

A miniaturized space-variant active vision system: Cortex-I

Benjamin B. Bederson^{1,*}, Richard S. Wallace^{2,**}, Eric Schwartz^{3,***}

¹ Computer Science Department, University of New Mexico, Albuquerque, NM 87131, USA

² Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA

³ Department of Cognitive and Neural Systems, Boston University, 111 Cummington Street, Boston, MA 02146, USA

Abstract. We have developed a prototype for a miniaturized, active vision system with a sensor architecture based on a logarithmically structured, space-variant, pixel geometry. The central part of the image has a high resolution, and the periphery has a smoothly falling resolution. The human visual system uses a similar image architecture. Our system integrates a miniature CCD-based camera, a novel pan-tilt actuator/controller, general purpose processors, a video-telephone modem and a display. Due to the ability of space-variant sensors to cover large work spaces, yet provide high acuity with an extremely small number of pixels, architectures with space-variant, active vision systems provide a potential for reductions in system size and cost of several orders of magnitude. Cortex-I takes up less than a third of a cubic foot, including camera, actuators, control, computers, and power supply, and was built for a (one-off) parts cost of roughly US \$2000. In this paper, we describe several applications that we have developed for Cortex-I such as tracking moving objects, visual attention, pattern recognition (license plate reading), and video-telephone communications (teleoperation). We report here on the design of the camera and optics ($8 \times 8 \times 8$ mm), a method to convert the uniform image to a space-variant image, and a new miniature pan-tilt actuator, the spherical pointing motor (SPM), ($4 \times 5 \times 6$ cm). Finally, we discuss applications for motion tracking and license plate reading. Potential application domains for systems of this type include vision systems for mobile robots and robot manipulators, traffic monitoring systems, security and surveillance, telerobotics, and consumer video communications. The long-range goal of this project is to demonstrate that major new applications of robotics will become feasible when small, low-cost, machine-vision systems can be mass produced. We use the term “commodity robotics” to express the expected impact of the possibilities for opening up new application niches in robotics and machine vision, for what has until now been an expensive, and therefore limited, technology.

Key words: Cortex-I – Active vision system, miniature – Space-variant sensors – Spherical pointing motor

1 Introduction

Computer vision is a field that typically requires fast and expensive machines. One reason for this is the need to process images of size $O(N^2)$, for $N = 250 \rightarrow 1000$ at frame rates of 30–60 Hz. By using novel image architecture, image processing, and actuation, we have been able to construct a system that performs competitively with existing systems that are orders of magnitude larger and more costly to build. In other work, a detailed examination of the design considerations and advantages of space-variant sensing is provided [20], and it is shown that the human visual system, which uses a similar space-variant sensor design, achieves up to four orders of magnitude of image compression from this strategy.

Any system that uses a space-variant sensor must aim the sensor properly. Since space-variant sensors only have high resolution in the fovea, the current region of interest must be continuously tracked or foveated. Such a sensor must be mounted in a device that can aim it with precision. A system with this capability is an example of an active or attentive vision system [4, 5]. In other words, the advantages of space-variant sensing are provided at the cost of increased algorithmic and robotic complexity: high speed and accurate actuators and control strategies, attentional algorithms, novel image processing, and pattern recognition must be developed.

In this paper, we describe Cortex-I, a prototype miniaturized active vision system with a space-variant image architecture. This system integrates a CCD sensor, a miniature pan-tilt actuator, a controller, a general purpose processor and a display (Fig. 1). The system can be connected to a host processor for applications development, integrated as the visual module of a larger robotic system, or operated in a stand-alone configuration.

As an architecture for image processing and computer vision, our prototype represents a significant departure from the types of large-scale parallel vision systems we and others

* e-mail: bederson@cs.unm.edu

** e-mail: rsw@cs.nyu.edu

*** e-mail: eric@thing4.bu.edu

Correspondence to: B.B. Bederson

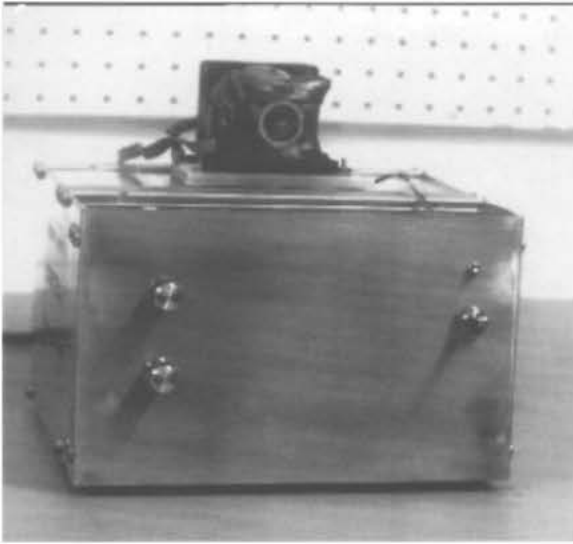


Fig. 1. Cortex-I: a prototype of a miniaturized, active vision system

have previously investigated [14, 19, 28, 31a]. Our design is a small, loosely coupled network of embedded microcomputer modules, integrated directly with a sensor and robotic pan-tilt actuator. Moreover, our architecture has benefited from research into the space-variant nature of the human visual system [23, 25–27, 32]. We use a logmap sensor, a useful type of space-variant sensor that is modeled on the human visual system. The space complexity aspect of logmap sensors is particularly attractive and has been analyzed in detail [20].

