

TUTORIALS / UNDERSTANDING ROBOT COORDINATE FRAMES AND POINTS

BEGINNER

Understanding Robot Coordinate Frames and Points

INDUSTRIAL ROBOTICS FANUC KUKA



By Niall Henry

Introduction

One of the most fundamental bases of a robotic system is motion and cartesian positioning in 3D space. To correctly program your robot, you need to understand how a robot interprets the space surrounding it.

This tutorial will cover the basics of robot positions, how you define them in a robotic system, and the standard robot coordinate frames. By the end of this tutorial, you will have a basic understanding of the skills you will need as a Robotics engineer.

Understanding Robot Positions

A Frame defines a coordinate system in a robot. It determines where it is in space and the locations of relevant objects or areas around it relative to the robot.

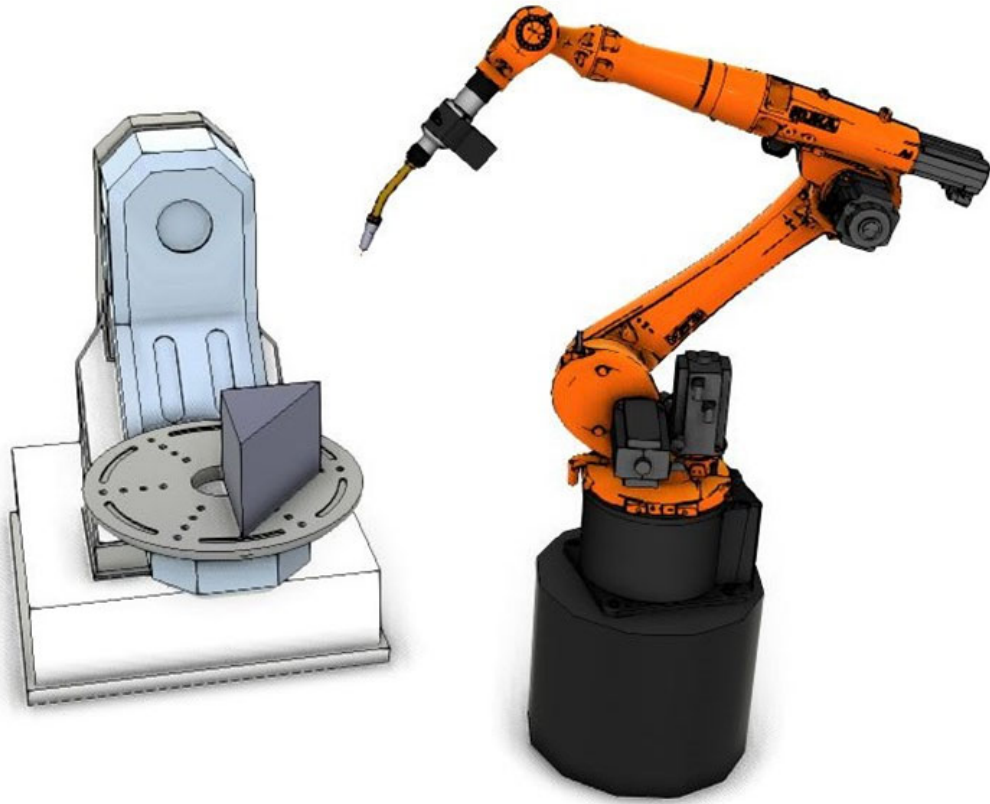


Figure 1 Robot System

In "Figure 1", we see a robot equipped with a welding tool, a workpiece positioner, and a workpiece. For example, suppose we wanted to operate on this piece manually. We would intuitively know how to position ourselves and the tool using our hands to perform the required operation on the correct part of the workpiece.

Although this is a simple task, a lot of processing is happening in our brains to complete it. For example, taking in the environment using our vision, processing that stimulus to gauge distance and location, moving over to the workpiece from our current position, repositioning and orientating the tool and ourselves for operating, and then performing the operation itself.

