A Binocular, Foveated Active Vision System

Brian Scassellati[*]
545 Technology Square
Room NE43-813
Cambridge, MA 02139
scaz@ai.mit.edu

Abstract

This report documents the design and implementation of a binocular, foveated active vision system as part of the Cog project at the MIT Artificial Intelligence Laboratory. The active vision system features a 3 degree of freedom mechanical platform that supports four color cameras, a motion control system, and a parallel network of digital signal processors for image processing. To demonstrate the capabilities of the system, we present results from four sample visual-motor tasks.

1 Introduction

The Cog Project at the MIT Artificial Intelligence Laboratory has focused on the construction of an upper torso humanoid robot, called Cog, to explore the hypothesis that human-like intelligence requires human-like interactions with the world (Brooks & Stein 1994). Cog has sensory and motor systems that mimic human capabilities, including over twenty-one degrees of freedom and a variety of sensory systems, including visual, auditory, proprioceptive, tactile, and vestibular senses. This paper documents the design and implementation of a binocular, foveated active vision system for Cog.

In designing a visual system for Cog, we desire a system that closely mimics the sensory and sensori-motor capabilities of the human visual system. Our system should be able to detect stimuli that humans find relevant, should be able to respond to stimuli in a human-like manner, and should have a roughly anthropomorphic appearance. This paper details the design decisions necessary to balance the need for human-like visual capabilities with the reality of relying on current technology in optics, imaging, motor control, as well as with factors such as reliability, cost, and availability.

Three similar implementations of the active vision system described here were produced. The first, shown in Figure 1, is now part of the robot Cog. The second and third implementations, one of which is shown in Figure 2, were constructed as desktop development platforms for active vision experiments.

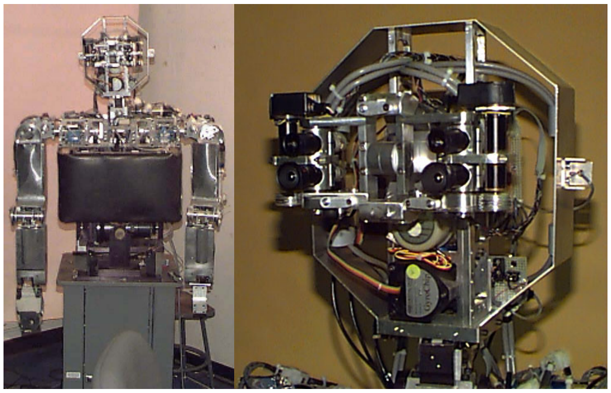


Figure 1: Cog, an upper-torso humanoid robot.

The next section describes the requirements of the active vision system. Sections 3, 4, 5, and 6 provide the details of the camera system, mechanical structure, motion control system, and image processing system used in our implementation. To demonstrate the capabilities of the system, we present four sample visual-motor tasks in Section 7.

2 System Requirements

The active vision system for our humanoid robot should mimic the human visual system while remaining easy to construct, easy to maintain, and simple to control. The system should allow for simple visual-motor behaviors, such as tracking and saccades to salient stimuli, as well as more complex visual tasks such as hand-eye coordination, gesture identification, and motion detection.

While current technology does not allow us to exactly

[*] The author receives support from a National Defense Science and Engineering Graduate Fellowship. Support for the Cog project is provided by an ONR/ARPA Vision MURI Grant (No. N00014-95-1-0600).

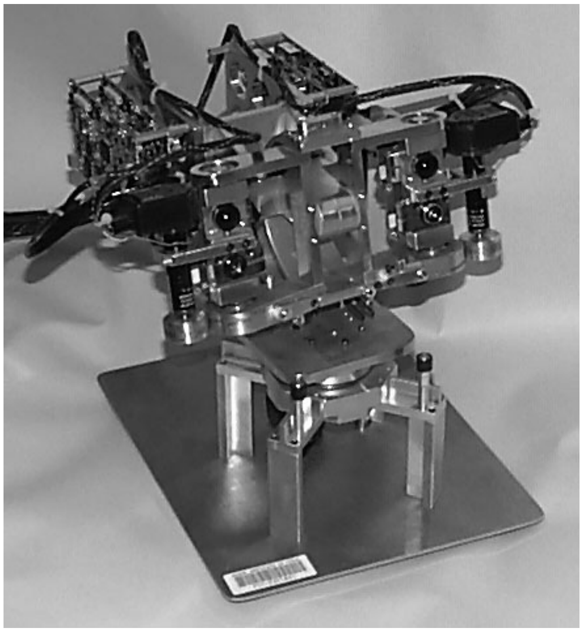


Figure 2: One of the two desktop active vision platforms.