Other researchers have developed a variety of active vision systems that mimic some biological functions such as pan, tilt, vergence and focus control, attentional algorithms, and real-time tracking [1–3, 6, 11, 13–16]. Through implementing image processing at the low data rate of a logmap sensor, and exploiting our novel pan-tilt mechanism, our prototype demonstrates a significant subset of these capabilities with only modest processing power and inexpensive components. The two principles of embedded system design and space-variant active sensing, taken together, enable us to develop new applications flexibly at a very low cost.

2 Space-variant images

A space-variant image sensor is characterized by an irregular pixel geometry. We have experimented with several sensor designs, including custom VLSI (in collaboration with Synaptics, Inc.). Their common features are large-scale changes between the smallest and largest pixels, a wide field of view, and a pixel population far smaller than that of a conventional uniform image sensor. One such space-variant geometry is logarithmic [20, 21], in which the pixel pattern approximates the sensor geometry of the human eye. In this section, we review space-variant images. See [30] for more detail on this subject.

The mapping defined by the logarithmic geometry is illustrated in Fig. 2. It is convenient to represent the point in the domain, P_1 , as $re^{i\theta}$ and the point in the range, P_2 , as the complex number $x+iy$. If we take the log of P_1 , we get:

$$\log(re^{i\theta}) = \log r + i\theta$$

Substituting

$$x = \log r \quad \text{and} \quad y = \theta,$$

we get

$$\log(re^{i\theta}) = x + iy$$

or,

$$\log(P_1) = P_2$$

so that the (complex) logarithm of a point in polar coordinates is transformed to a point in rectangular coordinates.

This mapping has some interesting geometric properties. Specifically, fovea-centered circles are mapped to vertical lines, and radial lines are mapped to horizontal lines. This has the effect of transforming the scale and rotation in fovea-centered retinal images to translation in cortical images. Computationally, translation is easier to handle than scaling or rotation, so this provides another justification for a complex log sensor geometry, which has attracted considerable interest during the past 15 years. In the present work, we do not attempt to use this feature of complex log mapping; we merely see it as a convenient representation for space-variant sensing, and one which is motivated by the similar structure of the human visual system.

One problem in using the analytic space-variant mapping, $\log(P)$, is the existence of a singularity at the origin of the coordinate system. Figure 2b shows this singularity. The left points of each triangle in the range all represent the same point in the domain. Our solution to this problem is the introduction of a small real constant, α , into the mapping and the use of a map function of the form $\log(z + \alpha)$. We call this mapping the “logmap”. The constant α is used in an analogous fashion in models of the human visual system: it is a measure of the size of the central linear representation of the human fovea [25].

As can be seen in Fig. 2d, the pixels on the edge are cut off in different places yielding 3, 4, and sometimes 5-sided pixels. This greatly complicates image processing. Even simple operators, such as convolution, are difficult to implement. We have found a general solution to this problem, which in effect, generalizes image processing to sensors of arbitrary neighborhood relations or topology. This work is described in detail in Sect. 4.2.

We have followed two approaches to acquiring logmap images in real time. The first involves the design and fabrication of a custom VLSI sensor, in collaboration with Synaptics. This work is in an early stage, and we do not describe it further here. The second approach is to use an imbedded processor, together with a conventional CCD imaging chip, to produce the logmap transformation by a real-time, lookup-table technique. Thus we can formally define the logmap as a mapping from a TV image, $I(i, j)$, where $i \in \{0, \dots, \text{NROWS} - 1\}$ and $j \in \{0, \dots, \text{NCOLS} - 1\}$, let $L(u, v)$ be the logmap, with $u \in \{0, \dots, \text{NSPOKES} - 1\}$ and $v \in \{0, \dots, \text{NRINGS} - 1\}$.

The forward mapping from TV image space to logmap space is specified by the spoke and ring lookup tables, $S(i, j)$ and $R(i, j)$ (Fig. 3), where again $i \in \{0, \dots, \text{NROWS} - 1\}$ and $j \in \{0, \dots, \text{NCOLS} - 1\}$. Let $a(u, v)$ be the area (in TV pixels) of a logmap pixel (u, v) .

