Primitive Parallax and Parallax Primitives

Joanna Bryson
MIT NE43-825
545 Technology Square
Cambridge MA 02139, USA
(617) 253–1299, fax 253–5060
joanna@ai.mit.edu

April 8, 1995

Abstract

This paper describes an implementation of parallax depth perception designed to be a contributing unit to a distributed perceptual intelligence. The work was done in the context of the Cog project, a large-scale effort to develop human or at least infant intelligence from the bottom up. Our implementation uses fast-approximation "good enough" region mapping and relevant motion detection over multiple observations to build and revise an approximate depth map. This map is intended to be an input for higher-level perception tasks such as object discrimination and attention to change. In turn, our model of parallax is also dependent on preexisting attention mechanisms that guarantee the ability to follow a target and notice independent motion.

Keywords:
Active Vision, Low-Level And High-Level Perception, Distributed Intelligence, Behavior Integration

1 Introduction

The problem with vision (and intelligence in general) is that although we think we know some of the components of the task, when it comes to implementing a functional system, we can't decide where to start. All of the pieces are interdependent. For an example, consider depth perception. Clearly we can use depth information to disambiguate outlines and distinguish between objects, but if we define parallax as relative motion between objects, don't we need to disambiguate the objects first?

This paper describes an implementation of single-camera motion parallax depth perception designed to be a contributing unit to a distributed perceptual intelligence. The work was done in the context of the Cog project, a large-scale effort to develop human or at least infant intelligence from the bottom up. Our implementation of parallax is not, in fact, dependent on pre-identified objects or symbolic reasoning. Instead, it uses fast-approximation "good enough" region mapping and multiple observations to build and revise an approximate depth map. This map is intended to be an input for higher-level perception tasks such as object discrimination and attention to change. In turn, our model of parallax is also dependent on prexisting attention mechanisms that guarantee the ability to follow a target and notice independent motion.

This paper begins with a discussion of the context in which this work was done, from the perspective of vision research and as an element of Cog. Next we present our approach and preliminary results. Finally, we discuss our current research — the task of integrating this piece into a real-time, distributed, behavior-based intelligent system.

2 Research Context

2.1 Depth Perception in Animals

Most research in depth perception focuses on stereo vision, which is taken essentially to be image differencing between two eyes. However, for humans at least by far the most important indicator of depth is occlusion. That is, if one object obscures part of another object, we perceive the former object to be closer to us. We are also very dependent on environmental cues such as scaling caused by reduction to a vanishing point – this has been used to great effect in optical illusions where artificial scaling makes even a familiar person standing at the "small" end of the room appear gigantic as we assume they must be far away.

Although the above depth-perceiving techniques are a seamless part of our visual experience, we are probably not born with these abilities, but rather acquire them as expertise. Recent research in peripheral vision indicates that even as adults we can quickly acquire new skills in unfamiliar motion perception tasks, even *without* feedback.(Sundareswaran & Vaina 1994) We are somehow able not only to acquire these skills, but to integrate them into the high-level perceptual experience.[1]

[1] There is now evidence for motor skills that acquired "reflexive" skills are actually stored in the same part of the brain as the "built-in" reflexes.(Barinaga 1994)

Even disregarding occlusion, motion parallax is more important to human depth perception than binocular disparity, which is not surprising considering that over 10% of the population is incapable of the latter. Stereo vision is most important in the case of detecting the depth of an object with relatively few environmental cues, such as a ball coming directly towards you. This task is particularly relevant in frogs and toads, where they need to plot the exact location of a target prey in the sky. Even here, research has shown that these animals are surprisingly competent with only one eye(Arbib & House 1987). Even using two eyes, frogs can function normally with lesions disconnecting the two hemispheres of their brains. This evidence indicates that it is unlikely that image image-differencing between the eyes is taking place. Rather, some monocular technique (possibly lens focus) seems to be simply triangulated. Triangulation can be done by look-up, with the look-up table modified and refined by experience. This would be similar to the method of hearing localization described for owls in (Gallistel, Brown, Carey, Gelman & Keil 1991). Or the triangulation could simply be achieved at the motor level, by having both eyes drive the muscles simultaneously.

