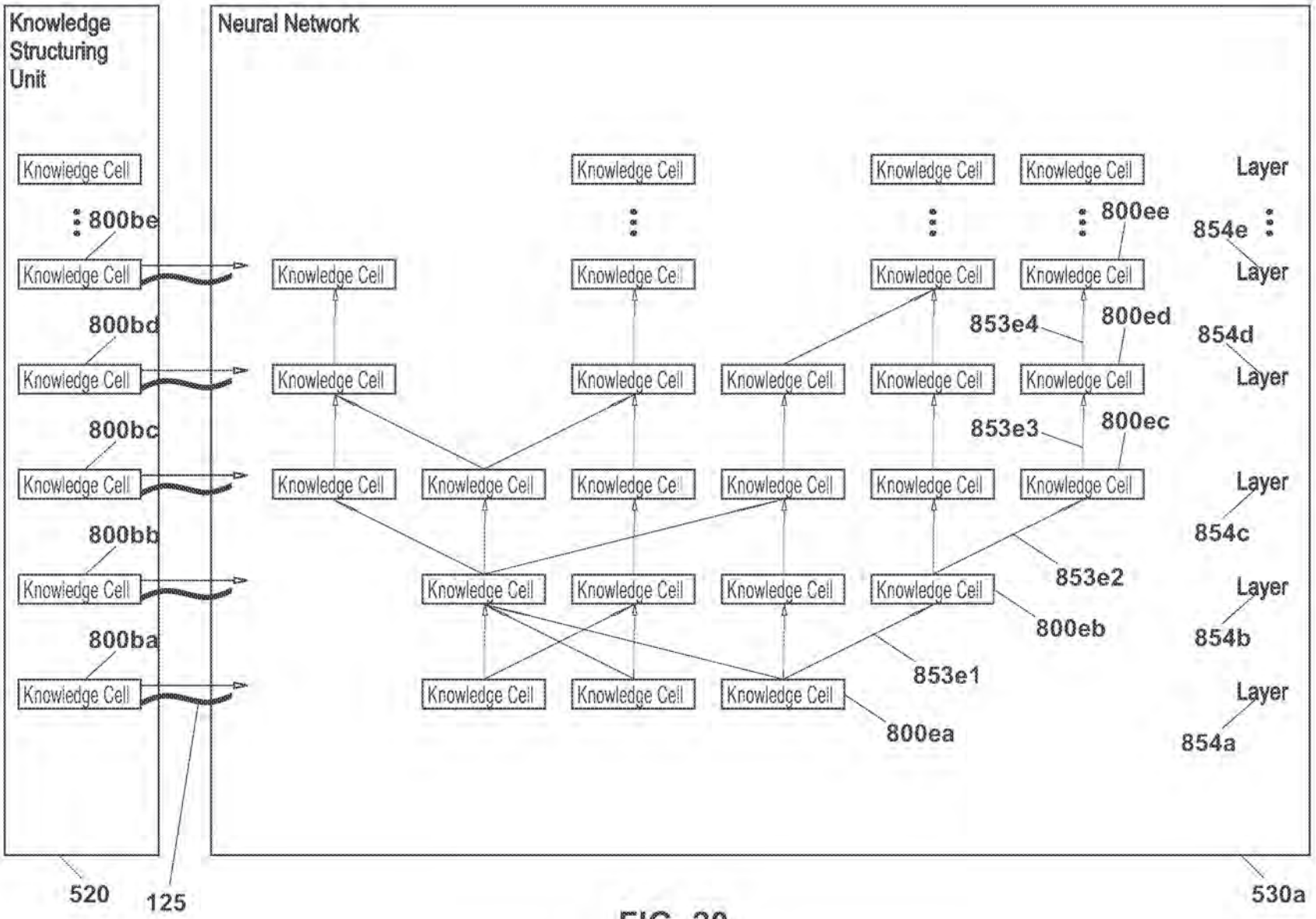


FIG. 20

Autonomous Ex. 2004 p. 484 of 510
Tesla v. Autonomous
IPR2024-00745

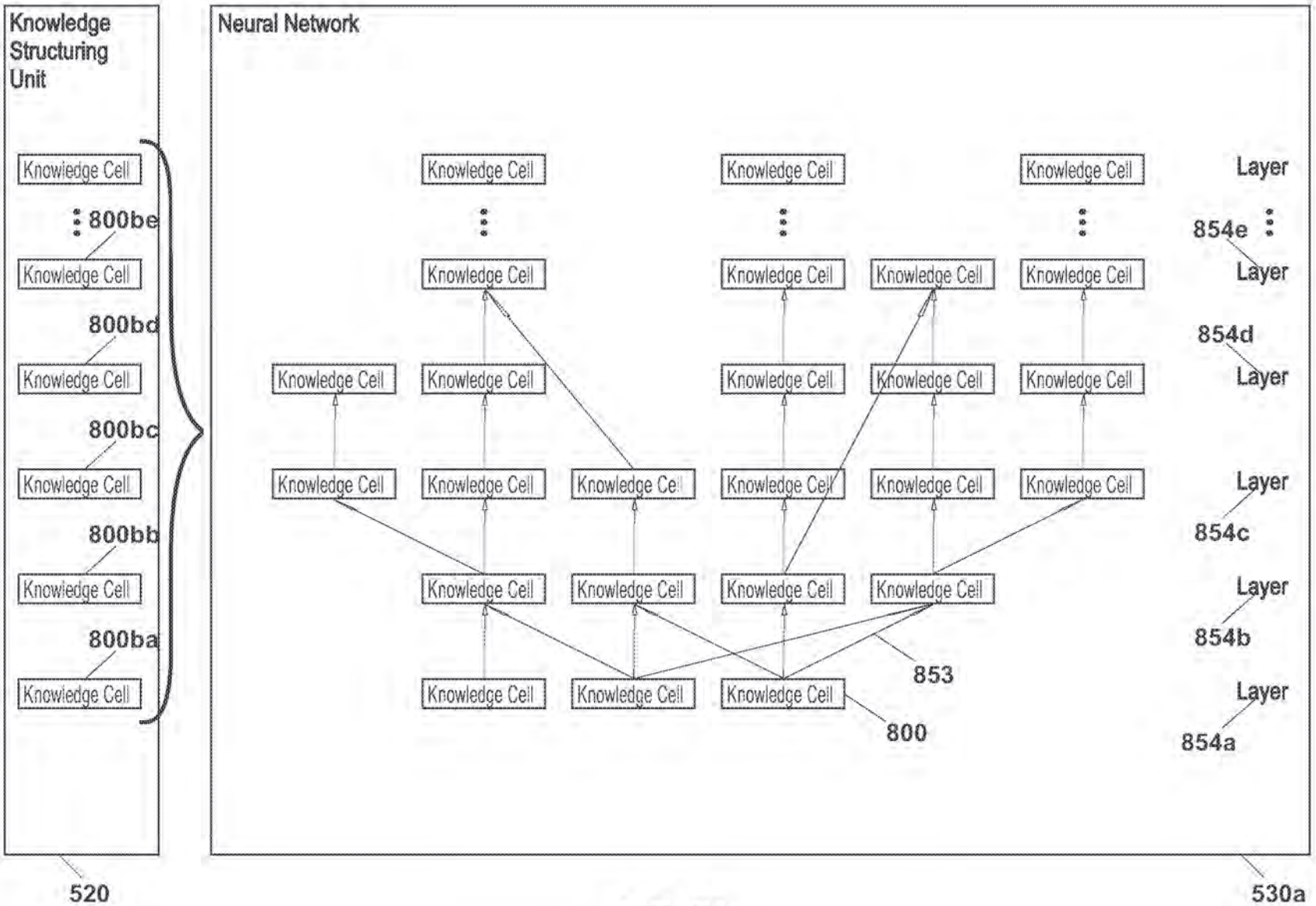


FIG. 21

Autonomous Ex. 2004 p. 485 of 510
Tesla v. Autonomous
IPR2024-00745

Knowledge Structuring Unit
520
Knowledge Cell
800be
800bd
800bc
800bb
800ba
Graph
530b
800he
800hd
853h4
853h3
800hc
853h1
853h2
800ha
800hb