A robot needs to go through a similar process. For that to happen, the information needs to be formatted so the robot can understand it. We do this using coordinate frames defined by a set of 3 vectors, each of unit length, forming a right angle. We describe these vectors using the standard annotation of X, Y, and Z.

Using these three values, you can express any point in a 3D space. Still, you must always do it relative to a coordinate frame defining the Origin (aka WORLD) frame. It isn't helpful to know that the workpiece corner point is at $(-100, 2.8, 560)$ unless you know where $(0,0,0)$ is and in which direction the three axes are pointing.

Understanding Robot Orientation

In addition to positions, the orientation of a point in space also needs to be defined. There are several approaches for specifying the robot's orientation in 3D. The most widely used and intuitive one is Euler angles. We express Euler angles as three rotations around the three coordinate axes commonly referred to using the notation of A, B, and C or as Roll, Pitch, and Yaw. A rotation is positive for a counterclockwise rotation when viewed by an observer looking down the axis toward the origin.

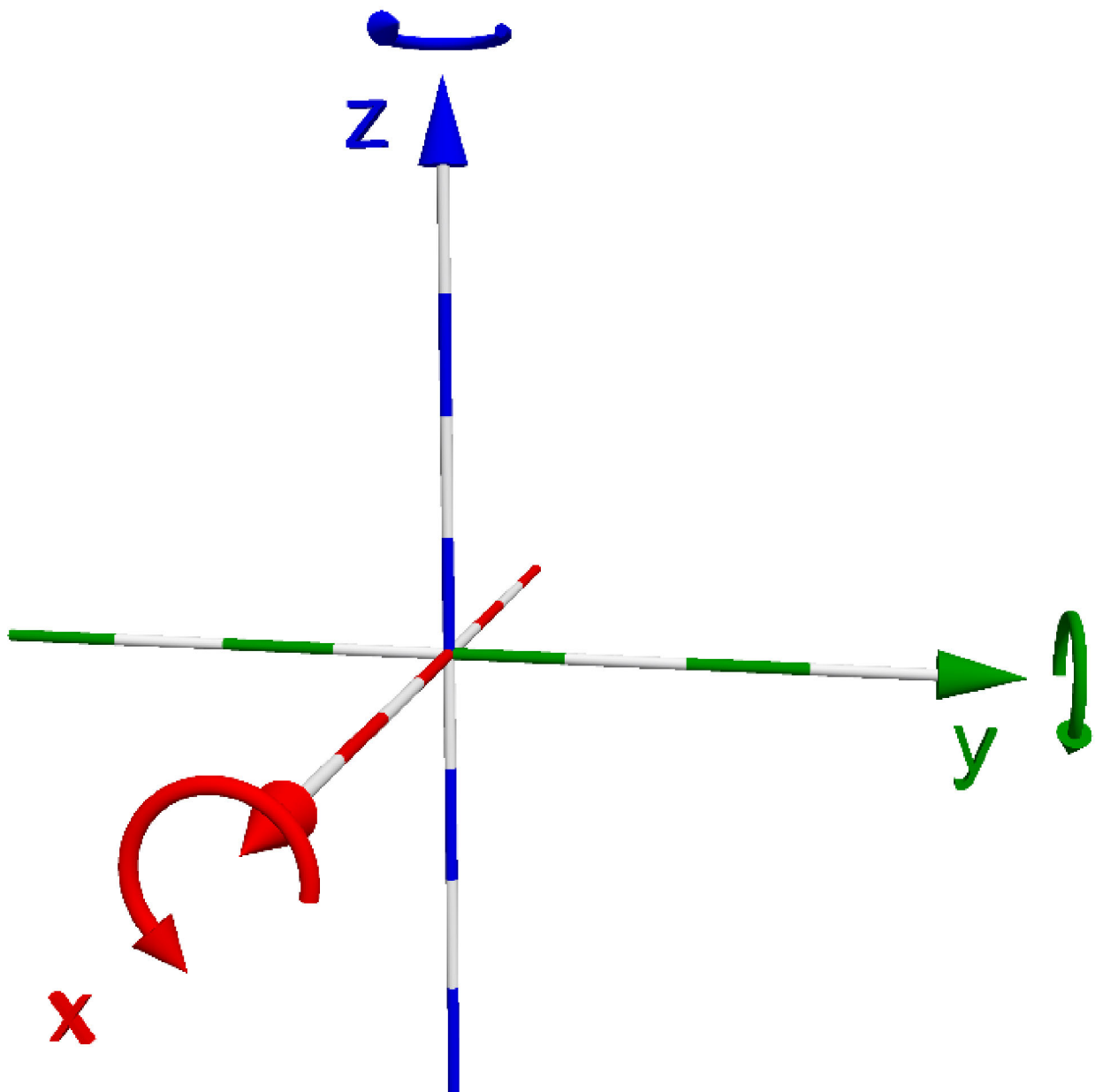


Figure 2 Coordinate System with rotation