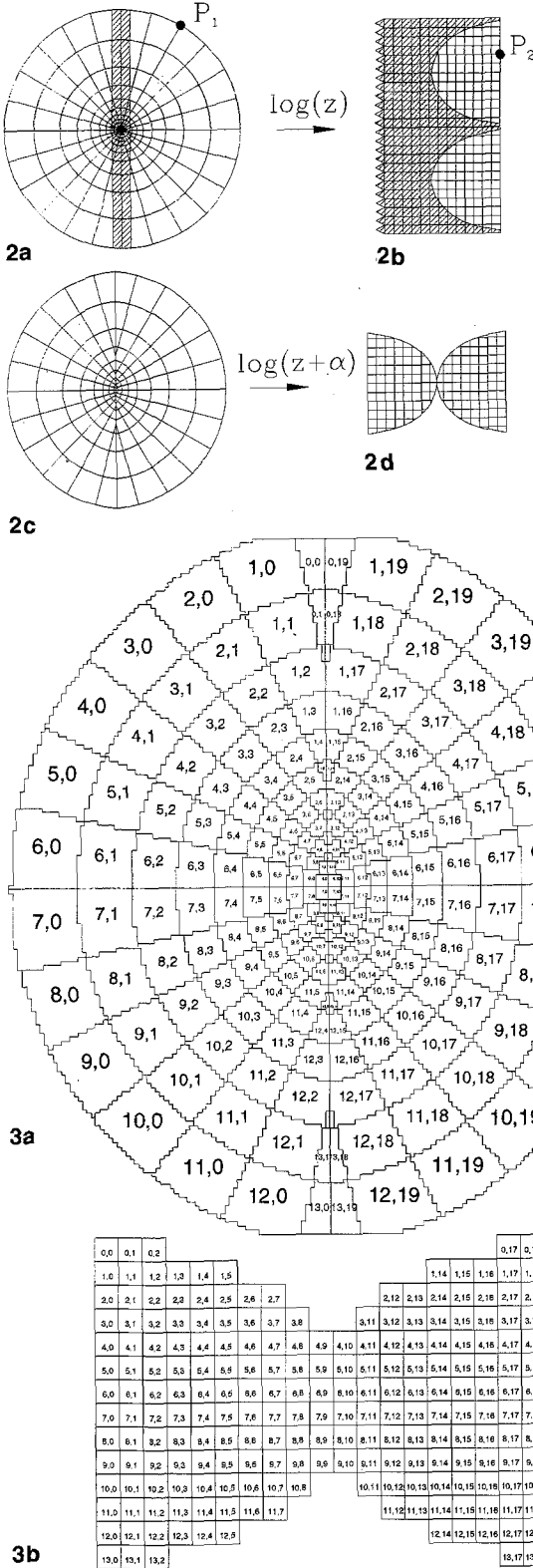


Fig. 2a-d. This shows the complex logmap: **a,b** $\log(z)$ **c,d** its relative, the mapping, $\log(z + \alpha)$. It can be seen that the $\log(z + \alpha)$ map results from cutting the hatched region out of the $\log(z)$ map. Here, P_1 is mapped to P_2 . The $\log(z)$ map has a singularity in the center while the $\log(z + \alpha)$ map has a branch-cut along the vertical meridian

Fig. 3. a The lookup tables $S(i, j)$ and $R(i, j)$ combined into one figure; **b** the forward map space. The correspondence between inverse and forward map pixels is shown

$$a(u, v) = \sum_{i,j} 1 \mid S(i, j) = u \text{ and } R(i, j) = v. \quad (1)$$

The logmap (or forward map) image (Fig. 2c) is defined by

$$L(u, v) = \frac{1}{a(u, v)} \sum_{i,j} I(i, j) \mid S(i, j) = u \text{ and } R(i, j) = v. \quad (2)$$

The inverse map, illustrated in Fig. 2b, is

$$L^{-1}(i, j) = L(S(i, j), R(i, j)). \quad (3)$$

The relationship between a space-variant image and the lookup tables $S(i, j)$ and $R(i, j)$ is illustrated by comparing Fig. 3 with Fig. 4. The values in the lookup tables depict the row and column addresses of pixels in the space-variant image array. We observe that if n is the number pixels for which $a(u, v) > 0$, then $n \leq \text{NSPOKES} \times \text{NRINGS}$, and we define $\text{NPIXELS} = n$.

3 System description

Cortex-I consists of the emulated logmap sensor, a miniature pan-tilt actuator, a controller, a general purpose processor, a display, and an optional video telephone interface. The controller consists of a camera driver, a 2-MIPS programmable microcontroller (Motorola MC68332), a video display driver, and three 12-MIPS digital signal processors (Analog Devices AD2101). The actuator and camera are mounted to the electronics chassis ($14 \times 22 \times 22$ cm) and connected by twisted-pair cables. The system is powered from a standard 110-volt AC line, but uses less than 25 watts and could be battery powered.

The camera consists of a miniature, commercially available, CCD image sensor and a custom lens assembly (fabricated in collaboration with Barry Levin) mounted in the actuator. The camera image is mapped to a logmap image with a fast lookup-table algorithm, as we have already outlined. For the mapping that we have used extensively, the maximum resolution is $0.175^\circ/\text{pixel}$ and the horizontal field of view measures 33° . The sensor has a fixed focus 4-mm lens with a set of fixed, manual apertures (three sizes). Since objects that are more than 10 focal lengths from the camera essentially have an infinite depth of field, we avoid the need of focus for working distances larger than about 40 mm. The system outputs up to 30 frames/s and measures 256 gray levels/pixel. The camera head (CCD and lens assembly) measures only $8 \times 8 \times 10$ mm, and is controlled by a lab-built driver board that provides timing signals to the sensor and converts analog sensor data to 8-bit digital data for the processors.

The spherical pointing motor (SPM) is a novel pan-tilt actuator using three orthogonal motor windings to achieve open-loop pan-tilt actuation of the camera sensor in a small, low-power package. The SPM can orient the sensor through approximately 60° pan and tilt, at speeds of several hundred degrees per second. It measures $4 \times 5 \times 6$ cm and weighs 170 g. It is actuated by currents of roughly 50 mA in each of the three coils.