FIG. 22

FIG. 23

Autonomous Ex. 2004 p. 487 of 510
Tesla v. Autonomous
IPR2024-00745

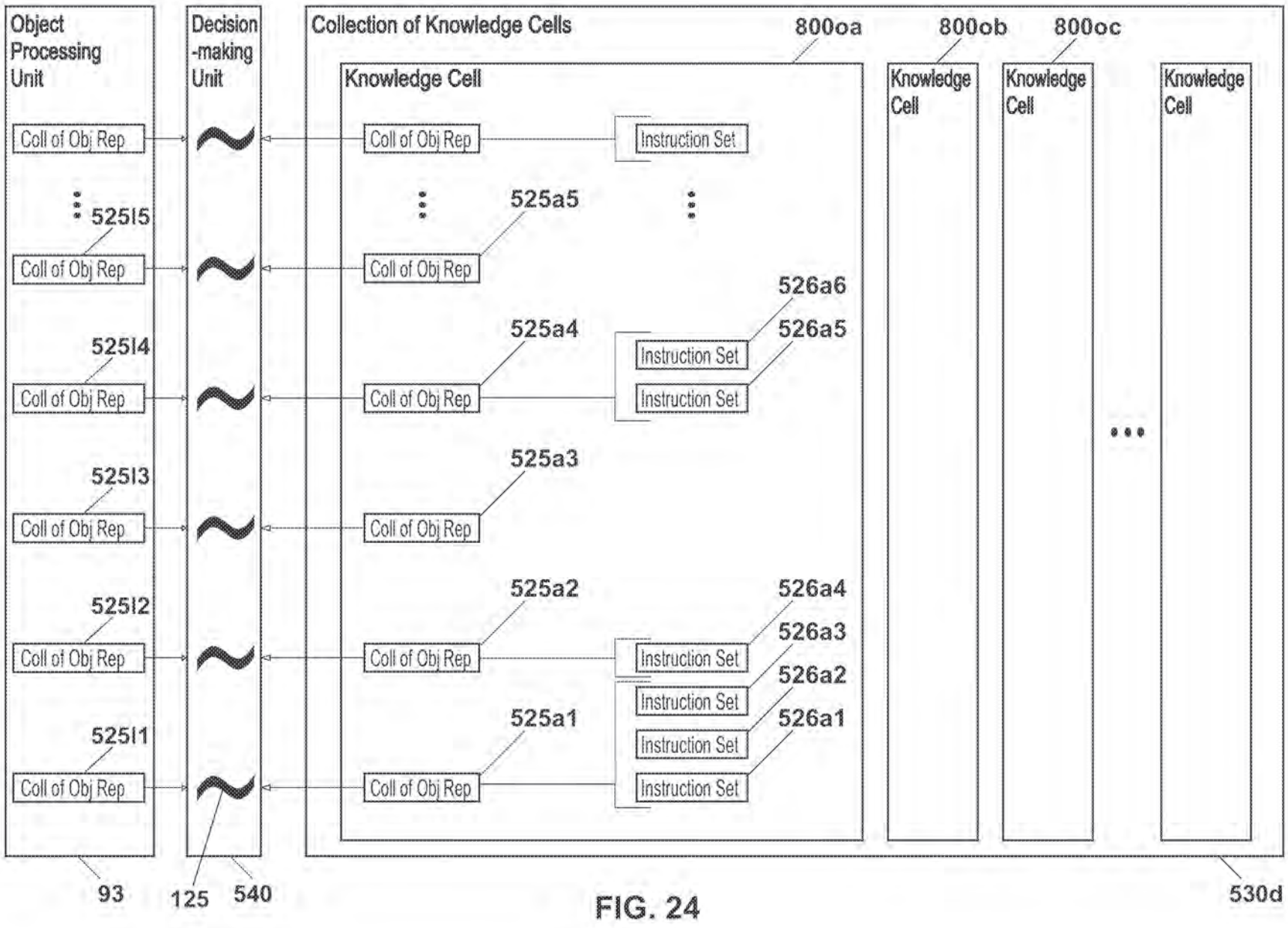


FIG. 24

Autonomous Ex. 2004 p. 488 of 510
Tesla v. Autonomous
IPR2024-00745

Object Processing Unit
93
Coll of Obj Rep
525l3
525l2
525l1
Decision -making Unit
540
125
Collection of Knowledge Cells
530d
800oa
800ob
800oc
Knowledge Cell
Instruction Set
525a5
525a4
525a3
525a2
525a1
526a6
526a5
526a4
526a3
526a2
526a1