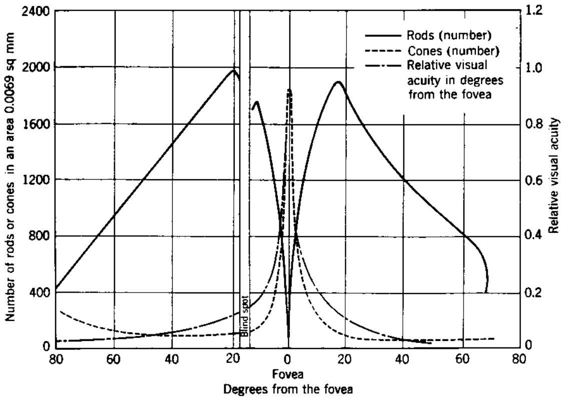


Figure 3: Density of retinal photoreceptors as a function of location. Visual acuity is greatest in the fovea, a very small area at the center of the visual field. A discontinuity occurs where axons that form the optic nerve crowd out photoreceptor cell bodies, resulting in a blind spot. From (Graham 1965).

mimic all of the properties of the human visual system, there are two important properties that we desire: wide field of view and high acuity. Wide field of view is necessary for detecting salient objects in the environment, providing visual context, and compensating for ego-motion. High acuity is necessary for tasks like gesture identification, face recognition, and guiding fine motor movements. In a system of limited resources (limited photoreceptors), a balance must be achieved between providing wide field of view and high acuity. In the human retina, this balance results from an unequal distribution of photoreceptors, as shown in Figure 3. A high-acuity central area, called the fovea, is surrounded by a wide periphery of lower acuity. Our active vision system will also need to balance the need for high acuity with the need for wide peripheral vision.

We also require that our system be capable of performing human-like eye movements. Human eye movements can be classified into five categories: three voluntary movements (saccades, smooth pursuit, and vergence) and two involuntary movements (the vestibulo-ocular reflex and the optokinetic response)(Kandel, Schwartz & Jessell 1992). Saccades focus an object on the fovea through an extremely rapid ballistic change in position (up to 900° per second). Smooth pursuit movements maintain the image of a moving object on the fovea at speeds below 100° per second. Vergence movements adjust the eyes for viewing objects at varying depth. While the recovery of absolute depth may not be strictly necessary, relative disparity between objects are critical for tasks such as accurate hand-eye coordination, figure-ground discrimination, and collision detection. The vestibulo-ocular reflex and the optokinetic response cooperate to stabilize the eyes when the head moves.

The goal of mimicking human eye movements generates a number of requirements for our vision system. Saccadic movements provide a strong constraint on the design of our system because of the high velocities necessary. To obtain high velocities, our system must be lightweight, compact, and efficient. Smooth tracking motions require high accuracy from our motor control system, and a computational system capable of real-time image processing. Vergence requires a binocular system with independent vertical axis of rotation for each eye. The vestibulo-ocular reflex requires low-latency responses and high accuracy movements, but these requirements are met by any system capable of smooth pursuit. The optokinetic response places the least demanding requirements on our system; it requires only basic image processing techniques and slow compensatory movements. [1]

With this set of requirements, we can begin to describe the design decisions that lead to our current implementation. We begin in Section 3 with the choice of the camera system. Once we have chosen a camera system, we can begin to design the mechanical support structures and to select a motor system capable of fulfilling our requirements. Section 4 describes the mechanical requirements, and Section 5 gives a description of the motor control system that we have implemented. If we were to stop at

[1] Implementations of these two reflexes are currently in progress for Cog(Peskin & Scassellati 1997). The desktop development platforms have no head motion, and no vestibular system, and thus do not require these reflexes.

this point, we would have a system with a standard motor interface and a standard video output signal which could be routed to any image processing system. Section 6 describes one of the possible computational systems that satisfies our design constraints which we have implemented with the development platforms and with Cog. In all of these sections, we err on the side of providing too much information with the hope that this document can serve not only as a review of this implementation but also as a resource for other groups seeking to build similar systems.

3 Camera System Specifications

The camera system must have both a wide field of view and a high resolution area. There are experimental camera systems that provide both peripheral and foveal vision from a single camera, either with a variable density photoreceptor array (van der Spiegel, Kreider, Claeys, Debusschere, Sandini, Dario, Fantini, Belluti & Soncini 1989) or with distortion lenses that magnify the central area (Kuniyoshi, Kita, Sugimoto, Nakamura & Suehiro 1995). Because these systems are still experimental, factors of cost, reliability, and availability preclude using these options. A simpler alternative is to use two camera systems, one for peripheral vision and one for foveal vision. This alternative allows the use of standard commercial camera systems, which are less expensive, have better reliability, and are more easily available. Using separate foveal and peripheral systems does introduce a registration problem; it is unclear exactly how points in the foveal image correspond to points in the peripheral image. One solution to this registration problem is reviewed in Section 7.4.

The vision system developed for Cog replaced an earlier vision system which used four Elmo ME411E black and white remote-head cameras. To keep costs low, and to provide some measure of backwards compatibility, we elected to retain these cameras in the new design. The ME411E cameras are 12 V, 3.2 Watt devices with cylindrical remote heads measuring approximately 17 mm in diameter and 53 mm in length (without connectors). The remote head weighs 25 grams, and will maintain broadcast quality NTSC video output at distances up to 30 meters from the main camera boards. The lower camera of each eye is fitted with a 3 mm lens that gives Cog a wide peripheral field of view (88.6°(V) ×115.8°(H)). The lens can focus from 10 mm to ∞. The upper camera is fitted with a 15 mm lens to provide higher acuity in a smaller field of view (18.4°(V) ×24.4°(H)). The lens focuses objects at distances from 90 mm to ∞. This creates a fovea region significantly larger than that of the human eye, which is 0.3°, but which is significantly smaller than the peripheral region.