The MC68332 controls the SPM and runs the application software. The MC68332 can be connected to a host processor for applications development, to upload sensor data, or

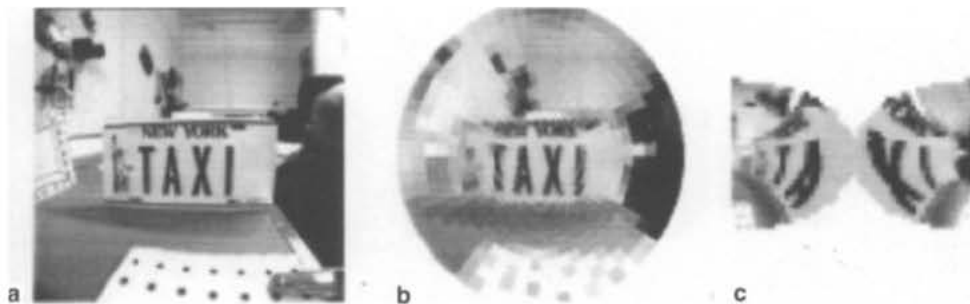


Fig. 4. **a** A TV image, $I(i, j)$; **b** the inverse logmap image $L^{-1}(i, j)$; **c** the forward logmap image $L(u, v)$

to receive pan-tilt commands. The MC68332 is a 32-bit 16-MHz microcontroller integrating peripheral controls, such as programmable digital control lines and timing signals, directly on chip. We connected a 192-kB RAM and a 128-kB EPROM to the microcontroller. The logmap data is output via a high-speed serial interface, either to an external device such as a host computer, to the internal display driver, or to the video-telephone transmitter.

Several image processing demonstrations have been implemented in the microcontroller ROM. These include a simple motion-tracking program that turns the camera to center the observed motion field, at 16 frames/s. The ROM also contains a test pattern, an image-binarization program (22 frames/s), and a motor motion demo. Other demonstrations illustrate the connectivity graph (Sect. 4.2) to implement image smoothing (3 frames/s), edge detection (2 frames/s) and connected components (1 frame/s), illustrating both the generality of the programming model and the limitations of the relatively low power microcontroller used in the current implementation of Cortex-I.

The Analog Devices 2101 digital signal processor (DSP) provides additional real-time, image-processing capabilities. The DSP controls the CCD readout, computes the logmap, and sends the logmap image to the MC68332 via a high-speed serial connection. The DSP board combines an AD2101 12 MHz DSP having two 2-MHz serial ports, an 8-kB internal RAM, an 80-kB external RAM, and a 64-kB boot EPROM. A second identical DSP board functions as a video display driver, generating an RS-170 video output suitable for display on a standard TV monitor. An optional third DSP board is used as a video transmitter that can send four logmap images (uncompressed) at 4 frames/s over a standard voice-grade telephone line. To achieve this, the DSP implements a custom-designed protocol for an analog modem that encodes the logmap frames.

3.1 Camera sensor

We have pursued two routes to real-time acquisition of logmap images. The first is to develop a custom VLSI sensor chip with a logmap geometry that consists of pixels varying in size and arranged in a logmap geometry. The second is to use a commercially available sensor chip that returns conventional rectangular images and maps them into logmap images.

The main advantages of a custom logmap sensor chip are the high frame rate and small resultant system size. This is

because the geometry of the chip allows intrinsic mapping. The sensor layouts that we have designed consist of several thousand pixels on chip, and this is a very small number of pixels to digitize. It is likely that a frame rate of thousands of frames per second could be supported, contingent on the ambient light intensity. However, at the present time, we do not have a fully working space-variant chip.

The main advantages of an off-the-shelf conventional (uniform) sensor chip are low price, high image quality, and immediate availability. Because uniform sensors have been refined for many years and are produced in great quantities, they have extremely high quality, which is difficult to match with custom sensor chips. The image from a uniform sensor, however, must be mapped to a logmap image and this is computationally expensive, since the final logmap image size of thousands of pixels must be supplied through a read-out bottleneck of hundreds of thousands of pixels on the chip. However, with careful design of the readout and logmap emulation algorithms, it is possible to achieve rates as high as 100 frames/s, without resorting to excessively complex readout electronics.

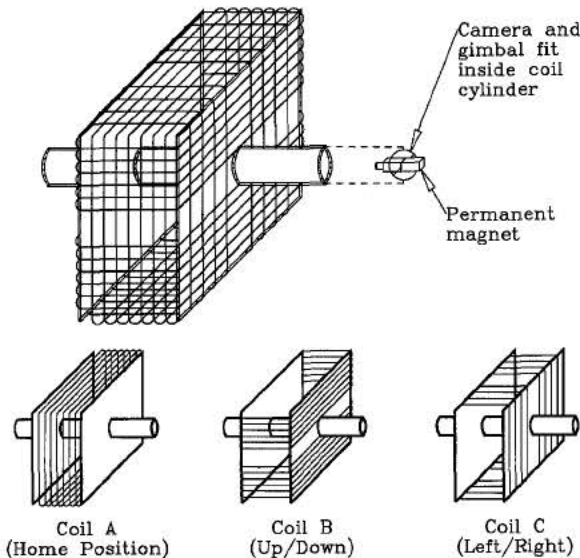
3.2 Spherical pointing motor (SPM)

A pan-tilt mechanism is a computer-controlled actuator designed to point an object such as a camera sensor. For applications in active vision, we prefer a pan-tilt mechanism to be accurate, fast, small, low-power, and inexpensive. We have constructed two actuators meeting these requirements: one based on a parallel linkage actuated by two identical, small, stepper motors [7, 9]; the other incorporates both pan and tilt into a single two-axis actuator (Fig. 5). The SPM [protected by the U.S. patent no. 5204573 entitled "Spherical Pointing Motor" (1993)] is described in detail in [8, 10, 31b, 31c], but is summarized here. The SPM consists of three orthogonal motor windings in a permanent magnetic field, configured to move a small camera attached to a gimbal.