FIG. 25

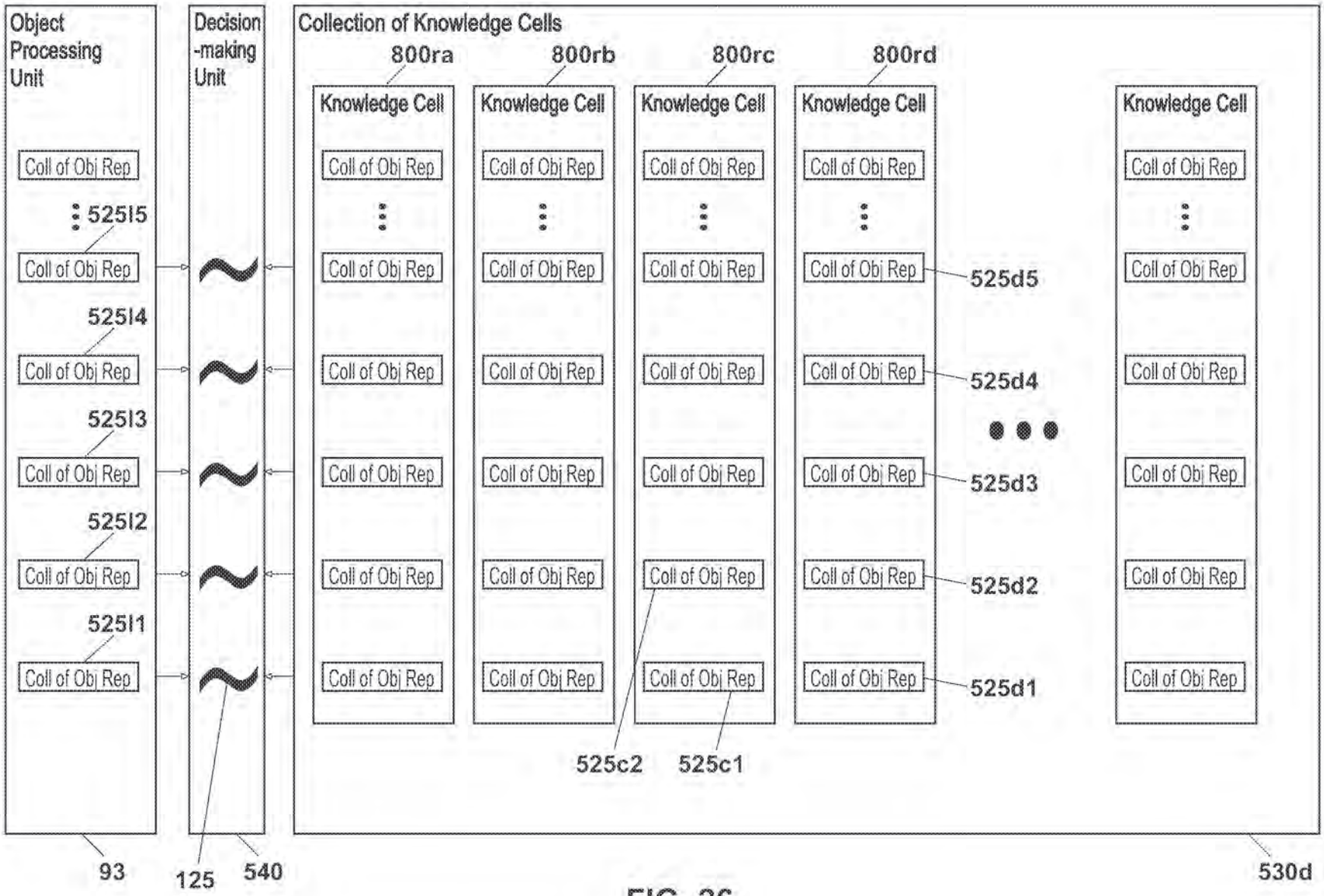


FIG. 26

Autonomous Ex. 2004 p. 490 of 510
Tesla v. Autonomous
IPR2024-00745

Object Processing Unit
Decision -making Unit
Neural Network
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
Knowledge Cell
Layer
854e
854d
854c
854b
854a
800te
800td
800tc
800tb
800ta
853t4
853t3
853t2
853t1
0.75
0.25
0.80
0.20
1
0.35
0.30
93
125
540
530a

FIG. 27

Object Processing Unit
Coll of Obj Rep
Coll of Obj Rep
Coll of Obj Rep 525en
Coll of Obj Rep 525e1
Coll of Obj Rep 525dn
Coll of Obj Rep 525d1
Coll of Obj Rep 525cn
Coll of Obj Rep 525c1
Coll of Obj Rep 525bn
Coll of Obj Rep 525b1
Coll of Obj Rep 525an
Coll of Obj Rep 525a1
93
Decision -making Unit
125
540
Graph
530b
Knowledge Cell
800ua
800ub
800uc
800ud
800ue
853u1
853u2
853u3
853u4
0.15
0.85
0.60
0.30
0.10
0.20
0.80
1