For the desktop development platforms, Chinon CX-062 color cameras were used.[2] These cameras were considerably less expensive than the Elmo ME411E models, and allow us to experiment with color vision. Small remote head cameras were chosen so that each eye is compact and lightweight. To allow for mounting of these cameras, a 3 inch ribbon cable connecting the remote head and the main board was replaced with a more flexible cable. The upper cameras were fitted with 3 mm lenses to provide a wide peripheral field of view. The lower cameras were fitted with 11 mm lenses to provide a narrow foveal view. Both lenses can focus from 10 mm to ∞. The CX-062 cameras are 12 V, 1.6 Watt devices with a remote board head measuring 40 mm (V) × 36 mm (H) × 36 mm (D) and a main camera board measuring 65 mm × 100 mm with a maximum clearance of 15 mm. The CX-062 remote heads weight approximately 20 grams, but must be mounted within approximately .5 meters from the main camera boards.

4 Mechanical Specifications

The active vision system has three degrees of freedom (DOF) consisting of two active "eyes". Each eye can independently rotate about a vertical axis (pan DOF), and the two eyes share a horizontal axis (tilt DOF). These degrees of freedom allow for human-like eye movements.[3] Cog also has a 3 DOF neck (pan, tilt, and roll) which allows for joint pan movements of the eyes. To allow for similar functionality, the desktop platforms were fitted with a one degree of freedom neck, which rotates about a vertical axis of rotation (neck pan DOF). To approximate the range of motion of human eyes, mechanical stops were included on each eye to permit a 120° pan rotation and a 60° tilt rotation.

To minimize the inertia of each eye, we used thin, flexible cables and chrome steel bearings.[4] This allows the eyes to move quickly using small motors. For Cog's head, which uses the Elmo ME411E cameras, each fully assembled eye (cameras, connectors, and mounts) occupies a volume of approximately 42 mm (V) × 18 mm (H) × 88 mm (D) and weighs about 130 grams. For the development platforms, which use the Chinon CX-062 cameras, each fully assembled eye occupies a volume of

[2] In retrospect, this choice was unfortunate because the manufacturer, Chinon America, Inc. ceased building all small-scale cameras approximately one year after the completion of this prototype. However, a wide variety of commercial remote-head cameras that match these specifications are now available.

[3] Human eyes have one additional degree of freedom; they can rotate slightly about the direction of gaze. You can observe this rotation as you tilt your head from shoulder to shoulder. This additional degree of freedom is not implemented in our robotic system because the pan and tilt DOFs are sufficient to scan the visual space.

[4] We used ABEC-1 chrome steal bearings (part # 77R16) from Alpine Bearings.

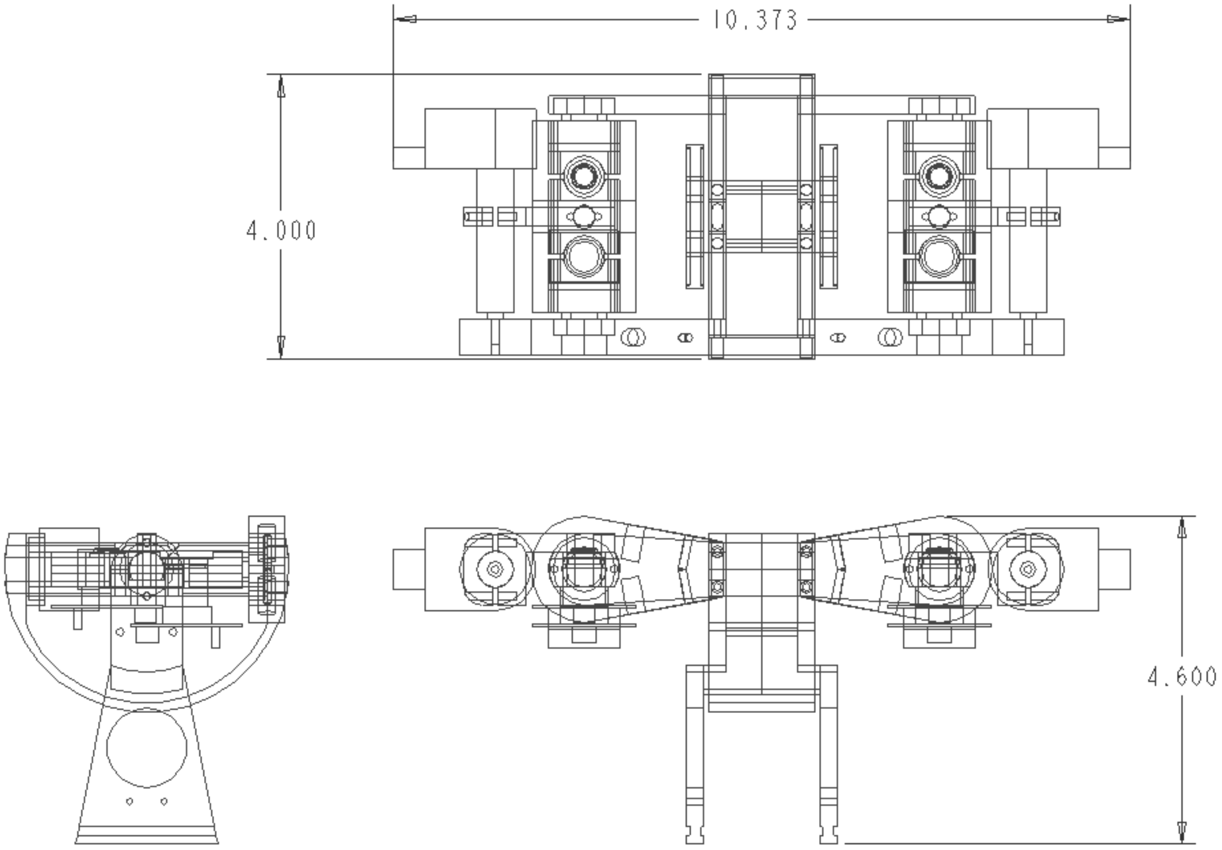


Figure 4: Three orthographic projections of the mechanical schematics of the desktop active vision system. All measurements are in inches.