2.2 Machine Vision

Most Artificial Intelligence work in depth perception has also been in binocular image differencing. Where motion parallax is used, the technique tends to be similar. For an example, see (Barnard & Thompson 1980). The theory is to match as many features as possible between two images, then to mathematically compute the depths of the matched points and extrapolate the depths of the entire image. This approach of determining perfect information from a single technique on a single pair of images is typical of what is now called "classical vision." The approach tends to be computationally expensive, and prone to pitfalls such as failures to match, or erroneous extrapolation assumptions. Because of these difficulties, the problem of using vision for real-time problems such as robot navigation and object avoidance was seen as computationally intractable until very recently.

The change that has made robot navigation doable has been the advent of a new philosophy known as "active vision." This approach has proved a fertile ground for for vision research (see (Blake & Yulle 1992)). It differs from classical vision in two important ways. First, it contests the assumption that in order for vision to be useful it must result in the construction of a perfect representation of the viewed scene. We now know that insect visual systems tend to be highly biased to only pick up the exact information needed to coordinate behavior. Also, human perception research shows that our higher-level perception is a constructive process much like memory – we have holes in our awareness which are only filled in when we attend to a particular feature, and even then can be filled in incorrectly(Dennett 1991). Instead, active vision concentrates on providing input that is "good enough" to get the desired behavioral output, and tends to leave the issue of representation alone. This parallels the development of decentralized, non- or reduced-representational strategies in general in artificial intelligence outlined in (Brooks 1991).

The second major difference in active vision is that vision is not seen as a static task – no single bout of processing is responsible for the analysis of a scene. Rather, visual information is a stream that may contain noise and inaccuracies, but can be compensated for over time or by fusing this stream with information from other processing modules. This change in paradigm is wedded to the intent of the first assumption – vision as a stream of input feeds behavior as a continuous stream of output, with no particular moment when "decisions" or other reasoning events occur.

Perhaps the most exciting demonstration has been the success of vision navigation for high-speed robots in office environments. Using techniques described in (Horswill 1993), Horswill has robots that using only vision for object avoidance, navigating at speeds up to 2 m/s (though in the vicinity of obstacles speed reduces to 1 m/s) These speeds are achieved using entirely on-board processing – in fact, the entire vision system, cameras and processors combined, costs less than $2000.

Horswill's methodology is exemplary of active vision. His robot Polly had a camera angled towards the floor in such a way that it could see the floor directly before it, but could also see the floor ahead rising higher in the visual plane. Assuming the floor directly in front of the robot is clear, then any obstacle on the floor would show up as an edge in contrast to the carpet. Thus, distance to an obstacle is proportional to the height in the visual plane of the edge between the floor and its base. Since Polly was only trying to choose a clear path, the only information it needed to compute is whether the furthest edge from it was directly in front of it, to its right, or to its left. It computed this simply as the height of columns of uninterrupted floor in it's visual field. This very simplistic version of depth perception is completely adequate for the role it plays in Horswill's robots' navigation. There are separate units that maintain overall navigational goals and such specialized tasks as following corridors smoothly.

2.3 The Humanoid Project

The most direct context for our research is that of the Humanoid Project currently being conducted at MIT. The goals and visual apparatus of the project are more fully described in (Ferrell, Scassellati & Binnard 1995), but fundamentally the aim of the project is to use the new approaches to AI and vision described above to construct a robot with something like the intelligence of a six-month-old child. This is a large-scale project which includes psychologists and philosophers of mind[2], but is at heart a robotics project. The robot, Cog, is humanoid from the waist up, with similar degrees of freedom and similar size to an MIT graduate student. Its vision system consists of two eyes, which each consist of two cameras, one for foveal (detailed) vision and the other for peripheral vision. The foveal cameras have a field of view of about 20 degrees; the peripheral cameras field is about 100 degrees. Images from the cameras are processed by frame grabbers into 128x128 arrays of pixels valued from 0 to 255 by intensity. Cog currently sees only in black and white, but other experimental heads for Cog such as Odie are using color vision.