FIG. 28

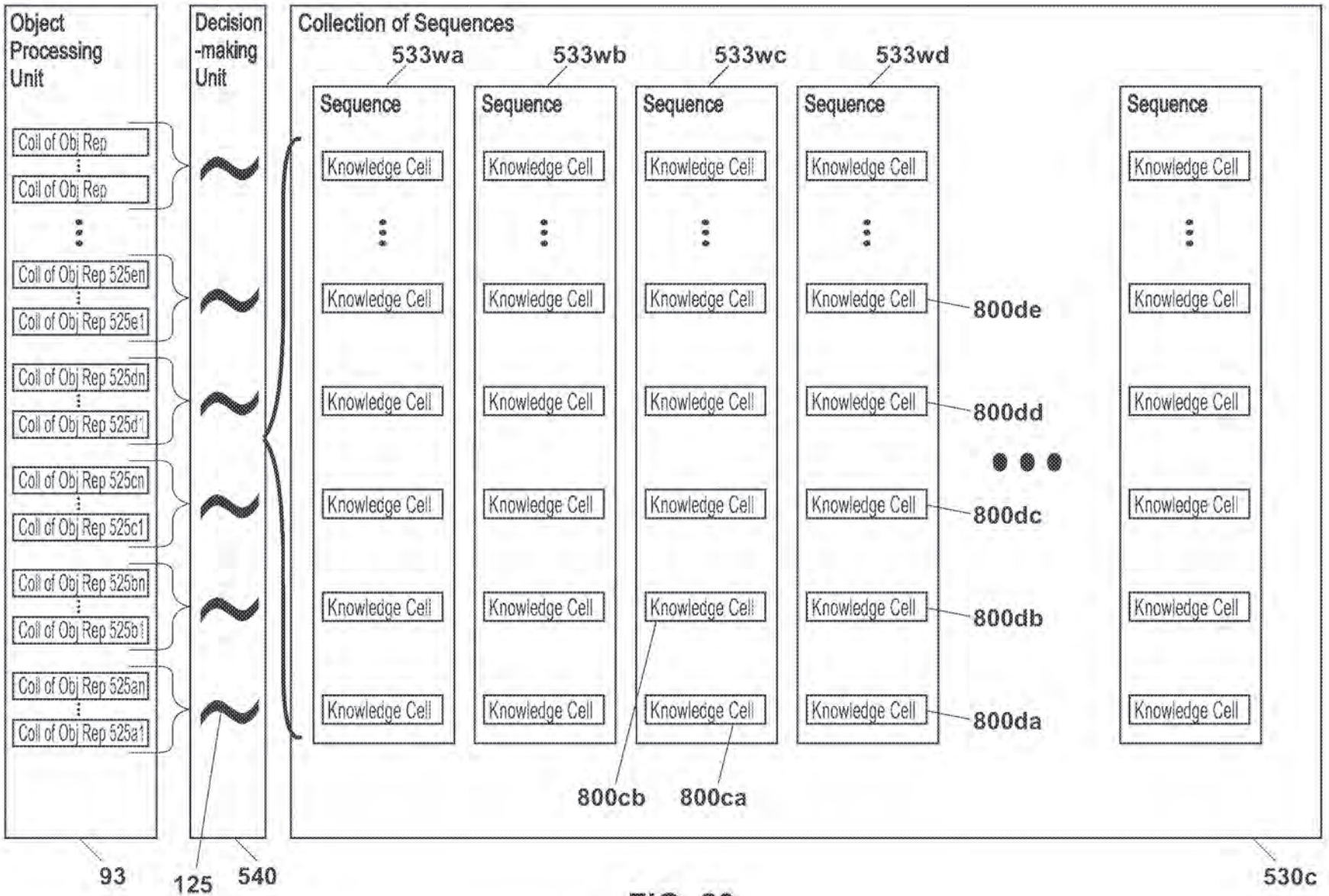


FIG. 29

Autonomous Ex. 2004 p. 493 of 510
Tesla v. Autonomous
IPR2024-00745

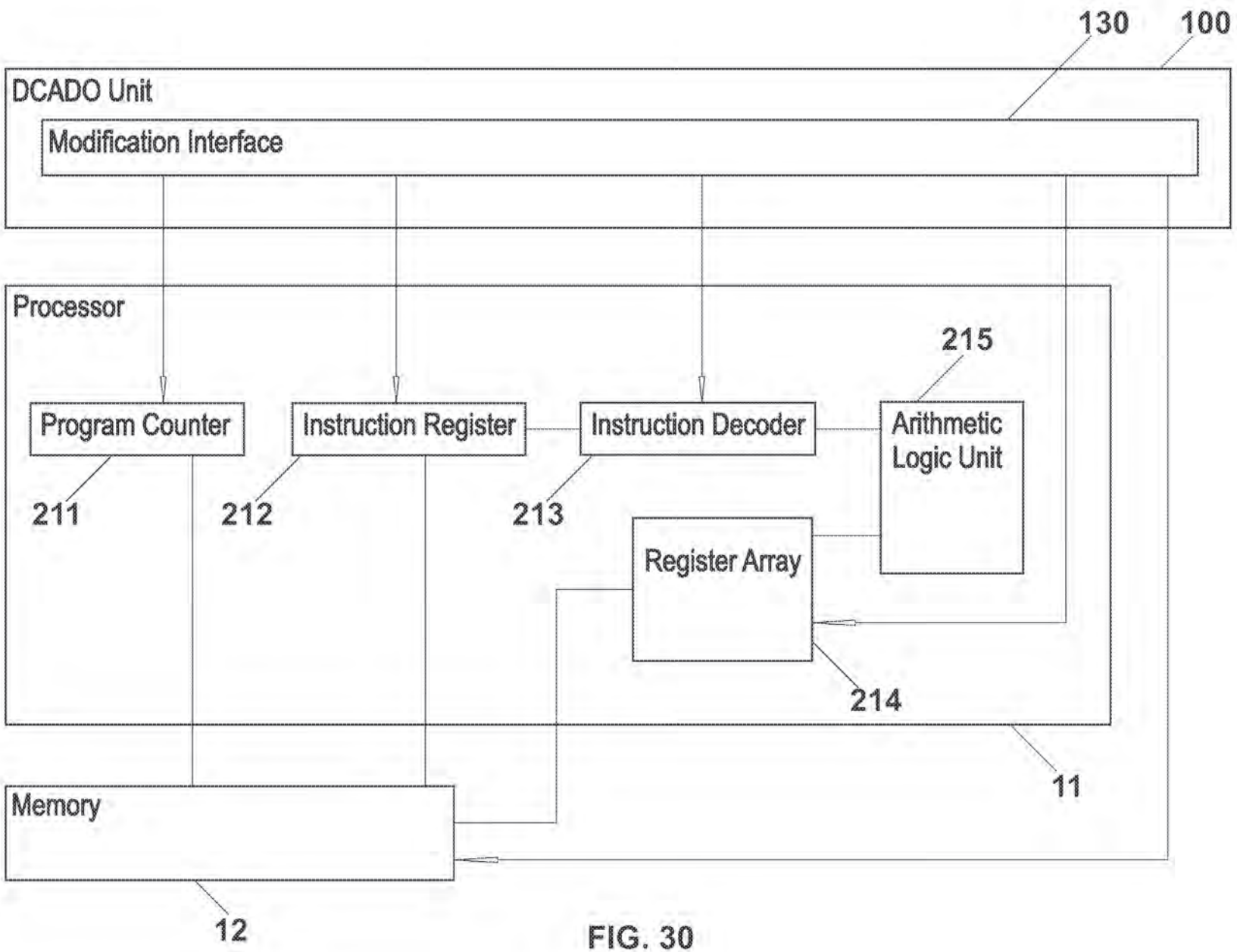


FIG. 30

Autonomous Ex. 2004 p. 494 of 510
Tesla v. Autonomous
IPR2024-00745

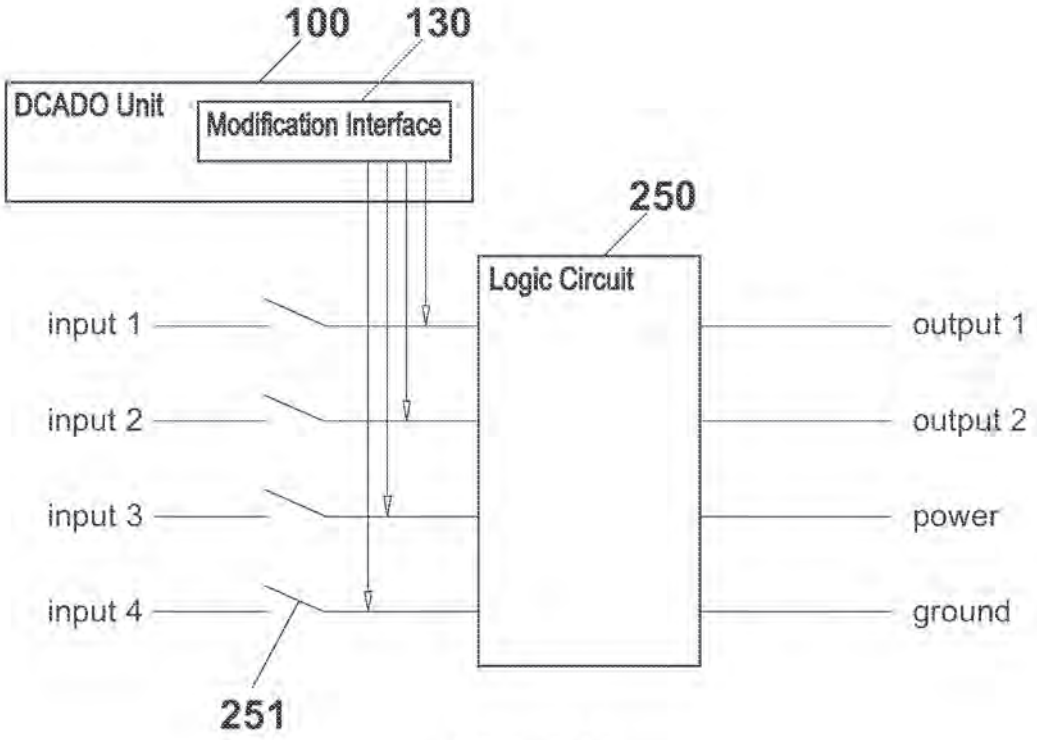


FIG. 31A

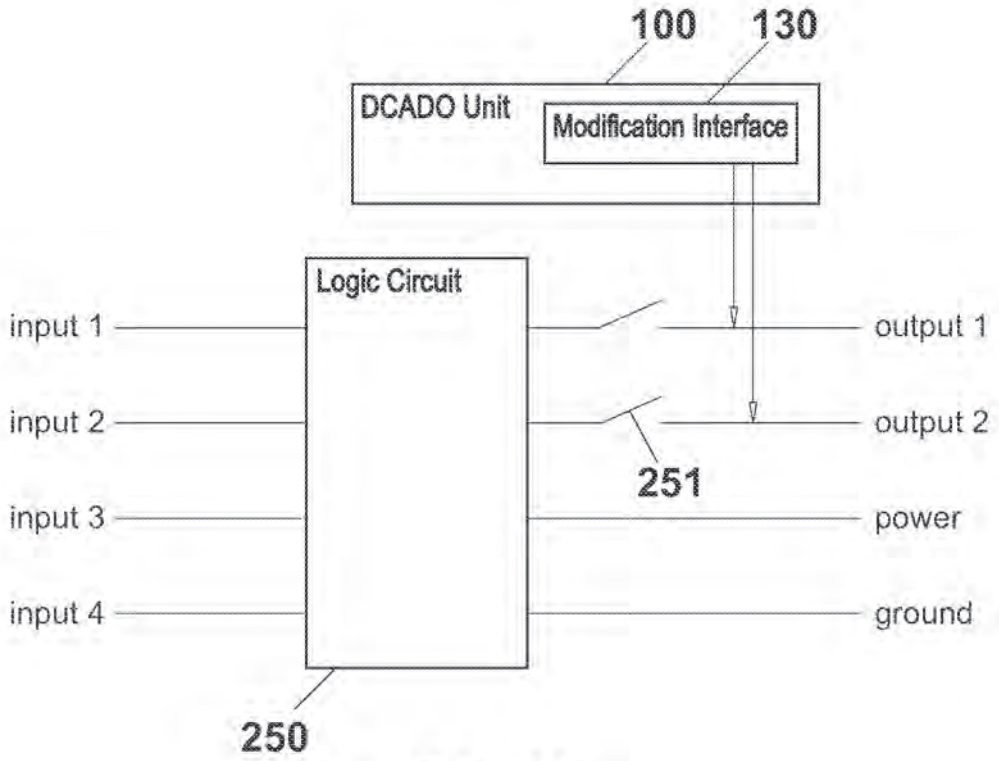


FIG. 31B

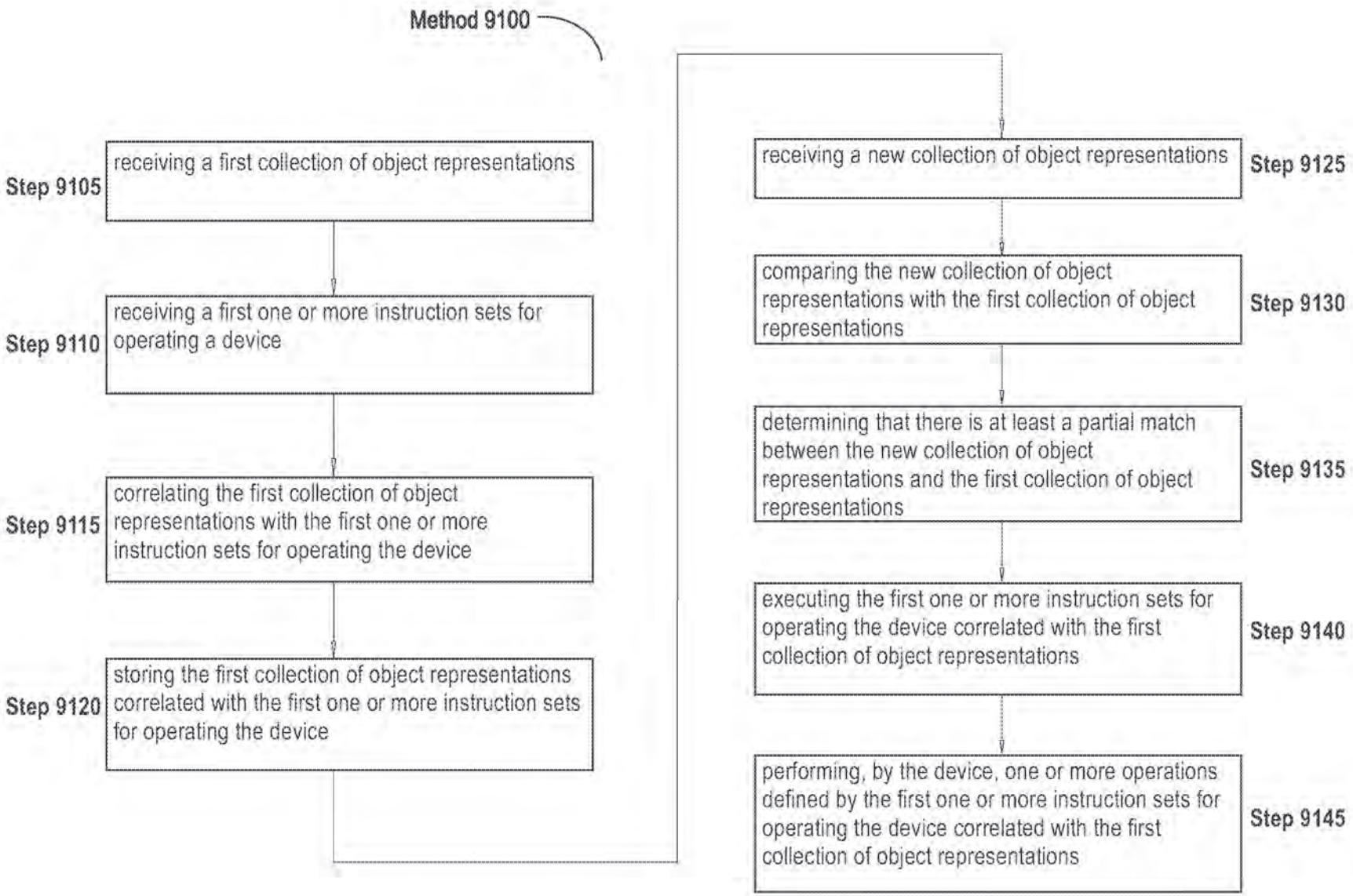


FIG. 32

Autonomous Ex. 2004 p. 496 of 510
Tesla v. Autonomous
IPR2024-00745

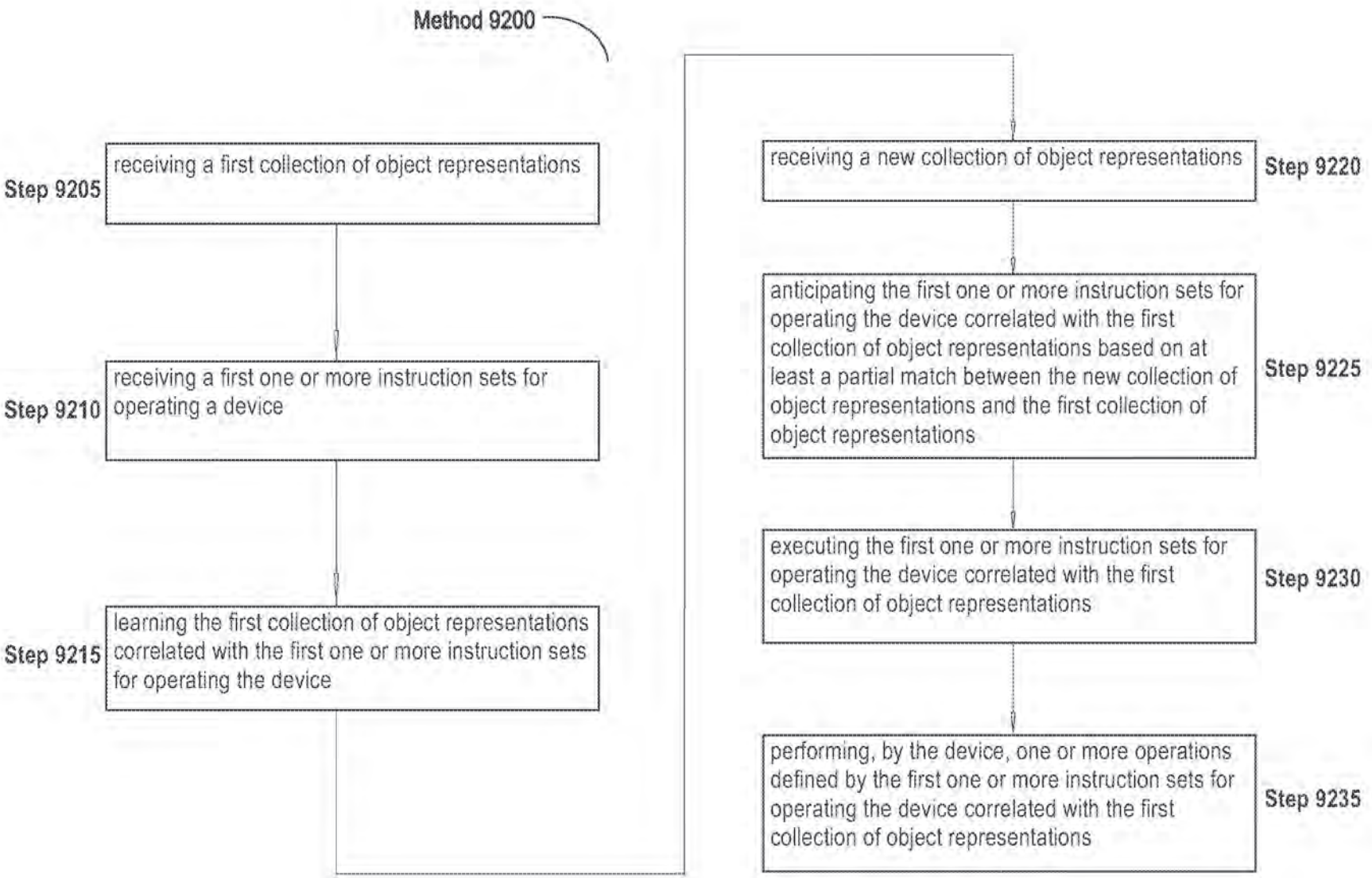


FIG. 33

Autonomous Ex. 2004 p. 497 of 510
Tesla v. Autonomous
IPR2024-00745

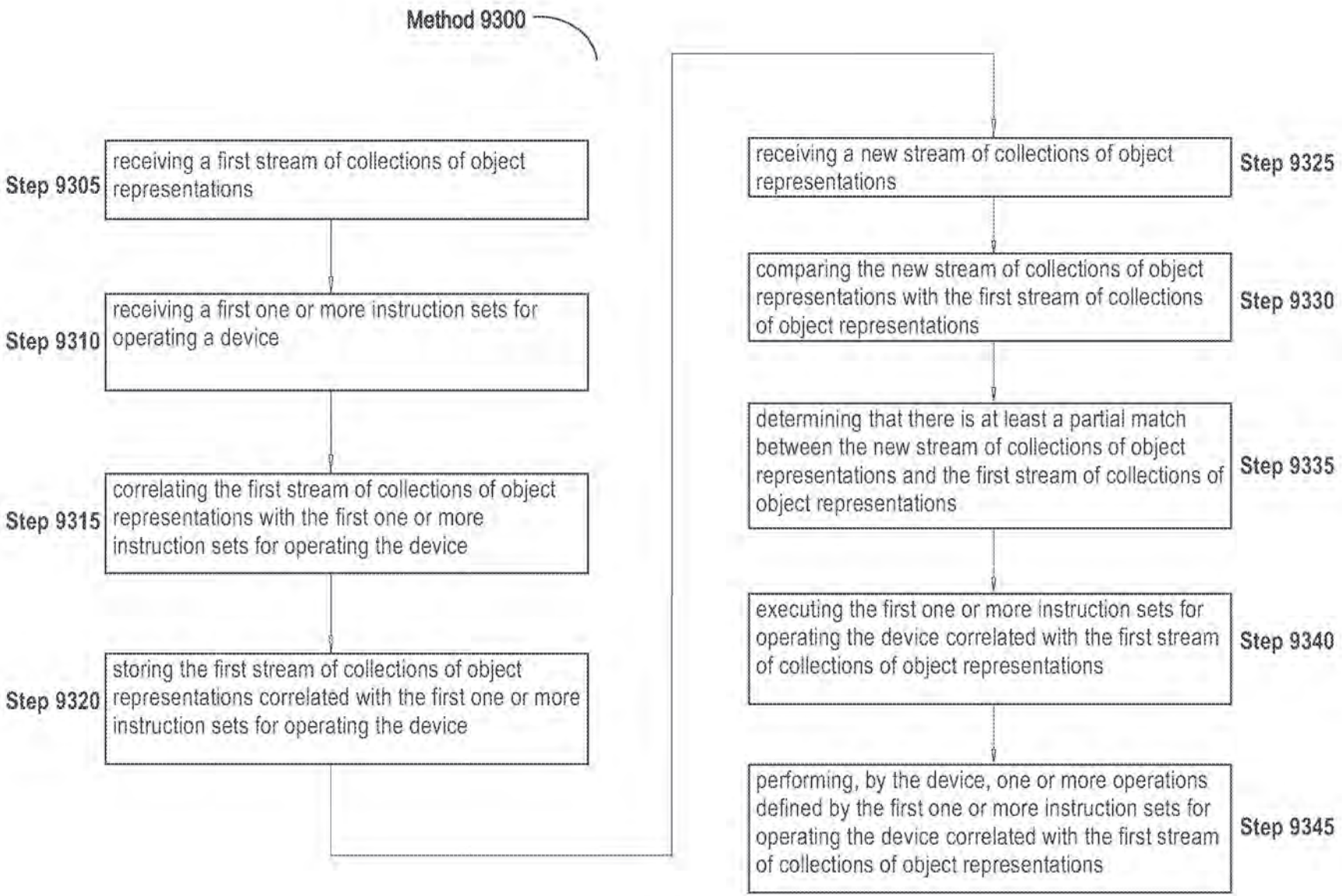


FIG. 34

Autonomous Ex. 2004 p. 498 of 510
Tesla v. Autonomous
IPR2024-00745

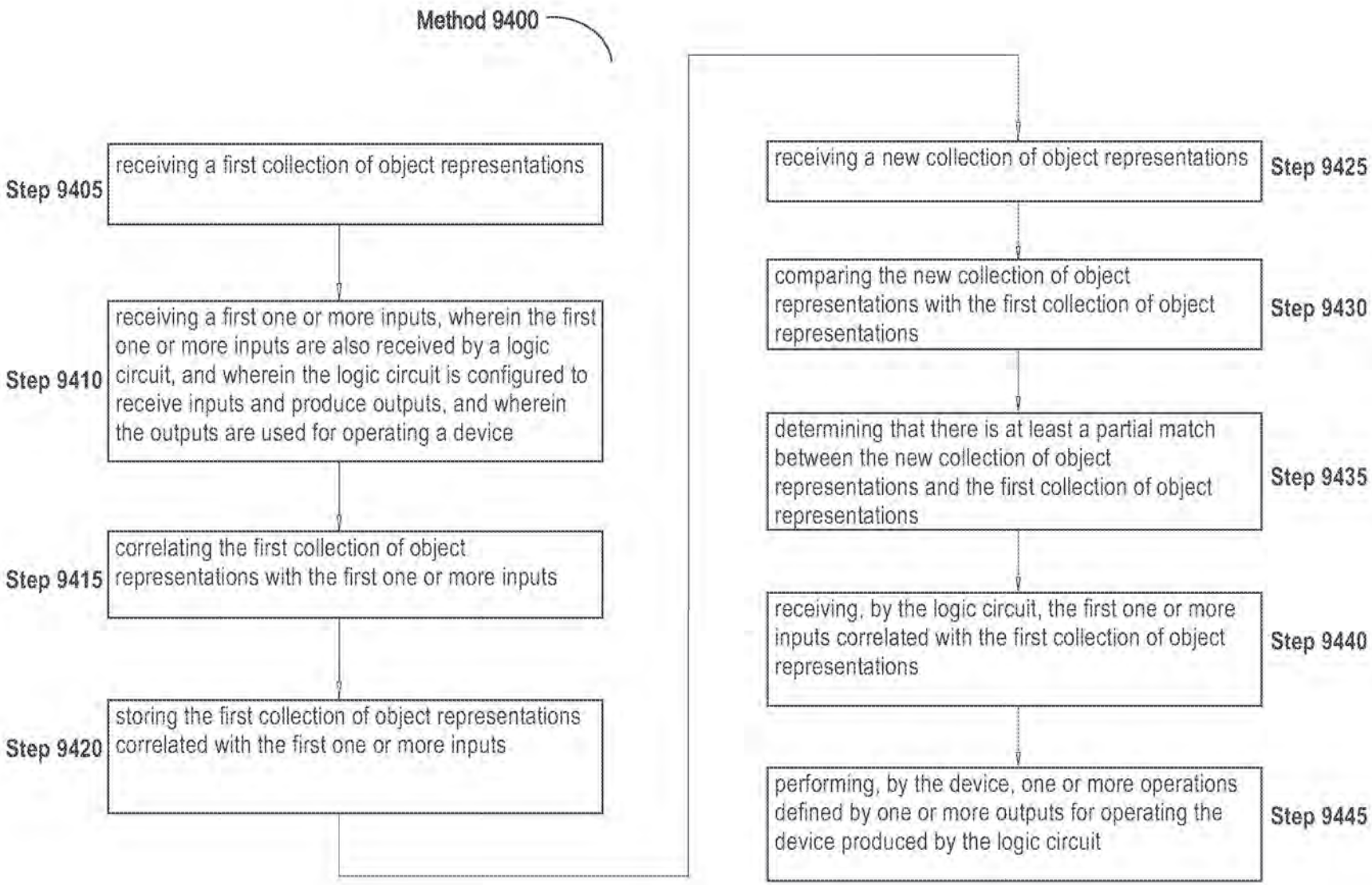


FIG. 35

Autonomous Ex. 2004 p. 499 of 510
Tesla v. Autonomous
IPR2024-00745

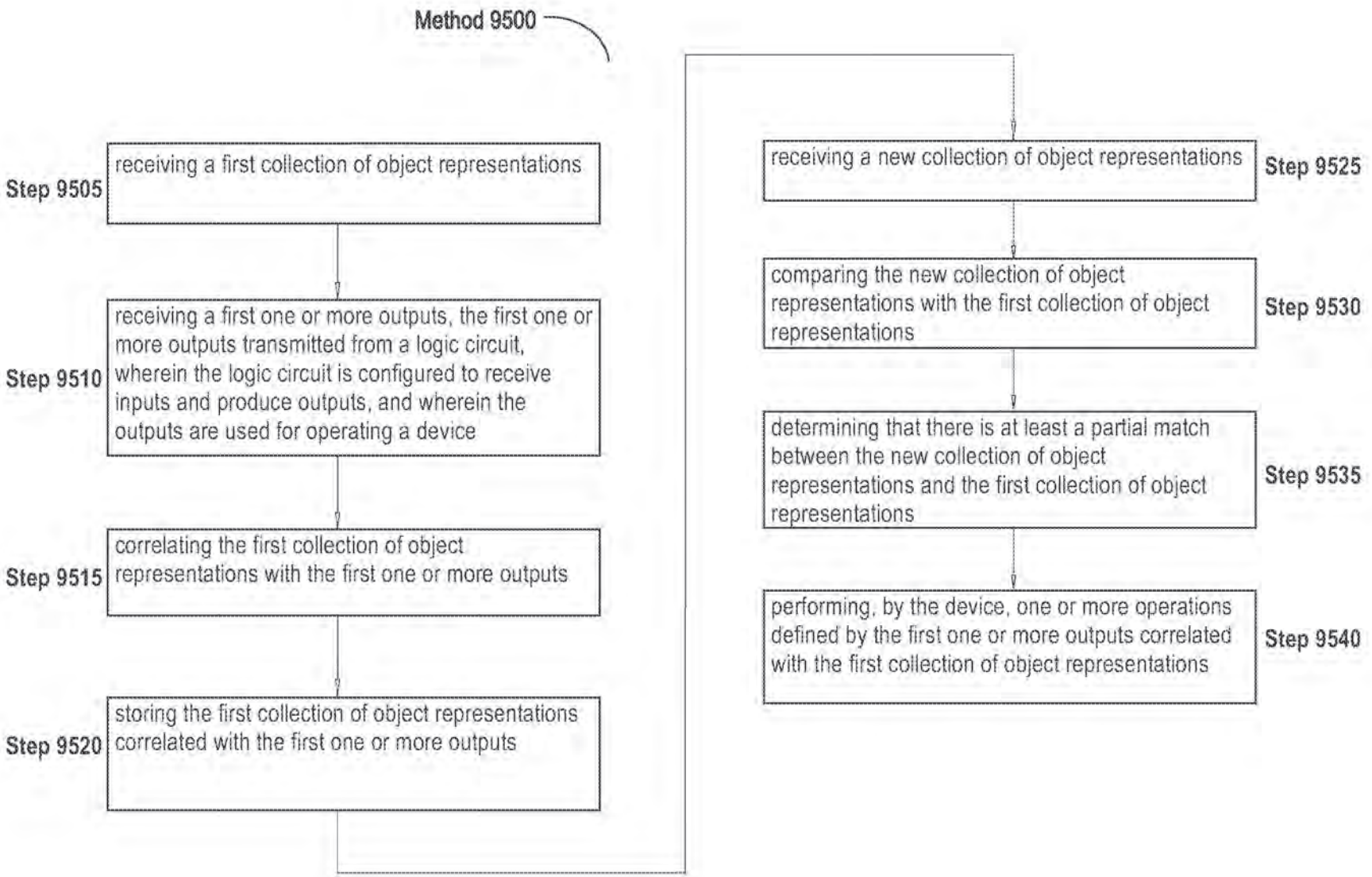