The sequence that the A, B, and C rotations follow is essential. ABB robots use the mobile ZYX Euler convention, while FANUC and Kuka use the fixed XYZ Euler angle convention. Stäubli and Kawasaki use the ZYZ convention.

Further details around these conventions won't get covered in this tutorial, but the main idea to understand is that by using the six indicators we learned about (X, Y, Z, A, B, C), you can now define the location and orientation of an object in space.

[Note] When dealing with multi-axis systems, you will need additional indicators to define the location and orientation of a robot since there are multiple ways for the robot to reach a specific point.

Understanding Robot Frames

There are three general coordinate frames used in robotics: the WORLD frame, the USER (Base) frame, and the TOOL frame.

World Frame

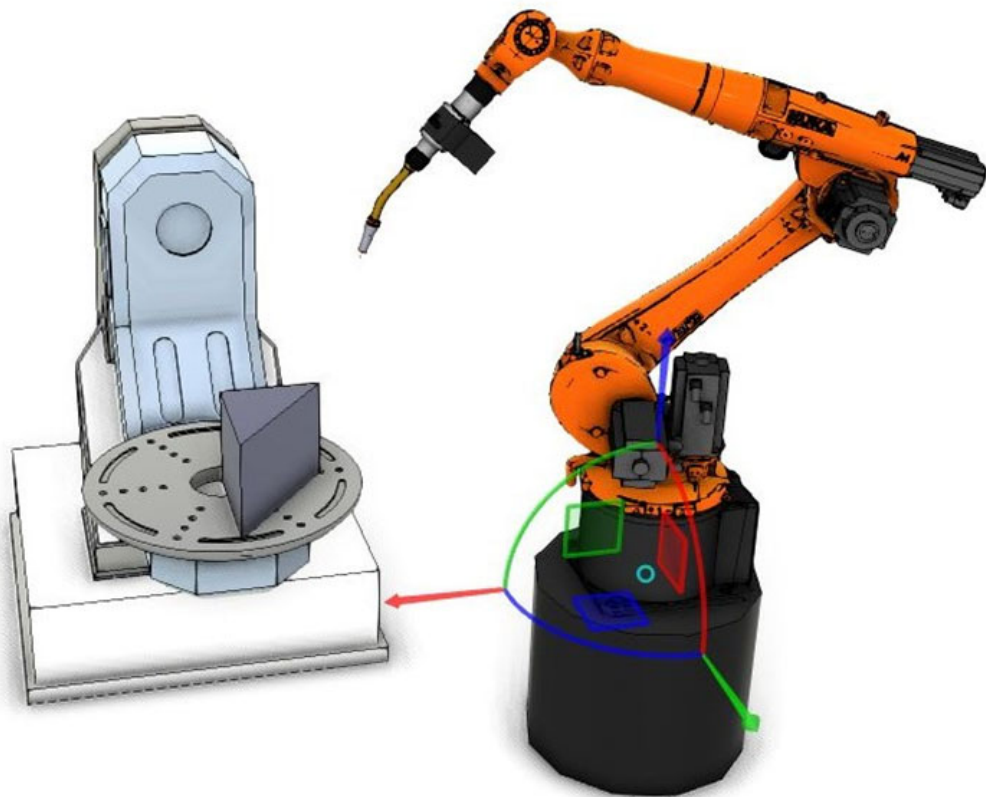


Figure 3 WORLD Frame

The WORLD frame is a fixed cartesian coordinate frame that represents the center point at the robot's base and defines the overall world for the robot. It specifies the three directions in which the axes are all pointing and the origin position. For example, in the (X, Y, Z, A, B, C) format, this would be (0, 0, 0, 0, 0, 0). The WORLD frame is normally defined as the center of the robot base, as shown in "Figure 3".