[2] for a full list of project members, please see our Web page at http://www.ai.mit.edu/projects/cog/cog-people.html

Cog's processing is done on Motorola 68332s arranged to run in parallel on backplanes of up to 15 processors. There can theoretically be any number of backplanes, right now there is only one. There is no shared memory between processors – communication will be limited to image arrays as described previously or simple integer streams. The processors are sparsely connected with only 6 cabling ports per processor. The design for the project is to have the individual processors be dedicated to particular cognitive tasks, like the different regions of the brain. For example, each camera has a dedicated processor. The processors run a multiprocessing version of Common Lisp called L(Brooks 1993). There are also peripheral motor and sensor boards based on the Motorola 6811 which daisy chain in series off individual processors. These motor boards provide a sort of distributed spinal chord for local reflexive behavior and sensing. It should be noted that there is some controversy over whether 68332's have the processing power for even simple vision algorithms such as Horswill's described above – the experimental Cog head Odie is using boards built around Texas Instruments C30 Digital Signal Processors. Cog will probably be using a hybrid 68332/DSP technology soon.

3 Methodology and Preliminary Results

Assuming the approach of active vision, the first step in designing a parallax vision system for Cog is to establish the desired output of the module. The goals for this unit are the following:

1. To provide depth information to a "background" module – that is, a module that would recognize the persistent structure of the visual environment in order to facilitate focusing attention on "important" (novel) things in the environment. This unit would be one of the building-blocks for higher-level perception and memory as described above.

2. To provide depth information to help to discriminate adjacent regions of color as single or multiple objects.

The inputs to the unit are the image arrays described above (section 2.3). Also, a vector of the motion the camera made between the two images is provided. The source of this vector is discussed below.

The form of the output is the same sort of array as an input image. However, for the output the individual array units represent not pixels in a camera's field of view, but the depth of a larger region, so that the array can represent the entire area surrounding Cog. The reason for using the same structure for output as input is twofold. First, as already mentioned, there are communication primitives for moving images between processors. Second, one of the assumptions of the Cog project is that knowledge acquired in one domain can be generalized to the next. This will be easiest to realize if it is true on a strictly functional level – if the exact same processes can be applied to data representing different types of information because that information is held in the same data structure.

The fundamental constraint of any vision primitive for real-time processing is that it must be very, very fast. This basically requires that each unit of information, each pixel in the array, be visited very few times. It also requires that no expensive memory-allocation be performed – that is, the process must run within fixed memory space. Our parallax algorithm consists of two phases The first finds regions in each image using two passes through each array. The second phase finds motion by doing a single pass through the first image array finding edges, with a quick localized search for the corresponding edge in the second array whenever an edge is found. This algorithm currently takes 300K and about six seconds to run on two full 128x128 images on Cog's 68332's or their Macintosh equivalents. The next stage of our research will be to significantly optimize this runtime by eliminating abstraction barriers, increasing function efficiency, changing platforms to the C30, and finally reducing the processed frame size. This approach has already been used successfully for Cog's tracking and motion-detection units(Wessler 1995).

The following subsections describe the algorithm in more detail.

3.1 Region Finding

Region finding was developed from scratch, but is fairly similar to the classical methodologies. Region finding is done by passing through the array from top to bottom, left to right, checking if each pixels neighbors are within a certain threshold of itself. If it is, the pixel is marked as being the same as the neighbor's. Otherwise, the pixel is tagged as a new region. Since the progress through the array is systematic, we can search only half the neighbors of the target pixel and assume the other half will be searched reciprocally when we get to the pixel's unsearched neighbor. (If all pixels searched all their neighbors, then every possible pair would be searched twice.) We search only the upper three and left neighbors, that is, only the neighbors whose regions have already been assigned, so that we have the regions' tag provided if we find a match.

If we find a match with more than one neighbor, we need to make certain that both of these neighbors are considered to be in the same region. This accounts for the second pass through the array for region matching above – all regions that were found to be matched are set to the same value tag.

In general, if the difference threshold between two regions is set too high, then too many unrelated regions are linked as one. If the threshold is set too low, then regions are broken into random noise. Initially we tried checking only half of a pixel's neighbors, those directly above and beside it, but this resulted in the threshold for preventing artificial segmentation being higher than the threshold for artificial region unification. Checking all four neighbors however had the opposite effect of making all regions bleed into one. We determined that this could be caused by a single pixel of noise fusing two nearby regions, so changed the requirement to having two adjacent neighbors of the same region as the target pixel. This gave us an acceptable threshold value of 4 on an intensity variance scale of 256 levels. Note that this still neither discriminates every region in every frame, nor resolves regions of rapid pixel fluctuation such as the vertical striping of heat vents on the backs of monitors. Areas where segmentation varies by pixel look like striping in the pictures, because the regions are assigned sequentially, and then the images are produced by taking the modulo of the region's tag with 256. What appears to be striping is simply slow gradation.