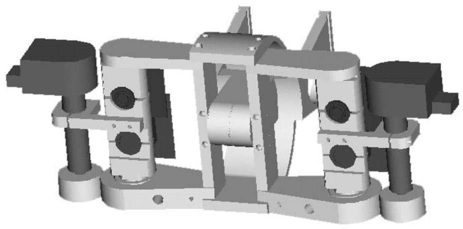


Figure 5: Rendering of the desktop active vision system produced from the engineering drawings of Figure 4.

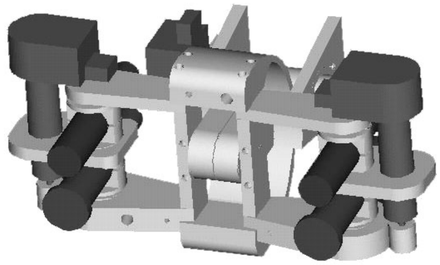


Figure 6: Rendering of Cog's active vision system. Different cameras produce slightly different mechanical specifications, resulting in a more compact, but heavier eye assembly.

approximately 70 mm (V) × 36 mm (H) × 40 mm (D) and weighs about 100 grams. Although significantly heavier and larger than their human counterpart, they are smaller and more lightweight than other active vision systems (Ballard 1989, Reid, Bradshow, McLauchlan, Sharkey, & Murray 1993).

The mechanical design and machining of the vision systems were done by Cynthia Ferrell, Elmer Lee, and Milton Wong. Figure 4 shows three orthographic projections of the mechanical drawings for the desktop development platform, and Figures 5 and 6 show renderings of both the desktop platform and the system used on Cog. The implementation of the initial Cog head prototype and the development platforms were completed in May of 1996.

5 Eye Motor System Specifications

Section 2 outlined three requirements of the eye motor system. For Cog's visual behaviors to be comparable to human capabilities, the motor system must be able to move the eyes at fast speeds, servo the eyes with fine position control, and smoothly move the eyes over a wide range of velocities.

On average, the human eye performs 3 to 4 full range saccades per second(Kandel et al. 1992). Given this goal, Cog's eye motor system is designed to perform three 120° pan saccades per second and three 60° tilt saccades per second (with 250 ms of stability in between saccades). This specification corresponds to angular accelerations of $1309\frac{radians}{s^2}$ and $655\frac{radians}{s^2}$ for pan and tilt.

To meet these requirements, two motors were selected. For the pan and tilt of the Cog prototype and for the neck pan and tilt on the desktop systems, Maxon 12 Volt, 3.2 Watt motors with 19.2:1 reduction planetary gearboxes were selected. The motor/gearbox assembly had a total weight of 61 grams, a maximum diameter of 16 mm and a length of approximately 60 mm. For the desktop development platforms, it was possible to use smaller motors for the pan axis. We selected Maxon 12 Volt, 2.5 Watt motors with 16.58:1 reduction planetary gearboxes. This motor/gearbox assembly had a total weight of 38 grams, a maximum diameter of 13 mm and a total length of approximately 52 mm.[5]

To monitor position control, each motor was fitted with a Hewlett-Packard HEDS-5500 optical shaft encoder. The HEDS-5500 has a resolution of 1024 counts per revolution. The motor/gearbox/encoder assembly was attached to the load through a cable transmission system. By modifying the size of the spindles on the cable transmission, it was possible to map one full revolution of the motor to the full range of motion of each axis. This results in an angular resolution of 8.5 encoder ticks/degree for the pan axis and 17 encoder ticks/degree for the tilt axis.

The motors were driven by a set of linear amplifiers, which were driven by a commercial 4-axis motor controller (see Figure 7).[6] This motor controller maintained a 1.25 kHz servo loop at 16 bits of resolution for each axis. The motor controller interfaced through the ISA bus to a PC and provided a variety of hardware supported motion profiles including trapezoidal profiles, S-curve acceleration and deceleration, parabolic acceleration and deceleration, and constant velocity moves.

6 Computational Specifications

To perform a variety of active vision tasks in real time, we desire a system that is high bandwidth, powerful, and scalable. The system must have enough bandwidth to handle four video streams at full NTSC resolution, and be powerful enough to process those streams. Ideally, the system should also be easily scalable so that additional processing power can be integrated as other tasks are required.

6.1 *Parallel Network Architecture*

Based on these criteria, we selected a parallel network architecture based on the TIM-40 standard for the Texas Instruments TMS320C40 digital signal processor. The TIM-40 standard allows third-party manufacturers to produce hardware modules based around the C40 processor that incorporate special hardware features but can still be easily interfaced with each other. For example, one TIM-40 module might have specialized hardware for capturing video frames while another might have special hardware to perform convolutions quickly. Distributed computation is feasible because modules communicate with each other through high-speed bi-directional dedicated hardware links called comports, which were designed to carry full size video streams or other data at 40 Mbits/second. Depending on the module, between 4 and 6 comports are available. Additional computational power can easily be added by attaching more TIM-40 modules to the network. Each TIM-40 module connects to a standardized backplane that provides power and support services. The entire network interfaces to a PC through an ISA card (in our system, we use the Hunt Engineering HEP-C2 card).