USER(Base) frames will be defined relative to the WORLD frame, and some or all of the points in space can be directly defined relative to this frame. Using the WORLD frame is a very intuitive way of jogging the robot. Visualizing the expected movement based on its mounting and using the right-hand rule for cartesian coordinate systems makes this easy.

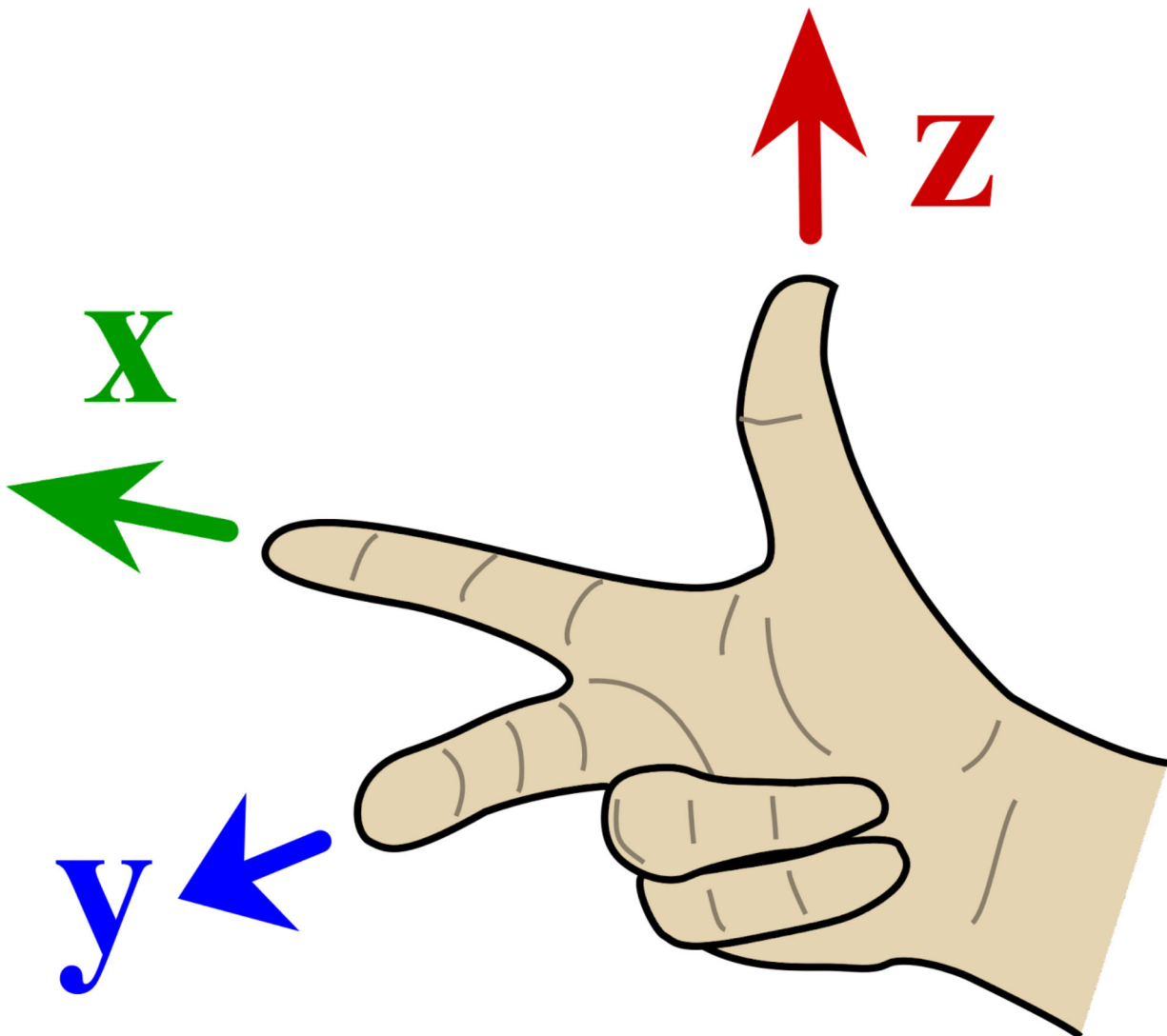


Figure 4 Right-Hand Rule

In a floor-mounted robot's standard configuration, anyone can use the right-hand rule to get the WORLD frame's orientation in space. The right-hand rule gets formed by placing your right hand's index finger, middle finger, and thumb at right angles, as shown in "Figure 4". Each of the fingers and thumb represents a different axis, the Index finger: X axis, the Middle finger: Y axis, and the Thumb: Z axis.