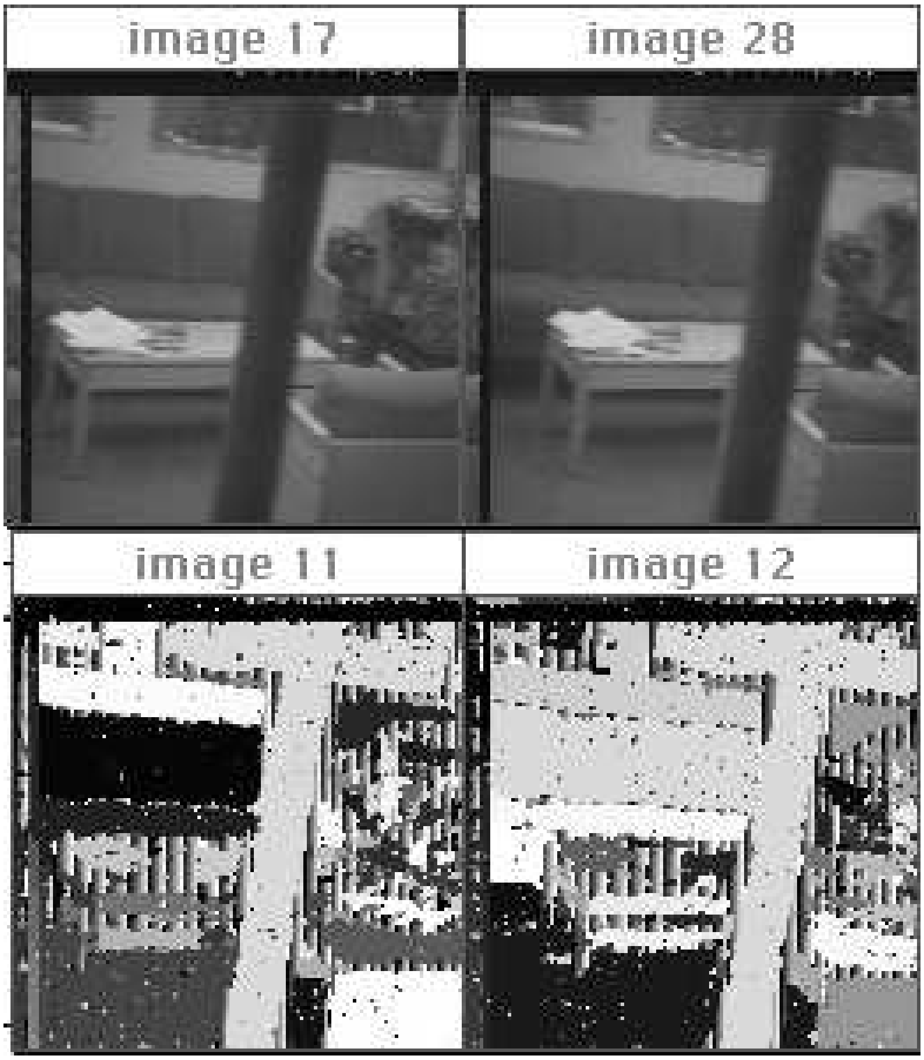

Figure 1: Two images for parallax and their segmentation by the region finder.

Notice that within these areas of gradation, distinct objects cannot be recognized with this method. But these areas can essentially be treated as the same as any other region in the image, since they may all contain more than one object. These different regions may be discriminated later by other visual means including learned expertise from exploring those areas of space, or parallax motion with their resolved boundaries.

The output of this unit is an array of the same size and representing the same space as the input array, only with pixel values representing region rather than intensity values.