Figure 8 shows both the general network architecture, and the specific TIM-40 modules that are currently attached to one of the development platforms. In this network, four types of TIM-40 module are used.[7] The first module type is a generic C40 processor with no additional capabilities. In this network, the two nodes labeled "ROOT" and "P2" are both generic processors. The "ROOT" node is special only in that one of its comports is dedicated to communications to the host computer. The second module type, labeled "VIP", for "Visual Information Processor", contains dedicated hardware to quickly compute convolutions. The third module type, labeled "AGD", or "Accelerated Graphics Display", has hardware to drive a VGA monitor. This module is very useful for displaying processed images while debugging. The fourth module type has hardware to grab frames from an incoming video signal. The four instances of this module are labeled "Right Wide", "Right Fovea", "Left Fovea",

[5] The 3.2 Watt Maxon motor is part # RE016-039-08EAB100A and its gearbox is part # GP016A019-0019B1A00A. The 2.5 Watt motor is part # RE013-032-10EAB101A and its gearbox is part # GP013A020-0017B1A00A.

[6] The linear amplifiers are model TA-100 amps from Trust Automation. The motor controller is an LC/DSP-400 4-axis motor controller from Motion Engineering, Inc.

[7] The four module types are sold by Traquair Data Systems, Inc., with catalog numbers HET40Ex, VIPTIM, AGD, and HECCFG44 respectively.

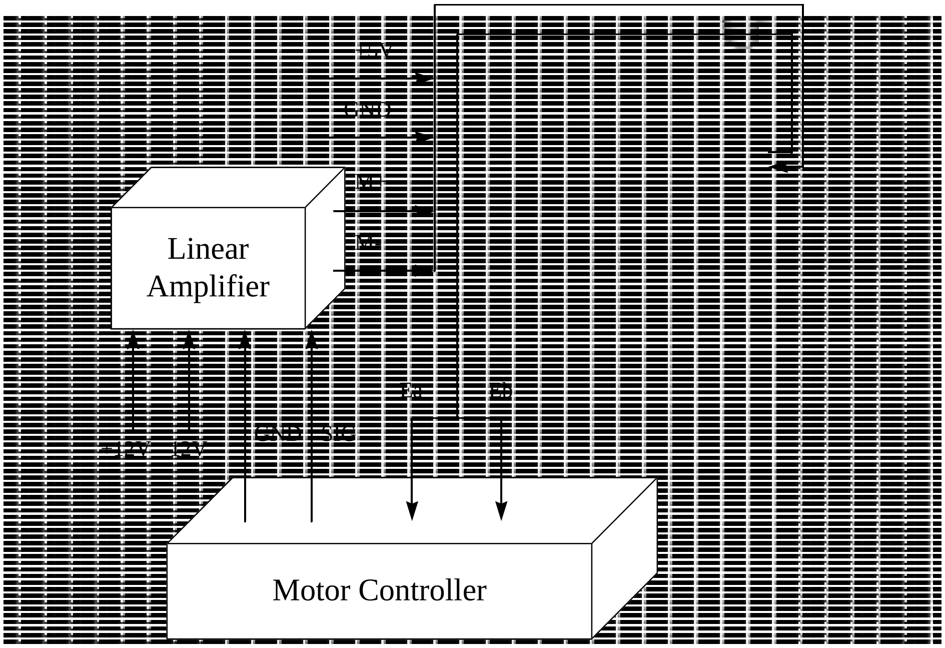


Figure 7: Schematic for the electrical wiring of the motor subsystem. The motor control signal (SIG) drives a linear amplifier, which produces a differential pair of amplified signals (M+ and M-). Two encoder channels (Ea and Eb) return feedback from the motor assembly.

and “Left Wide” in the figure. Connections between processors are shown by single lines. Because the number of comports are limited, the connectivity in the network is asymmetric. As we will see in the next section, this only presents a minor problem to programming, since virtual connectivity can be established between any two processors in the network.

6.2 Software Environment

To take advantage of the high-speed interprocessor connections in the C40 network, we use a commercial software package called *Parallel C* from 3L, Ltd. *Parallel C* is a multi-threading C library and runtime system which essentially creates a layer of abstraction built upon the ANSI C programming language. *Parallel C* consists of three main parts:

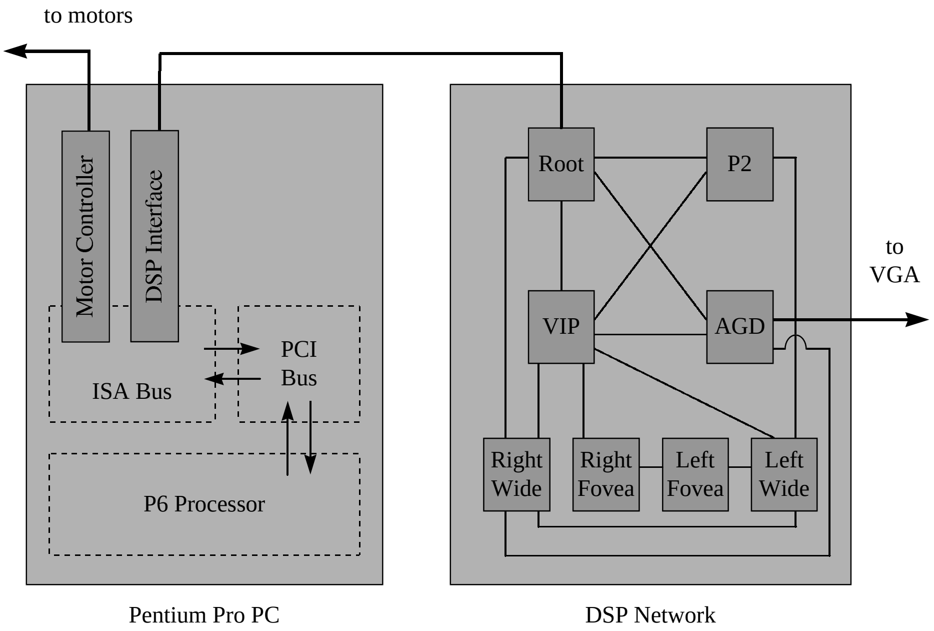
- Runtime libraries and compiler macros, which provide routines for multi-threading and interprocessor communication, as well as standard ANSI C functions.
- A microkernel, running on each C40 node, which handles multitasking, communication, and transparent use of I/O throughout a network.
- A host server, running on the PC, which handles the front-end interface to the C40 network, including downloading applications and providing a standard input and output channels.

Compiling and linking are done with the Texas Instruments C compiler.

Parallel C also provides facilities for connecting tasks on processors that do not share a physical comport connection through the use of virtual channels. Virtual channels are one-way data streams which transmit data from an output port to an input port in an in-order, guaranteed way. A channel might be mapped directly to a physical comport connection or it might travel through several nodes in the network, but both cases can be treated identically in software. The microkernels on each processor automatically handle virtual channels, ensuring that data gets from one task’s output port to another task’s input port, as long as some chain of available physical comport connections exists.

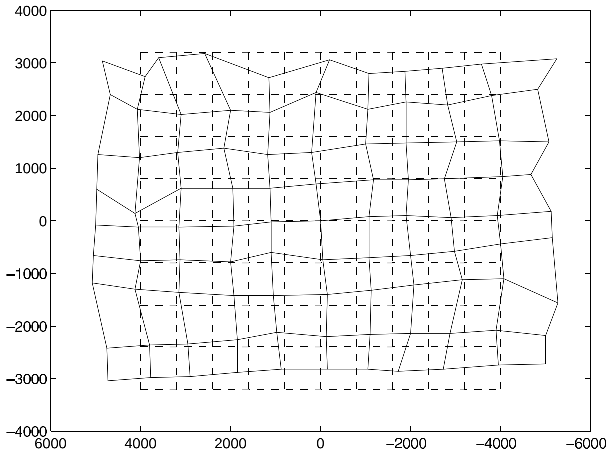
7 Example Tasks

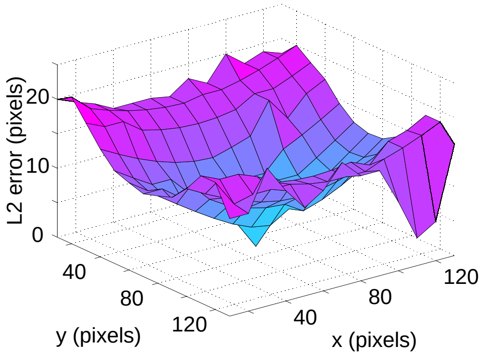
A number of research projects have made use of these active vision platforms (Marjanović, Scassellati &



× 17 interpolated lookup table to estimate the saccade function. We are currently completing a comparative study between various machine learning techniques on this task (Banks & Scassellati 1997).

Saccade map training begins with a linear estimate based on the range of the encoder limits (determined during self-calibration). For each learning trial, we generate a random visual target location (x_t, y_t) within the 128 × 128 image array and record the normalized im-



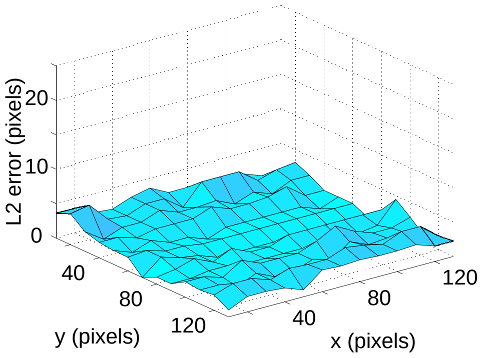


L_2 error for saccades to image positions (x,y) after 0 training trials.

age intensities $\overline{I_t}$ in a 13 × 13 patch around that point. The reduced size of the image array allows us to quickly train a general map, with the possibility for further refinement after the course mapping has been trained. Once the random target is selected, we issue a saccade motor command using the current map estimate. After the saccade, a new image $\overline{I_{t+1}}$ is acquired. The normalized 13 × 13 center of the new image is then correlated against the target image. Thus, for offsets x_0 and y_0, we seek to maximize the dot-product of the image vectors:

$$\max_{x_0,y_0}\left(\sum_i \sum_j \overline{I_t}(i,j)\cdot \overline{I_{t+1}}(i+x_0,j+y_0)\right) \qquad (1)$$

Because each image was normalized, maximizing the dot product of the image vectors is identical to minimizing the angle between the two vectors. This normalization also gives the algorithm a better resistance to changes in background luminance as the camera moves. In our experiments, we only examine offsets x_0 and y_0 in the range of $[-32,32]$. The offset pair that maximized the expression in Equation 1, scaled by a constant factor, is used as the error vector for training the saccade map.