Aligning your hand by facing your index finger toward the robot's connection interface will allow you to orientate yourself with the robot's WORLD frame.

The connection interface is the interface for all connections to the robot, such as power, data, motor, pneumatics, etc., and is typically on the robot base. Therefore, you know the axis required to move when jogging the robot in the WORLD frame.

The application can also change the robot's WORLD frame. An example would be when using a ceiling-mounted robot or if the robot is mounted slightly offset in its position but when the direction of operation needs to stay aligned with a specific coordinate axis.

User (Base) Frame

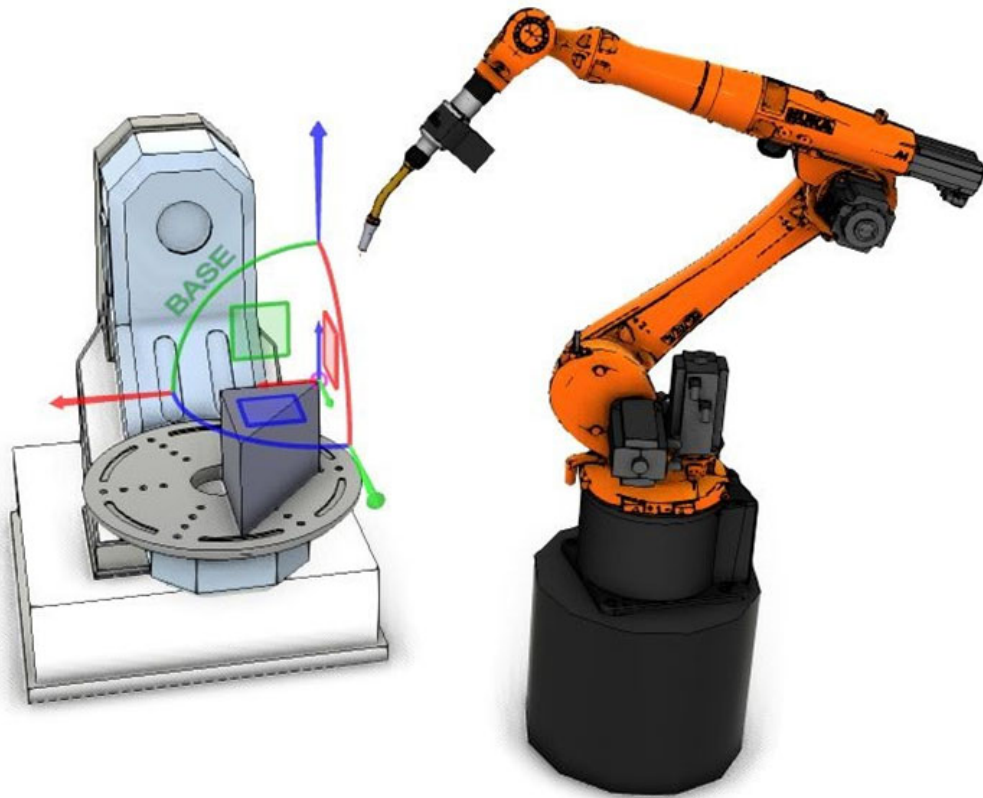


Figure 5 User (Base) Frame

The USER (Base) Frame is a cartesian coordinate system that can define a workpiece or an area of interest, such as a conveyor or a pallet location. This frame is defined relative

to the WORLD frame.