FIG. 36

Autonomous Ex. 2004 p. 500 of 510
Tesla v. Autonomous
IPR2024-00745

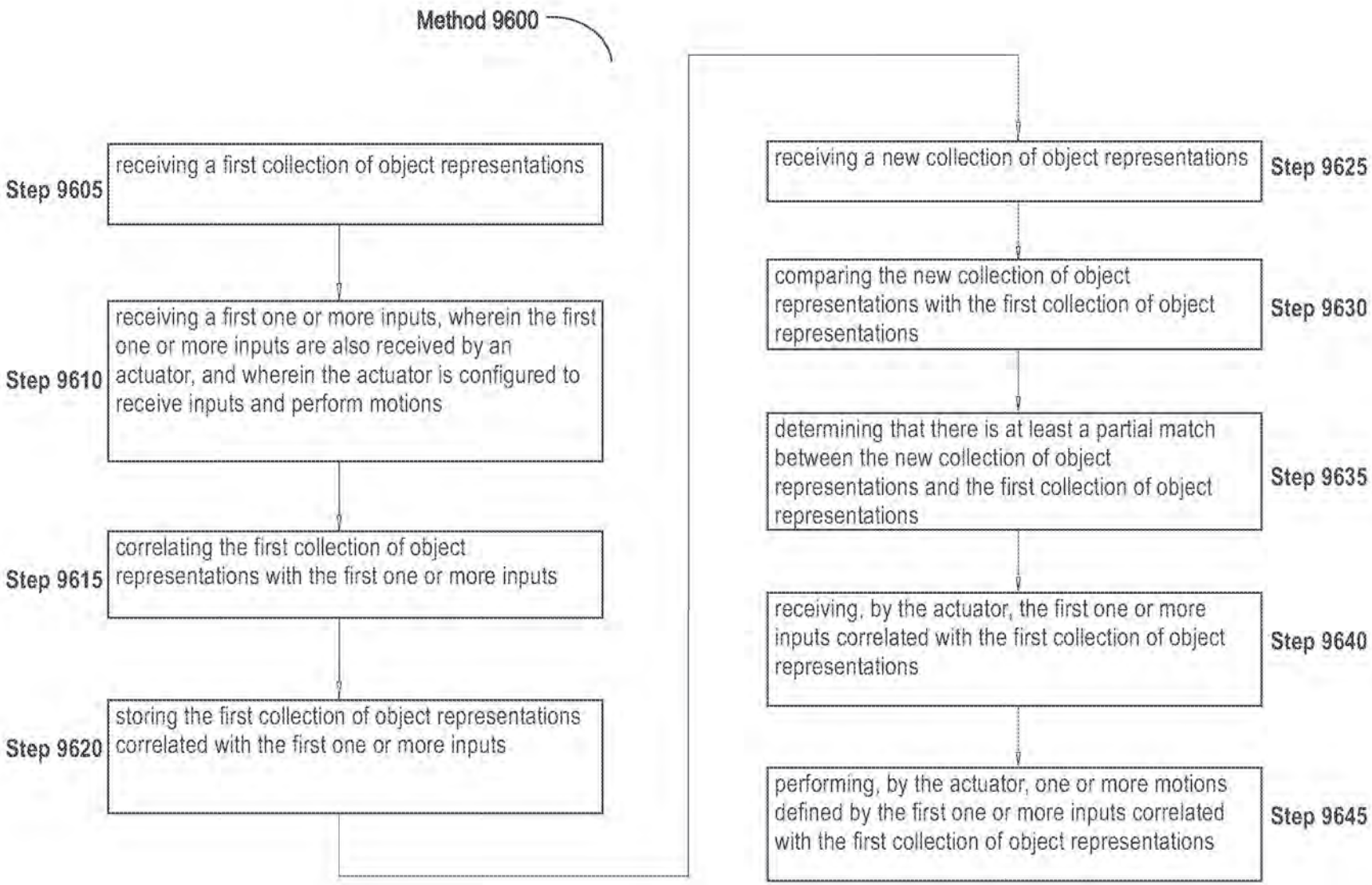


FIG. 37

Autonomous Ex. 2004 p. 501 of 510
Tesla v. Autonomous
IPR2024-00745

615aa
615ab
98a
98a
98a
615ac
615ad
7m
15m
26m
35m
97°
199°
274°
346°

FIG. 38

FIG. 39

615bb
90m
615ba
110m
149°
71°
98b
308°
358°
120m
615bd
450
190m
615bc

FIG. 40

Electronic Patent Application Fee Transmittal

Application Number:	
Filing Date:	
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Filer:	Jasmin Cosic
Attorney Docket Number:	

Filed as Small Entity

Filing Fees for Utility under 35 USC 111(a)

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	70	70
UTILITY SEARCH FEE	2111	1	300	300
UTILITY EXAMINATION FEE	2311	1	360	360
Pages:				
UTILITY APPL SIZE FEE PER 50 SHEETS >100	2081	1	200	200
Claims:				
Miscellaneous-Filing:				