L_2 error for saccades to image positions (x,y) after 2000 training trials.

Figure 9 shows the data points in their initial linear approximation (dashed lines) and the resulting map after 2000 learning trials (solid lines). The saccade map after 2000 trials clearly indicates a slight counter-clockwise rotation of the mounting of the camera, which was verified by examination of the hardware. Figure 10 shows the L_2 error distance for saccades after 0 learning trials. After 2000 training trials, an elapsed time of approximately 1.5 hours, training reaches an average L_2 error of less than 1 pixel (Figure 11). As a result of moving objects during subsequent training and the imprecision of the correlation technique, this error level remained constant regardless of continued learning.

7.2 *Saccades to Motion Stimuli*

By combining the saccade map with visual processing techniques, simple behaviors can be produced. To demonstrate this, we provide here a simple example using visual motion as a saliency test. Any more complex evaluation of saliency can easily be substituted using this simple formulation.

A motion detection module computes the difference between consecutive wide-angle images within a local field. A motion segmenter then uses a region-growing technique to identify contiguous blocks of motion within the difference image. The centroid of the largest motion block is then used as a saccade target using the trained saccade map from section 7.1.

The motion detection process receives a digitized 64 × 64 image from the right wide-angle camera. Incoming images are stored in a ring of three frame buffers;

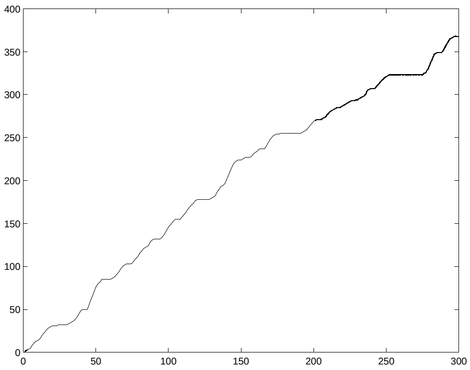
one buffer holds the current image I_0, one buffer holds the previous image I_1, and a third buffer receives new input. The absolute value of the difference between the grayscale values in each image is thresholded to provide a raw motion image ($I_{raw} = \mathcal{T}(|I_0 - I_1|)$). The difference image is then segmented using a region-growing technique. The segmenter process scans the raw motion image marking all locations which pass threshold with an identifying tag. Locations inherit tags from adjacent locations through a region grow-and-merge procedure. Once all locations above threshold have been tagged, the tag that has been assigned to the most locations is declared the "winner". The centroid of the winning tag is computed, converted into a motor command using the saccade map, and sent to the motors.

7.3 Smooth Pursuit Tracking

While saccades provide one set of requirements for our motor system, it is also necessary to examine the performance of the system on smooth pursuit tracking. [8] Our example of smooth pursuit tracking acquires a visual target at startup and attempts to maintain the foveation of that target.

The central 7 × 7 patch of the initial 64 × 64 image is installed as the target image. In this instance, we use a very small image to reduce the computational load necessary to track non-artifact features of an object. For each successive image, the central 44 × 44 patch is correlated with the 7 × 7 target image. The best correlation value gives the location of the target within the new image, and the distance from the center of the visual field to that location gives the motion vector. The length of the motion vector is the pixel error. The motion vector is scaled by a constant (based on the time between iterations) and used as a velocity command to the motors.

While simple, this tracking routine performs well for smoothly moving real-world objects. Figure 12 shows the cumulative pixel error while tracking a mug moving continuously in circles in a cluttered background for ten seconds. An ideal tracker would have an average pixel error of 1, since the pixel error is recorded at each timestep and it requires a minimum of one pixel of motion before any compensation can occur. In the experiment shown here, the average pixel error is 1.23 pixels per timestep. (This may result from diagonal movements of the target between consecutive timesteps; a diagonal movement results in a pixel error of $\sqrt{2}$.) This example demonstrates that the motor system can respond quickly enough to track smoothly.



L_2 pixel error accumulated while tracking a continuously moving object. There are thirty timesteps per second.

7.4 Registering the Foveal and Peripheral Images

Using two cameras for peripheral and foveal vision allows us to use commercial equipment, but results in a registration problem between the two images. We would like a registration function that describes how the foveal image maps into the peripheral image, that is, a function that converts positions in the foveal image into positions in the peripheral image. Because the foveal image has a small aperture, there is little distortion and the image linearly maps to distances in the environment. The peripheral image is non-linear near the edges, but was determined to be relatively linear near the center of the field of view (see section 7.1). Because the relevant portions of both images are linear, we can completely describe a registration function by knowing the scale and offsets that need to be applied to the foveal image to map it directly into the peripheral image.

One solution to this problem would be to scale the foveal image to various sizes and then correlate the scaled images with the peripheral image to find a corresponding position. By maximizing over the scale factors, we could determine a suitable mapping function. This search would be both costly and inexact. Scaling to non-integer factors would be computationally intensive, and exactly how to perform that scaling is questionable. Also, arbitrary scaling may cause correlation artifacts from features that recur at multiple scales.