An application can have multiple USER frames depending on the number of operating areas for the robot. This type of frame is beneficial as it allows us to define points relative to a USER frame instead of directly relative to the WORLD frame. That way, if an offset occurs or if the USER frame is something that may move, such as a mobile robot top or positioner, you won't need to adjust every point, just the USER Frame itself.

Imagine a situation where we want to perform a welding application on the workpiece with each triangle corner defined as a point relative to the USER(Base) frame called WORKPIECE[1]. If the positioner's rotation is misaligned or if we placed the workpiece with an offset onto the positioner, all we would need to do is locate a point of relevance. Once the robot has determined the WORKPIECE[1] frame offset, the robot will offset all points on the frame based on this.

In this example, the point of relevance which would be the origin of the workpiece frame relative to the robot could be defined as the closest corner to the robot. If we know that the workpiece might get offset during the concept stage in this application, we will design the robot cell with sensing capabilities to locate this corner. Once found, the robot could continue the welding application as usual, but the offset now has been dealt with without having to modify all points individually.

This frame becomes especially useful when dealing with multiple locations on the USER (Base) frame, such as in a palletizing application where the pallet could come in slightly offset, so the pallet gets defined as a USER (Base) frame.

Tool Frame

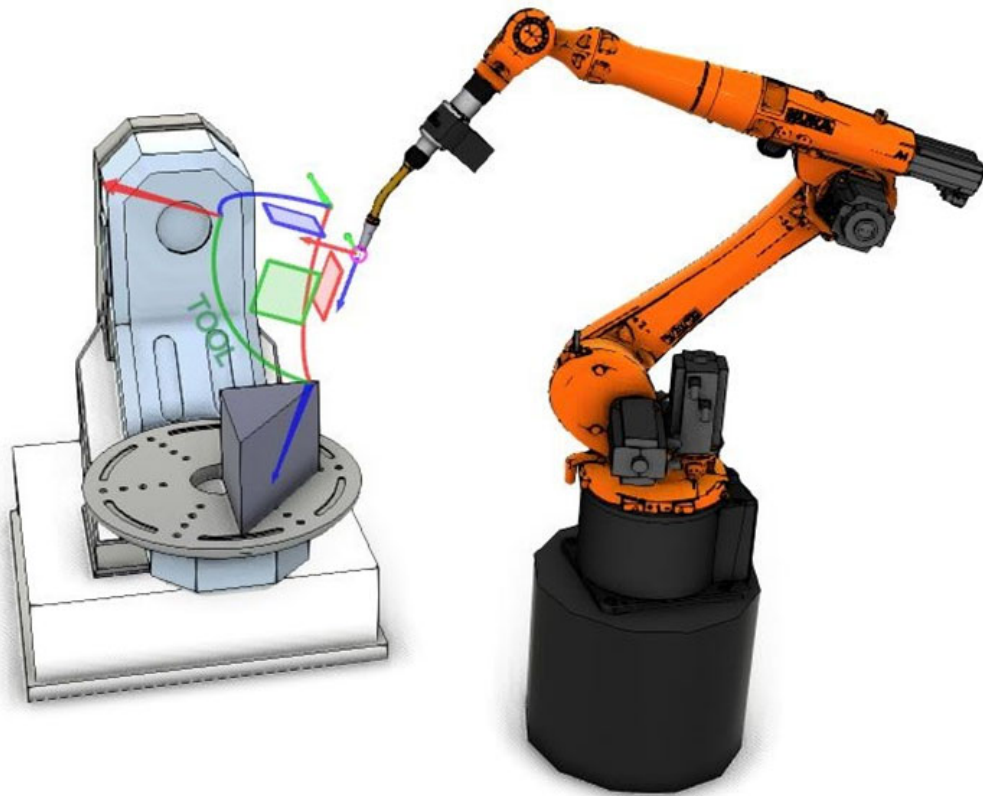


Figure 6 Tool Center Point (TCP)