The simplest and most obvious direct drive pan-tilt mechanism is the two-stage motor-on-motor (MOM) design. The first motor turns the mechanism through one degree of freedom, usually pan, and the second through the other degree of freedom usually tilt. The second motor must be powerful enough to move the camera sensor. The first must move both the camera and the second motor. The MOM design therefore usually consists of one larger motor and one smaller one. Because of the large accelerations involved in starting and stopping a high-speed device accurately, the ex-



5



6

Fig. 5. The spherical pointing motor. At the center is a miniature camera consisting of a single CCD sensor chip and a lens assembly that fits on the rotor of this motor

Fig. 6. Illustration of the external spherical pointing motor shown in its home position with labels for the three coils

tra inertial load presented by the MOM design requires very large torques in the primary motor.

An alternative to the direct drive MOM design requires some form of linkage to couple the camera to the motor. We have built a stepper-motor actuated, modified Stewart-platform device, but it is our impression that direct drive DC motors are advantageous, except for the limitations (already noted) of the MOM design. It is clear that some form of direct drive design, in which the MOM design is avoided, but in which no linkages are required, would be ideal. We have designed such an actuator, which provides a spherically symmetric, direct drive, DC motor actuator, namely the SPM.

The SPM is an absolute positioning device, designed to orient a small camera sensor in two degrees of rotational freedom. The basic principle is to orient a permanent magnet in three orthogonal motor windings by applying the appropriate ratio of currents to the three coils. The SPM can have the coils either on the outside with the permanent mag-

net rotating on the inside, or vice versa. Figure 6 depicts an example of the former.

3.3 Microcontroller board

A Motorola MC68332 microcontroller controls the SPM and runs some applications software. The MC68332 can be connected to a host processor for applications development, or to upload sensor data and to receive pan-tilt commands. The MC68332 is a 32-bit 16-MHz microcontroller integrating peripheral controls, such as programmable digital control lines and timing signals, directly on chip. The logmap data is output via a high-speed serial interface, either to an external device such as a host computer or to the internal display driver.

We have implemented image processing demonstrations in the microcontroller ROM. A simple motion tracking program, which turns the camera to center the observed motion field, runs at 16 frames per second (16 frames/s). The ROM also contains a test pattern, an image binarization program (22 frames/s) and a motor scan-sequence demo. We also use the connectivity graph (Sect. 4.2) to smooth images (3 frames/s), detect edges (2 frames/s) and connect components (1 frame/s), illustrating both the generality of the programming model and the limitations of the microcontroller in the current prototype.

3.4 Digital signal processors (DSPs)

The Analog Devices AD-2101 DSP provides additional real-time image processing capabilities on logmap images. The DSP reads the camera board, computes the logmap and sends the resulting image to the M68332 via a high-speed serial connection. The DSP board combines an AD 2101 12 MHz DSP having two 2-MHz serial ports, an 8-kB internal RAM, an 80-kB external RAM, and a 64-kB boot EPROM. An second identical DSP board functions as a video display driver, generating an RS-170 video output suitable for display on a standard TV monitor. An optional third DSP board is used as a video transmitter that can send logmap images over a standard voice-grade telephone line at 4 frames/s.

4 Programming and development model

4.1 Embedded system model

Each of the three microprocessors in our prototype is a general purpose computer, so the programming model bears great resemblance to that for embedded control systems on general purpose processors. One of the key elements in this model is a distinction between a development system and a target system. We initially developed our applications on a Sun workstation-based system. This development system simulates the target system closely: it combines a robotic pan-tilt actuator made by Klinger Scientific, a Sony XC-77RR camera and Analogic video digitizer, with software tools to simulate a variety of sensor geometries.

There are four stages to developing an application.

1. We design a prototype program on the development host. At this stage, there are few memory, timing, or numeric precision restrictions. The commercial camera and actuator provide a high degree of reliability.
 2. Development continues on the host, but we use the camera and actuator of Cortex-I. The host controls the SPM by sending commands over an RS-232 serial line, and receives the logmap image from the miniature camera, digitizing the video signal output. This allows the same software to be tested with the new hardware.
 3. The application is debugged on the target hardware using the host as a terminal. Since *C* language cross-development environments are available for both the DSPs and the microcontroller used in this system, we develop *C* code on the Sun host, then download it to the target system. The code must account for the more stringent hardware restrictions such as integer arithmetic, less memory, and less processing power. Such restrictions are a result of the inherent tradeoff between a highly flexible, programmable development system and an inexpensive target system.
 4. We burn the application into the ROM of the target system, at which point we can take the target system "into the field."
- Some applications, like an attentional OCR program for reading license plates [17, 18] are at stage two, while many simpler image processing algorithms, such as motion tracking, are at stage four.

4.2 Space-variant image processing

As mentioned earlier, the complex logmapping has a singular point at the origin, and a branch cut associated with the "phase-wrapping" of the angles 0 and 2π . In addition, the local pixel connectivity does not generally possess the simple 4-connectivity or 8-connectivity familiar in conventional computer graphics and image processing. The local topology of the complex log sensor is variable and complicated, and this is a feature that might be shared by other novel sensor architectures. To deal with image processing on an arbitrary sensor topology, we developed an approach to image processing based on the use of a connectivity graph. We define the connectivity graph, $G = (V, E)$, as the graph whose vertices, V , stand for sensor pixels and the edges, E , represent the adjacency relations between pixels. Associated with a vertex p is a pixel address (u, v) . Thus we write $(u(p), v(p))$ for a pixel coordinate identified by its graph vertex.