3.2 Parallax Motion Finding

Parallax motion is done by passing through one of the arrays of tagged regions produced by the previous phase and searching for edges. It is currently recorded in terms relative to a target object Cog is presumably fixated on. An "edge" is an area of two pixels adjacent to each other which share a common neighbor which is *not* of the same region. This routine looks at successive windows of four pixels (2x2) searching for the four configurations of edges – up/down, right/left, and both diagonals. When an edge is found, a search is carried out in the second image's region map starting from one pixel either way in the direction of motion, then incrementing up to the magnitude of motion. This search is only for the same edge form as was originally discovered. If the edge is found to be moving in the direction of frame motion, then the object is presumed to be behind the fixation target. If it moves in the opposite direction, it is presumes to indicate a region in front of the fixation point.

The results of this method were initially very noisy. So we added a few simple biases. First, we checked if the region of the edge we were matching had a similar color within the difference parameter used in the region section above times a scaling factor for intensity noise between images. This greatly reduced the noise by eliminating false matches, but slightly slowed the algorithm. Next, we stopped searching for edges parallel to the direction of motion of the image. This both reduced noise (since parallel edges can logically match at arbitrary points along their boundaries) and search time (since it eliminated an edge to check). Finally, we limited the search to only edges perpendicular to the direction of motion. This gave similar results to the previous version, again with slightly faster search times. However, the "intensity" of the depth information is lower, presumably as the result of fewer matches.

4 Module Integration

As mentioned in the section on the Humanoid Project, this module is designed as a single behavior in an extremely large-scale behavior-based intelligence project. As such it is important to integrate it into both the current and the planned software environment. This section discusses some of the elements of this integration.

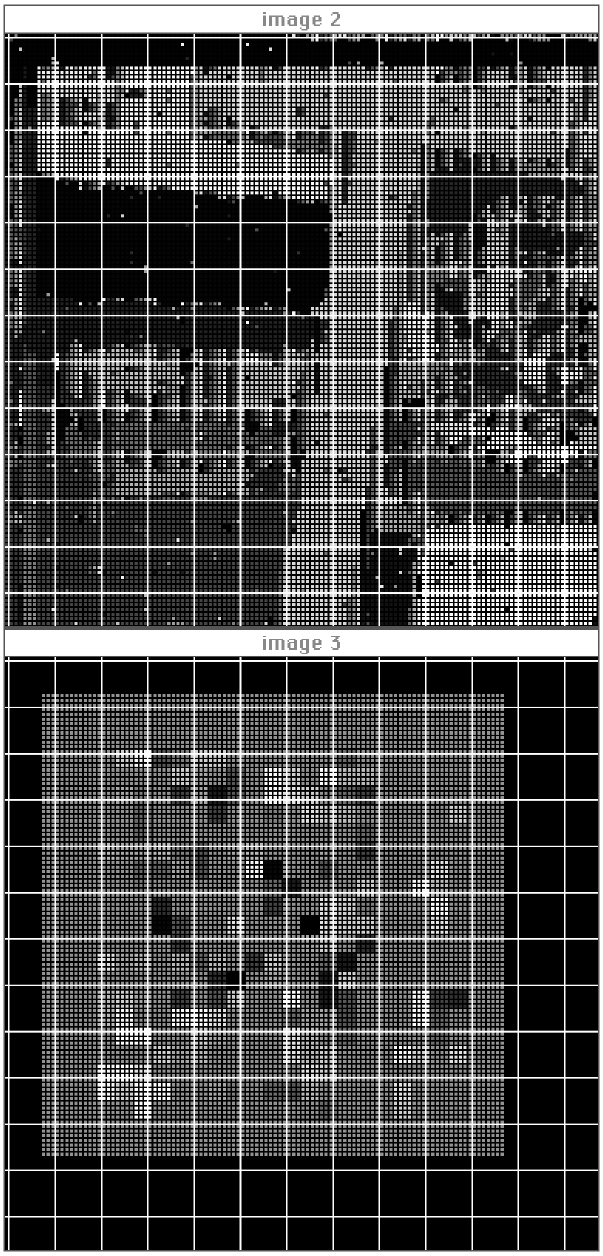

Figure 2: The depth map for the previous images, shown with the matching region map. The fixation point is the rear corner of the paper furthest back on the coffee table (25, 66). The parallax has successfully identified the coffee table (particularly its left-front leg and corner) as "in front," while the features of the couch are "behind". It also catches the right edge of the pole in the foreground, and the chair to the lower right, and some of the robot museum as being "behind". It is confused by the windows in the upper left of the picture, which should be "behind" but aren't.