The TOOL frame is a cartesian coordinate system defined at the tool center point (TCP). The tool TCP is the point used to move the robot to a cartesian position. The origin of the TOOL coordinate system is the robot's flange or faceplate, the tool's mounting point on axis 6. If no tool is defined, then the point used to move to a cartesian position is the robot flange itself. A TCP gets generally defined at the center of the tool or the point intended for positioning.

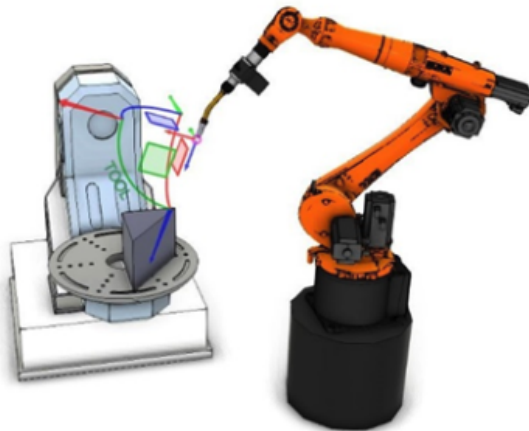
An example is a welding tooltip, gluing tool, gripper's center, or grippers matrix. When the robot is programmed to move to a specific point with the tool, the tool's TCP will end up at that point. Since a single robot can use multiple tools, the user can also define numerous tool frames.

Manually teaching a robot position in 3D space will be done using the currently defined TCP that will typically get expressed about the WORLD frame or a specified USER frame. Usually, points will get taught using the robot with the intended tool. But suppose a point is programmed offline, for example, when using data from a simulation. In that case, care needs to get taken to ensure that the point is programmed with the correct orientation so that the robot tool gets lined up accurately.

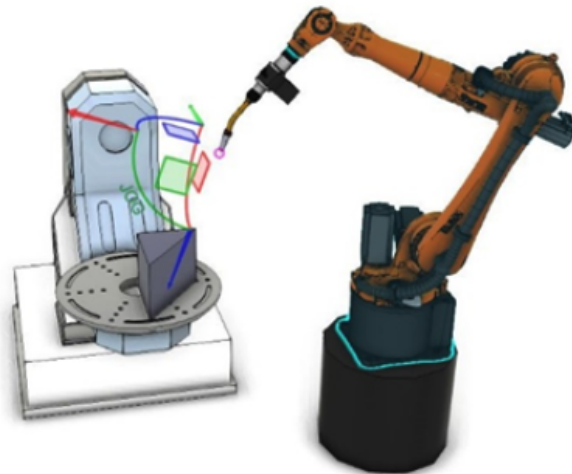
Configuring a Robot

Another set of indicators briefly mentioned earlier regarding point data is the robot configuration. When dealing with robots with multiple degrees of freedom and generous axis limits, the kinematic solvers can encounter problems if just given the position data in 3D space, i.e., X, Y, Z, A, B, and C. The issue is that points can get represented in multiple ways when using a robot with numerous degrees of freedom.

An example of this gets shown in "Table 1". In all these images, the robot is at the same position in space, the only difference being the robot configuration. "Table 1" displays this by showing the sign differences of the individual six-axis positions. A negative sign is an anticlockwise rotation around the axis center point, and a positive sign is a clockwise rotation about its center point.