Each graph node, p , in the connectivity graph is represented by a constant size data structure in memory. The graph edges $\{(p, q)\}$ are represented by one field containing a list of pointers. The structure for p may also contain fields such as the pixel array coordinates $(u(p), v(p))$, the pixel centroid $\mu(p)$, the pixel area $a(p) = a(u(p), v(p))$, and the number of neighbors $|\mathcal{N}(p)|$. The pixel centroid $\mu(p)$ represents the centroid of the TV pixels that map to p and is given by

$$\mu(p) = \frac{1}{a(p)} \sum_{i,j} (i, j)^T \mid (S(i, j) = u(p) \text{ and } R(i, j) = v(p)) . \quad (4)$$

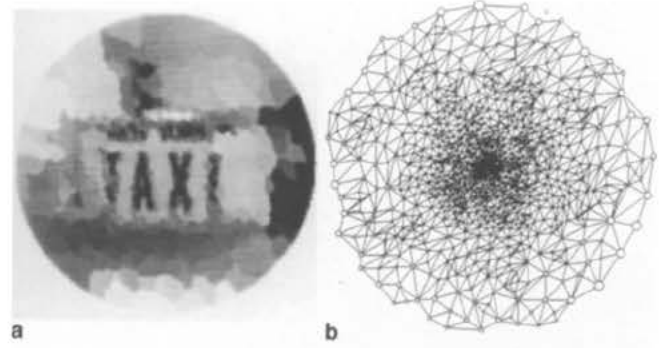


Fig. 7. a The image from a sensor having a random pixel geometry; b the connectivity graph for this sensor

We define the set of neighbors of p to be denoted by

$$\mathcal{N}(p) = \{q \mid (p, q) \in E\} \quad (5)$$

We can define a space-variant sensor geometry that is not a complex logarithm, but is the result of a stochastic process. We define this map by placing seed points randomly within the image space with a density function defined so that more points will be in the center of the image. The map is then created by growing each seed point until its neighbor is met via a simple relaxation procedure. The map is called the random map and is illustrated in Fig. 7.

One application for the random map would be to use it with large area CCD image sensors with defective pixels. Large CCD sensors ($1k \times 1k$ or $2k \times 2k$ pixels) are extremely expensive because of their low production yield. These sensors are much cheaper when they have defective pixels. A random map with the associated connectivity graph could be created for each of these sensors, skipping the faulty pixels. This would allow for a very high-resolution camera that would not be prohibitively expensive.

A variety of space-variant sensors have been designed and fabricated [12, 20–22, 24, 27]. The connectivity graph can be defined for any of these mappings, and then used to develop a library of image processing routines that can run on data from any of these sensors, and therefore we provide a generic approach to image processing on an unconstrained sensor geometry [29].

Using the connectivity graph, we define a variety of image processing operations. For example, if p represents a pixel, $L(p)$ its gray value, and $\mathcal{N}(p)$ its neighbors, then we can define a simple edge operator $e(p)$ as

$$e(p) = \frac{1}{|\mathcal{N}(p)|} \sum_{q \in \mathcal{N}(p)} (L(p) - L(q))^2 \quad (6)$$

Note that the definition of $e(p)$ contains no special cases for pixels having differing numbers of neighbors, and thus applies equally to pixels at the boundary of the image and to those in the interior.

Another simple example is the Laplacian, in sensor coordinates, $\ell(p)$, given by

$$\ell(p) = |\mathcal{N}(p)| L(p) - \sum_{q \in \mathcal{N}(p)} L(q) \quad (7)$$

Using such definitions, we can build a library of image processing routines that is independent of the sensor geometry. We report the details of the connectivity graph in another paper [29], where we show how to develop programs for template matching, connected components analysis, pan-tilt calibration, and pyramid operations.

5 Applications

5.1 Motion tracking

We have implemented a motion-tracking application on Cortex-I in ROM that integrates all the capabilities of the system. It is based on a simple frame-differencing algorithm and is able to track moving objects at approximately 5 frames/s.

The algorithm subtracts successive pairs of frames, computes the centroid of motion, and moves the motor to point the camera at the computed centroid. Because this algorithm needs two frames to compute the centroid of motion, and then misses a frame while the motor is moving, it can only make one motor movement every three frames. Therefore, although it can compute the frame difference and centroid at 16 Hz, the motor is only moved about 5 times/s.

5.2 License plate tracking and reading

We solved a difficult tracking and pattern recognition problem using a combination of Cortex-I and a Sun Sparcstation, bringing it to the second stage of development (as described in Sect. 4.1). This application is described in detail in [17, 18], but is summarized here. The task is to find the license plate on a moving car, track it as the car moves towards the camera, and finally read the characters on the license plate when it gets close enough to the camera.

The experimental setup consists of a real license plate attached to a toy truck that is pulled by a string wound around a motor-driven pulley. The truck moves at approximately 5 cm/s. It is tracked for about 165 cm before the license plate is read.

Cortex-I is set up to allow commands sent via an RS-232 serial port to control the SPM position. The commands specify the pan and tilt angles of the motor in tenths of a degree. Cortex-I outputs a standard RS-170 video signal that contains the inverse logmap image. This signal is then digitized on a Sun SPARC-2, and the forward logmap is computed from the inverse image just as if it were a TV image. This method is very inefficient as it wastes communication bandwidth in transmitting an inverse image, and computational resources in computing the forward map from the inverse map. We chose this approach because it allows the existing application running on the host to use the Cortex-I hardware.