Another alternative is to exploit the mechanical system to obtain an estimate of the scale function. Since both cameras share the pan axis, by tracking the background as we move the eye at a constant velocity we can determine an estimate of the scale between cameras. With the eye panning at a constant velocity, separate processors for the foveal and peripheral images track the background, keep-

[8] Given saccades and smooth pursuit, vergence does not place any additional requirements on the responsiveness of the motor system.

ing an estimate of the total displacement. After moving through the entire range, we estimate the scale between images using the following formula:

$$Scale_{pan} = \frac{Displacement_{peripheral}}{Displacement_{foveal}} \quad (2)$$

While the tilt axis does not pass through the focal points of both cameras, we can still obtain a similar scaling factor for the tilt dimension. Because we average over the entire field, and do not compare directly between the foveal and peripheral images, a similar equation holds for the tilt scaling factor. Once the scaling factor is known, we can scale the foveal image and convolve to find the registration function parameters.

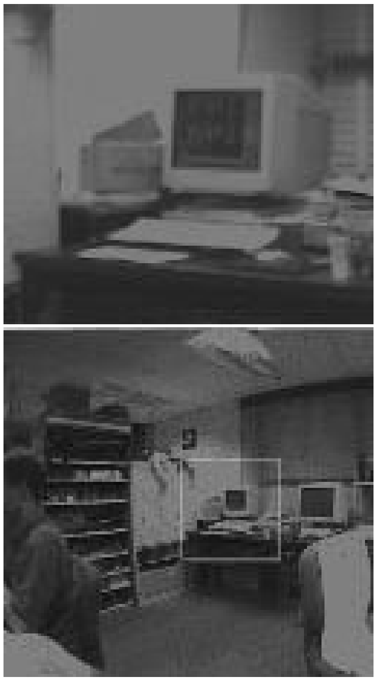


Figure 13: Registration of the foveal and peripheral images. The foveal image (top) correlates to a patch in the 128 × 128 peripheral image (bottom) that is approximately one-fourth scale and at an offset of 2 pixels above and 14 pixels right from the center.

We have experimentally determined the registration function parameters for the desktop development platform using this method. Over a series of ten experimental trials using the above method, the average scale factor for both the pan and tilt dimension were both determined to be 4.0, with a standard deviation of .1. The scaled foveal image was best located at a position 2 pixels above and 14 pixels from the center of the 128 × 128 peripheral image (see Figure 13). As a control, the same experiment produced on the cameras of the other eye produce exactly the same scaling factor (which is a product of the camera and lens choices), but different offset positions (which are a result of camera alignment in their respective mounts).

8 Conclusions

This report has documented the design and construction of a binocular, foveated active vision system. The vision system combines a high acuity central area and a wide peripheral field by using two cameras for each eye. This technique introduces a registration problem between the camera images, but we have shown how simple active vision techniques can compensate for this problem. We have also presented a number of sample visual behaviors, including adaptive saccading, saccades to salient stimuli, and tracking, to demonstrate the capabilities of this system.

9 Acknowledgments

Elmer Lee and Milton Wong designed and constructed the mechanical platform for the active vision systems. Cynthia Ferrell, Matt Marjanovic, and Matt Williamson contributed to both the hardware and software designs.

The author also wishes to thank the other members of the Cog group (past and present) for their continual support: Rod Brooks, Robert Irie, Jonah Peskin, and Lynn Stein.

References

Ballard, D. (1989), ‘Behavioral Constraints on Animate Vision’, *Image and Vision Computing* **7:1**, 3–9.

Banks, B. S. & Scassellati, B. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Learning Visual-Motor Tasks: A Comparison Study.

Brooks, R. & Stein, L. A. (1994), ‘Building Brains for Bodies’, *Autonomous Robots* **1:1**, 7–25.

Ferrell, C. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Learning Social Behaviors in an Altricial Context.

Graham, C. H. (1965), *Vision and Visual Perception*, John Wiley and Sons, Inc.

Irie, R. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Multimodal Sensory Integration for a Humanoid Robot.

Kandel, E. R., Schwartz, J. H. & Jessell, T. M., eds (1992), *Principles of Neural Science*, Appleton and Lange, chapter chapter title.

Kemp, C. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter A Platform for Visual Learning.

Kuniyoshi, Y., Kita, N., Sugimoto, K., Nakamura, S. & Suehiro, T. (1995), A Foveated Wide Angle Lens for Active Vision, *in* 'Proc. IEEE Int. Conf. Robotics and Automation'.

Marjanović, M., Scassellati, B. & Williamson, M. (1996), Self-Taught Visually-Guided Pointing for a Humanoid Robot, *in* 'Society of Adaptive Behavior'.

Peskin, J. & Scassellati, B. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Image Stabilization through Vestibular and Retinal Feedback.

Reid, I., Bradshow, K., McLauchlan, P., Sharkey, P., & Murray, D. (1993), From Saccades to Smooth Pursuit: Real-Time Gaze Control using Motion Feedback, *in* 'International Conference on Intelligent Robots and Systems', Yokahama, Japan, pp. 1013–1020.

Scassellati, B. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Mechanisms of Shared Attention for a Humanoid Robot.

van der Spiegel, J., Kreider, G., Claeys, C., Debusschere, I., Sandini, G., Dario, P., Fantini, F., Belluti, P. & Soncini, G. (1989), *A foveated retina-like sensor using CCD technology*, Kluwer Academic Publishers.

Yamato, J. (1997), *Research Abstracts*, MIT Artificial Intelligence Laboratory, chapter Learning Pointing Action in 3D space using depth information from stereo cameras.