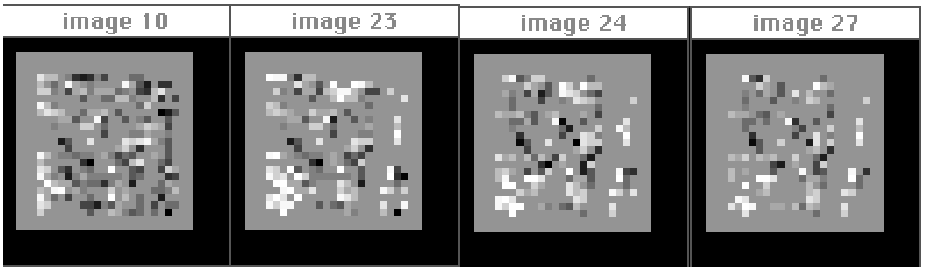

Figure 3: Depth maps for the previous images with various biases. The first map requires only that the edges be of the same orientation. The second requires the regions of the edges to have similar intensity. The third ignores edges parallel to the direction of motion. The fourth ignores edges that are not perpendicular to motion.

4.1 Dependencies

First, as mentioned above, the parallax algorithm is based on the assumption that there is a fixation region, and that the motion of the camera can be gauged by the relative motion of that object within the image frame. Of course, this assumption is not necessary. Parallax could instead be computed off of absolute camera motion. In fact, the eye-motor positions will need to be taken into account in their role in moving the fixation point within the image. There are three reasons we chose to use the fixation point instead of the gross motion. First, it is easier to hand simulate for the early experimental phases using saved frames. Second, it is more biologically plausible. Humans have receptors specifically for finding stimuli in front of and behind the plane of fixation(Rosenzweig & Leiman 1989). We appear to only learn to judge distance in absolute senses experientially; our natural depth perception seems more geared to this simple relative information. Third, as mentioned above, target tracking has already been successfully implemented by Wessler(Wessler 1995) so we knew this information was available.

Another dependency of our parallax module is that it could be confused by motion in the environment. However, Wessler has also already provided motion detection information. We can use this information to inhibit the building of depth maps in the regions known to contain current activity.

4.2 Utility

There are two currently intended applications of this module. One is as an integral part of higher-level perception, that is, a simplified version of the visual input that can be used for dealing with the surrounding environment. For example, in coordinating motion of the arm, Cog could learn the relation between extent of motion and the depth of regions affected by its motion. (We intend to have blocks for Cog to play with – one of its early tasks will be to knock the blocks off the table, the next to do it intentionally.) In human infants, this phase of learning coordination takes a long time, it is presumably done by a lot of trial and error, possibly to build mutable look-up tables.

Another use for this module, possibly combined with its own region maps, is as a building block for object identification. Once Cog has a successful model learning permanence, possibly from the above application of establishing background depth maps, and analogous process might be developed for achieving object permanence. By this time, parallax itself is of little consequence, but the methods and expertizes developed for using its output could play a major role.

It should be noted that this same approach to parallax could also be exploited for other robotic applications. For example, the depth-map building could be used by a mobile robot as an alternate method of obstacle avoidance and possibly as a building block for location recognition and map building. Or another interesting application would be to defeat the problem with Polly reported in (Horswill 1993) where, on shiny floors, Polly mistook the reflection of overhead lights for obstacles. To a parallax motion detector these reflections would look very far away, and could thus be potentially ignored by a specialized module implemented for the purpose.

4.3 Integration Constraints

None of this integration has actually been achieved yet. Cog is one of the first projects to involve the integration of systems of behaviors constructed by multiple programmers onto a single robot, and as such the software project management alone could be considered a research project. Coordinating team software projects and integrating behaviors in distributed systems are both hard problems. Of course, these two problems are both solved every day by researchers and developers, but this will be an interesting project for Cog nonetheless.

One element of coordination is determining how the information will be made available and how and when processor time will be monopolized. The idea for the output of the parallax vision is that it should be more or less uniform anytime it is polled, though it is continuously or at least frequently updated with new information. In contrast, if the region information were made available to outside processes as suggested above, then there would be a synchronization problem since it is dependent on a single frame which vanished from the video input long before the region information was processed.