Axis Number	Sign
Axis 1	Positive
Axis 2	Negative
Axis 3	Positive
Axis 4	Negative
Axis 5	Positive
Axis 6	Positive



Axis Number	Sign
Axis 1	Negative
Axis 2	Negative
Axis 3	Negative
Axis 4	Negative
Axis 5	Positive
Axis 6	Positive

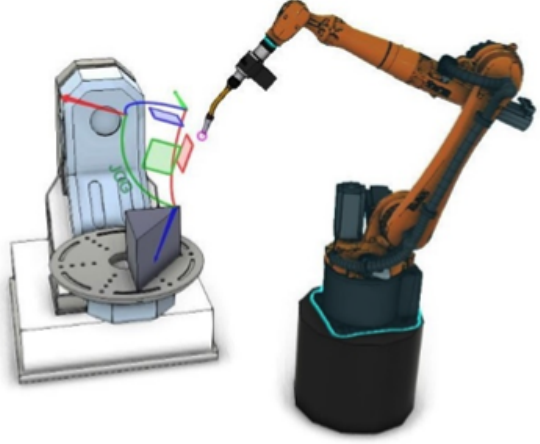
	<table border="1"> <thead> <tr> <th>Axis Number</th><th>Sign</th></tr> </thead> <tbody> <tr> <td>Axis 1</td><td>Negative</td></tr> <tr> <td>Axis 2</td><td>Negative</td></tr> <tr> <td>Axis 3</td><td>Negative</td></tr> <tr> <td>Axis 4</td><td>Negative</td></tr> <tr> <td>Axis 5</td><td>Negative</td></tr> <tr> <td>Axis 6</td><td>Negative</td></tr> </tbody> </table>	Axis Number	Sign	Axis 1	Negative	Axis 2	Negative	Axis 3	Negative	Axis 4	Negative	Axis 5	Negative	Axis 6	Negative
Axis Number	Sign														
Axis 1	Negative														
Axis 2	Negative														
Axis 3	Negative														
Axis 4	Negative														
Axis 5	Negative														
Axis 6	Negative														

Table 1 Robot Configuration

Since this robot can reach a point in multiple ways, a method or indicator is needed in which the point can describe its position in space unambiguously. These indicators are descriptions of the physical position of the robot's joints and their current interaction.

- Kuka uses two multi-bit indicators called status and turn, which consist of the sign of the current axis position and the axis intersection.
- Fanuc typically uses a combination of 3 joint placement and turn indicators for axes 4, 5, and 6.
- Stäubli uses three indicators representing the configuration of the robot arm's shoulder, elbow, and wrist, which correspond to sets of joints and their interactions.

This way, when the robot's TCP is at a point, it meets the position requirements of X, Y, Z, A, B, and C. The physical robot arm must also be representative of the configuration indicators. Therefore all the conditions necessary for a robot cartesian position are X, Y, Z, A, B, C, and the additional indicators.

Regardless of the robot used, they all require more than just the position data in 3D space for the point to be defined unambiguously. Since languages vary based on the platform, the method they use to describe the points tends to change slightly.

Conclusion

This tutorial covered the basics of robot position representation in 3D space. You now understand the concept of Frames and the three most essential frame types in robot programming. We also covered how a point gets represented in cartesian space and how this position needs additional indicators for a multi-axis robot.

Understanding these concepts will allow you to understand a robot system in more detail and build the skills necessary to develop more effective robot programs and applications.

[Start Discussion](#)

0 replies

Related tutorials

This is some text inside of a div block.

BEGINNER

Best PLC
Programming
Tools - Laptops,
Multimeters,
Cables,
Screwdriver Sets,
Switches, and
more...

[Read tutorial](#)

ADVANCED

Optimizing Torque
Control of Siemens
Servo Drives
Through TIA
Portal

[Read tutorial](#)

ADVANCED

Siemens S7-300 to
S7-1500 PLC
migration using
SIMATIC Manager
and TIA Portal

[Read tutorial](#)

The #1 platform for automation training and workforce development.	LEARN	RESOURCES	PLANS	COMPANY
	PLC programming	Tutorials	For Business	About us
	HMI development	Experts [NEW]	For Individuals	Careers
	Industrial robotics	Courses		Instructors
	Industrial networking	Video tutorials		Reviews
	Machine vision	Books		Contact Us
	Distributed IO	Blog		
	Variable frequency drives	Podcast		
	OPC-UA	Community		
	MQTT	Become an instructor		
		Become a content partner		