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Miscellaneous:				
			Total in USD ($)	930

Electronic Acknowledgement Receipt

EFS ID:	27391591
Application Number:	15340991
International Application Number:	
Confirmation Number:	1993
Title of Invention:	ARTIFICIALLY INTELLIGENT SYSTEMS, DEVICES, AND METHODS FOR LEARNING AND/OR USING A DEVICE'S CIRCUMSTANCES FOR AUTONOMOUS DEVICE OPERATION
First Named Inventor/Applicant Name:	Jasmin Cosic
Customer Number:	116094
Filer:	Jasmin Cosic
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	02-NOV-2016
Filing Date:	
Time Stamp:	02:17:15
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	CARD
Payment was successfully received in RAM	$930
RAM confirmation Number	110216INTEFSW02222800
Deposit Account	
Authorized User	
The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:	

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Abstract	ABSTRACT_flattened.pdf	220691 daaae9f7d0b2f5ba2cf0ec8b463a23fc7d665c06	no	1
Warnings:					
Information:					
2	Application Data Sheet	ADS.pdf	1822777 9f190540542 3b472d1600a2612af4322cdb5864 38	no	8
Warnings:					
Information:					
3	Oath or Declaration filed	DECLARATION_flattened.pdf	80488 203698553466595c777e59543f418bd69a3e225a	no	1
Warnings:					
Information:					
4	Specification	SPECIFICATION_flatenned.pdf	11824929 7169a9698cb43429c7efec2330d9b2154bc58743	no	152
Warnings:					
Information:					
5	Claims	CLAIMS_flattened.pdf	2974744 2fd110cc9195f6641b83a783d8a865851cd651e5	no	8
Warnings:					
Information:					
6	Drawings-only black and white line drawings	DRAWINGS_flattened.pdf	1137891 0551a2e1e7754760edd6da3a1041bf9c4e8edd7e	no	40
Warnings:					

The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing

Information:					
7	Fee Worksheet (SB06)	fee-info.pdf	36505 c5001e6f77638214a3677f56d2df6907a863a87	no	2
Warnings:					
Information:					
			Total Files Size (in bytes):	18098025	

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

DocCode - SCORE

SCORE Placeholder Sheet for IFW Content

Application Number: 15340991 Document Date: 11/02/2016

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

- Drawings – Other than Black and White Line Drawings

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

To access the documents in the SCORE database, refer to instructions below.

At the time of document entry (noted above):

- Examiners may access SCORE content via the eDAN interface.
- Other USPTO employees can bookmark the current SCORE URL (http://Score.uspto.gov/ScoreAccessWeb/).
- External customers may access SCORE content via the Public and Private PAIR interfaces.

Form Revision Date: September 30, 2013