One strategy for making this system more real-time would be to dedicate three separate processors to it – one for each region finder, and the third for the parallax on the previous images. But besides considering processor cost, we must also consider that many processors will probably be vying for images which can only be distributed to one processor first, then 6 others afterwards. The time nature of the parallax task is such that being slightly down the chain of image receptors won't hurt – in fact, the second processor could receive its images from the first. These problems are not currently an active area of research, however, because inter-process communication is.

5 Conclusions and Further Work

The results of this work are too preliminary to really assert any conclusions – we would view it more as a demonstration of the principals of "new AI" and active vision, and a report on progress in one aspect of the Humanoid project. As mentioned above, our next steps will be to optimize this algorithm into real time, then to start integrating it with other visual and motor routines. The real proof of this research will come when Cog or another robot is able to manipulate it's environment using the information this module provides.

This paper has described an active-vision algorithm for parallax depth perception which is currently being implemented as part of humanoid robot's vision system. We have also discussed the modularity of natural depth perception and the approaches to vision that are currently making vision-based robotics successful, and some of the integration tasks of a large-scale behavior-based robotics project.

Acknowledgments

First I would like to thank Ian Horswill and Rodney Brooks for teaching me what little I know about computer vision. Many of the ideas in this paper came from discussions with the entire Cog group: Rodney Brooks, Lynn Andrea Stein, Daniel Dennett, Cynthia Ferrell, Robert Irie, Charles Isbell, Robyn Kozierok, Matthew Marjanovic, Yoky Matsuoka, Brian Scassellati, Mark Torrance, Mike Wessler, Matt Williamson and Pushpinder Singh. Discourse credit also to Bill Smart and Marvin Minsky. Finally, robotics research is not possible without robots – special thanks not only to the engineers in the group mentioned above, but to Eleni Kapogiannis, Mike Binnard, James McGlurkin, Chris Barnhart, and Nick Shectman.

References

Arbib, M. A. & House, D. H. (1987), Depth and Detours: An Essay on Visually Guided Behavior, *in* M. A. Arbib & A. R. Hanson, eds, 'Vision, Brain, and Cooperative Computation', The MIT Press, Cambridge MA.

Barinaga, M. (1994), 'Watching the Brain Remake Itself', *Science* **266**, 1475–1476.

Barnard, S. T. & Thompson, W. B. (1980), 'Disparity Analysis of Images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-2**(4), 527–534.

Blake, A. & Yulle, A. (1992), *Active Vision*, MIT Press, Cambridge, MA.

Brooks, R. A. (1991), Intelligence Without Reason, *in* 'Proceedings of the 1991 International Joint Conference on Artificial Intelligence', pp. 569–595.

Brooks, R. A. (1993), L: A Subset of Common Lisp, Technical report, Massachusetts Institute of TechnologyArtificial Intelligence Lab.

Dennett, D. C. (1991), *Consciousness Explained*, Little, Brown, Boston, MA.

Ferrell, C., Scassellati, B. & Binnard, M. (1995), A Robot for Natural Human-Machine Interaction, Under consideration by IJCAI-95.

Gallistel, C., Brown, A. L., Carey, S., Gelman, R. & Keil, F. C. (1991), Lessons From Animal Learning for the Study of Cognitive Development, *in* S. Carey & R. Gelman, eds, 'The Epigenesis of Mind', Lawrence Erlbaum, Hillsdale, NJ.

Horswill, I. D. (1993), Specialization of Perceptual Processes, PhD thesis, MIT, Department of EECS, Cambridge, MA.

Rosenzweig, M. R. & Leiman, A. L. (1989), *Psysiological Psychology, second edition*, Random House, New York, NY.

Sundareswaran, V. & Vaina, L. M. (1994), Learning Direction in Global Motion: Two Classes of Psychophysically-Motivated Models, *in* G. Tesauro, D. Touretzky & T. Leen, eds, 'Advances in Neural Information Processing Systems 7', Denver, CO.

Wessler, M. (1995), Reubens: A Modular Visual Tracking System, Under consideration by IJCAI-95.