We track the license plate using a combination of three techniques that are a modified hough transform [21], a foreground-background segmentation, and a corner detector. We assume that the car moves at a constant velocity, and we estimate the license plate's position in each frame, using a Kalman filter. When the car is close enough, and thus the characters on the license plate are large enough, the

camera scans the license plate from left to right with five frames. The program analyzes each frame to segment the characters and recognize them optically. It then merges the results from the five frames to yield the string of characters on the license plate.

5.3 Cortex-I limitations

The development of the application described in the previous section provides a better understanding of some of the limitations of using Cortex-I in a real situation. We found that, although Cortex-I works well, it is limited in several ways, which are:

1. *Not enough processing power.* The Motorola M68332, which is capable of about 2 MIPS, processes all the imaging in the system, as well as controlling the motor. This is barely enough to track the license plate even at these low speeds.
2. *Not enough memory.* The M68332 board has only 192 kB of RAM. This is not enough to hold the character recognition tables, and thus the application could not be advanced further than stage two at this point.
3. *Not enough sensor resolution.* By the time the license plate is close enough to the camera so that the characters are reliably recognizable, the truck is so close to Cortex-I that there is not enough time to scan the license plate and read the characters. Because of this, the characters on the license plate must be read while the license plate is still too far away, resulting in the imperfect recognition results.
4. *Camera does not rotate about its focal point.* The application software assumes the camera rotates about its focal point when it scans the license plate to read the characters. Because the camera does not, in fact, rotate about its focal point, the rotation results in an effective translation that throws off the expected position of the camera. Because of this, the OCR results from the various frames are not merged accurately, yielding an incorrect resultant string.

6 Conclusion

To build a miniature system, it is necessary that *every* component be miniature. We have successfully built a working miniature system that consists of a miniature camera, an actuator, a DSP board to read out and map the camera image to a logmap image, a microcontroller board to control the motor and process the images, a DSP board to display the image, and a DSP board to transmit the images over a voice-grade telephone line at 4 frames/s.

We have shown that relatively little processing power is required to effect significant real-time vision with a logarithmically structured sensor architecture. Our current system has roughly 2 MIPS of processing power. For a future version, we estimate that 50 MIPS would provide enough power for extensive real-time applications of computer vision.

Although we are currently participating in the fabrication of a custom VLSI sensor, we have been able to emulate a logmap sensor with commercially available hardware. This

sensor architecture, coupled with currently available general purpose microcontrollers and DSPs, enables the building of a high-performance machine vision system without extensive investment in custom VLSI.

The SPM is comparable in capabilities to, yet at least an order of magnitude less in size and cost than, other similar pan-tilt actuators. The current prototype (camera and motors) is about 5 cm on a side, draws about 10 watts, and is able to actuate our 6-g camera at speeds of several hundred degrees per second. It is also easy to use as it is an absolute positioning device and is run open-loop. A set of currents applied to the coils moves the motor and holds it at a fixed position, which can be specified to a fraction of a pixel.

The long-range goal of this project is to demonstrate that major new applications of robotics will become feasible when small low-cost machine vision systems can be mass produced. This notion of "commodity robotics" is expected to parallel the impact of the personal computer, in the sense of opening up new application niches for what has until now been an expensive, and therefore limited, technology.

Acknowledgements. This work supported in part by DARPA/ONR Contract No. N00014-90-C-0049, and AFOSR Contract No. 88-0275.

References

- Abbott AL, Ahuja N (1990) Active surface reconstruction by integrating focus, vergence, stereo, and camera calibration. International Conference on Computer Vision. IEEE Computer Society Press, Los Alamitos, CA
- Abbott AL, Ahuja N (1991) The University of Illinois active vision system. Technical Report CV-91-8-2, University of Illinois, Beckman Institute
- Allen PK, Yoshimi B, Timcenko A (1991) Real-time visual servoing. Proceedings of the 1991 IEEE International Conference on Robots and Automation, IEEE Computer Society Press, Los Alamitos, CA
- Bajcsy R (1988) Active perception. IEEE Proceedings, 76:996-1005
- Ballard DH (1989) Animate vision. Technical Report, Computer Science Department, University of Rochester, Rochester, NY
- Baloch AA, Waxman AM (1991) Behavioral control of the mobile robot mavin. In: Antognetti, Milutinov (eds) Neural networks: concepts, applications, and implementations, Volume IV. Prentice Hall, Englewood Cliffs, NJ
- Bederson BB (1992) A miniature space-variant active vision system: Cortex-I. PhD Thesis, Computer Science Department, Graduate School of Arts and Science, New York University, New York
- Bederson BB, Wallace RS, Schwartz EL (1992) A miniature pan-tilt actuator: the spherical pointing motor. Technical Report 264, New York University, Computer Science Department, Robotics Research, New York, NY
- Bederson BB, Wallace RS, Schwartz EL (1992) Two miniature pan-tilt devices. IEEE International Conference on Robotics and Automation. IEEE Computer Society Press, Los Alamitos, CA, pp 658-663
- Bederson BB, Wallace RS, Schwartz EL (1994) A miniature pan-tilt actuator: the spherical pointing motor. IEEE Trans Robotics Automation
- Brown C (ed) The Rochester robot. Technical Report 257, University of Rochester, Rochester, NY
- Chaikin GM, Weiman CFR (1981) Image processing system. US Patent No. 4 267 573
- Clark JJ, Ferrier NJ (1988) Modal control of an attentive vision system. Second International Conference on Computer Vision, p 514
- Dickmanns ED, Graefe V (1988) Applications of dynamic monocular machine vision. Machine Vision Appl 1:241-261
- Kawarabayashi H, Watanabe M, Shirai Y, Asade M, Miura J (1991) Tracking of a moving object using an active vision system. Japan Mechanical Society, Robotics Mechatronics Conference Proceedings, pp 207-212
- Krotkov E, Fuma F, Summers J (1988) An agile stereo camera system for flexible image acquisition. IEEE J Robotics Automation 4:108-113
- Ong P-W (1992) Image Processing, pattern recognition and attentional algorithms in a space-variant active vision system. PhD Thesis, Computer Science Department, Graduate School of Arts and Science, New York University, NY
- Ong P-W, Wallace RS, Schwartz EL (1992) Space-variant optical character recognition. 11th International Conference on Pattern Recognition. IEEE Computer Society Press, Los Alamitos, CA
- Rizzi AA, Whitcomb LL, Koditschek DE (1991) Distributed real-time control of a spatial robot juggler. IEEE Computer Magazine
- Roger AS, Schwartz EL (1990) Design considerations for a space-variant visual sensor with complex-logarithmic geometry. 10th International Conference on Pattern Recognition Atlantic City, NJ, IEEE Computer Society Press, 2:278-285
- Roger AS (1989) Space-variant computer vision with a complete-logarithmic sensor geometry. PhD thesis, Computer Science Department, Graduate School of Arts and Science, New York University, New York
- Sandini G, Bosero F, Bottino F, Ceccherini A (1989) The use of an anthropomorphic visual sensor for motion estimation and object tracking. Image Understanding and Machine Vision Workshop. Optical Society of America, pp 1-5
- Reference deleted
- Sandini G, Dario P, Debusschere I (1989) Active vision based on space-variant sensing. 5th International Symposium on Robotics Research, MIT Press, MA
- Schwartz EL (1977) Spatial mapping in primate sensory projection: analytical structure and relevance to perception. Biological Cybernetics 25:181-194
- Schwartz EL, Merker B, Wolfson E, Shaw A (1988) Computational neuroscience: applications of computer graphics and image processing to two and three dimensional modeling of the functional architecture of visual cortex. IEEE Comput Graph Appl 8:13-28
- Spiegel J van der, Kreider F, Claeys C, Debusschere I, Sandini G, Dario P, Fantini F, Belluti P, Soncini G (1989) A foveated retinal-like sensor using ccd technology. In: Mead C, Ismail M (eds) Analog VLSI Implementations of Neural Networks. Kluwer, Boston
- Wallace RS, Howard MD (1984) HBA vision architecture: built and benchmarked. IEEE Trans Patt Anal Machine Intell
- Wallace RS, Ong P-W, Bederson BB, Schwartz EL (1991) Space-variant image processing. Technical Report 256, New York University, Computer Science Department, Robotics Research, New York, NY
- Wallace RS, Ong P-W, Bederson BB, Schwartz EL (1994) Space-variant image processing. Int Comp Vision 13: 71-94
- Wallace RS, Webb JA, Wu IC (1989) Architecture independent image processing: performance of apply on diverse architectures. Comp Vision Graph Information Processing, 48:265-276
- Wallace RS (1994) Miniature direct drive rotary actuators II; Eye, finger and leg. Robotics and autonomous systems 13:97-105
- Wallace RS (1993) Miniature direct drive rotary actuators. Robotics and autonomous systems II: 129-133
- Weiman CF, Chaikin G (1979) Logarithmic spiral grids for image-processing and display. Comp Graph Image Processing 11:197-226

Benjamin B. Bederson completed his Ph.D. in 1992 and his M.S. in 1989 at New York University in the Courant Institute of Mathematical Sciences in Computer Science. He graduated with a B.S. from Rensselaer Polytechnic Institute in 1986. From 1990–1992, he was a research scientist at Vision Applications, Inc. performing the work described in this paper. Dr. Bederson worked as a research scientist at Bellcore in the Computer Graphics and Interactive Media research group, and as a visitor at the New York University Media Research Laboratory in 1993 and 1994. He is now Assistant Professor of Computer Science at the University of New Mexico, and is working on intuitive user interfaces.

Richard S. Wallace completed his Ph.D. in 1989 and his M.S. in 1986 at Carnegie Mellon School of Computer Science. He graduated with a B.A. from University of Southern California in 1982. From 1986–1989, Wallace was a Member of the Technical Staff, Hughes Research Labs, Malibu, CA. In 1989–1990 he worked as Post-doctoral Fellow at NTT Human Interface Labs, Yokosuka, Japan. Then, from 1990–1992 he was General Manager of Vision Applications, Inc., a research and development firm. Dr. Wallace is now Assistant Professor of Computer Science at the Courant Institute of Mathematical Sciences, New York University. His research interests include robotics, computer vision and pattern recognition.

Eric L. Schwartz received the Ph.D. degree in high energy physics from Columbia University in 1973, and has served as Associate Professor of Computer Science (Courant Institute of Mathematics) and Psychiatry (New York University Medical Center). He is currently Professor of Cognitive and Neural Systems, Professor for Electrical and Computer Systems and Professor of Anatomy and Neurobiology at Boston University. His research interests include experimental studies of cortical visual architectures in monkeys and humans, and computational modeling of visual anatomy, psychophysics, physiology and function. In 1989, he founded Vision Applications, Inc., where he has served as Chief Research Scientist, and consulted on the design and implementation of miniaturized computer vision systems for performing real-time space-variant active vision.