A Foveated Retina System for Robotic Vision

Marc Bolduc and **Martin D. Levine**
Centre for Intelligent Machines and
Department of Electrical Engineering
McGill University, 3480 University St.,
Montréal, Québec, Canada H3A 2A7
email: bolduc@cim.mcgill.edu

Abstract

Image data reduction achieved by the retina of primates provides a wide field-of-view and high resolution where needed, without creating a large amount of data at the outset. To design a sensory system that is biologically motivated, a data reduction model is required. Existing systems employ models based on conformal mappings which provide uniformly averaged non-overlapping receptive fields (RF's). These use a look-up-table (LUT) technique to produce the variable resolution output. In contrast, models with overlapping RF's permit a variety of RF functions at the cost of additional computational complexity. To process images using such a model, an extension of the LUT technique could be used, but an adaptation of a scan-line algorithm proves to be more efficient, both with respect to computational time and memory requirements. Notwithstanding the improved efficiency, a parallel network of DSP processors is required to ensure adequate throughput. This type of implementation raises issues of task scheduling and data distribution. We are currently experimenting with a small network of TMS320C40[1] DSP processors to implement a foveated system which we intend to mount on an autonomous mobile robot.

*This research is partially supported by the Natural Sciences and Engineering Research Council(NSERC) of Canada and by the National Centers of Excellence Program through IRIS. MB would like to personally thank NSERC for their scholarship support. MDL would like to thank CIAR and PRECARN for their support.

[1]Trademark of Texas Instruments

1 Introduction

This paper is concerned with one aspect of the development of a stereo foveated camera system for an autonomous mobile robot. The motivation behind using such a camera for robotic visual systems is to simultaneously provide both a high resolution image of a scene focussed on the task being performed, and a wide field-of-view for collision avoidance and potential interest point determination. If a uniform resolution sensor were used, and these two requirements were still paramount, the sensory output image would have to be very large. This in turn would make processing of the output extremely time-consuming. To reduce the amount of data, and at the same time provide a wide field-of-view and high resolution where needed, we can turn to the primate visual system for guidance.

Primate eyes have the property that they subtend a wide field-of-view and provide high resolution detail at the center of gaze, yet transmit through the optic nerve much fewer signals than there are photoreceptors in the retina. There are approximately 10^6 optic nerve fibers, whereas there are approximately 10^8 rods and $5x10^6$ cones in each retina. If we consider the spatial distribution of the cone cells, we see that there are approximately $14x10^5$ cones per mm^2 in the central area of the retina (fovea), and about $5x10^3$ cones per mm^2 in the remaining area (periphery) [1]. This suggests that in the case of scotopic vision, there is a substantial amount of subsampling in the periphery as compared to the fovea. However, this alone does not account for all of the data reduction exhibited by the retinal system. Cone outputs are also averaged together and spatially organized into clusters called receptive fields (RF's). This is achieved by a network of cells which connects cone signals to gan-

glion cells, which in turn are responsible for sending the averaged signals via their axons (forming the optic nerve) to the visual cortex [1]. The ganglion cell RF's are roughly circular with a center-surround averaging configuration(like a doughnut) that can be modelled by a difference-of-Gaussians profile. Within the fovea, most cones project to the centre area of two ganglion cell RF's, leaving the possibility for having one ganglion cell per centre-surround polarity (on-centre or off-centre) per cone [1]. Outside the fovea, the size (diameter) of the RF's, which are said to overlap by about 50% [2], increases linearly with eccentricity[2]. In addition, the density of RF's has a inverse square dependence on eccentricity [1], which provides a linear decrease in the resolution with respect to eccentricity. Since the cone distribution is roughly uniform in the periphery, RF signals from larger eccentricities carry the average of a greater number of cone outputs. Maps of the retina in the first area of the visual cortex (V1), which receives signals from the ganglion cells via the LGN, show that the correspondence between retinal and cortical locations resembles a complex logarithmic map. This is by no means a complete picture of the processing performed in the primate retina, but the apparent variable resolution image compression scheme just outlined is suitable to the requirements of a mobile robot visual system.

The data reduction afforded by the retinal models, and the potential of reduced processing time, have led to the development of sensors which perform the data reduction and techniques for using nonuniform resolution images in vision applications. Sandini et al. [3] have designed an analog implementation of a retinal model based on CCD technology, containing a 1920 pixel periphery a 120 pixel fovea. Some digital implementations of retinal models include the TRC Mapper [4][5], the TI/JSC Programmable Remapper [6], and the Cortex-I[7], all of which use a look-up-table (LUT) mapping technique. Retinal data reduction has been proposed as a compression scheme for visual communication over standard voice bandwidth telephone lines [8], and for remote driving (telemetry) [5]. Using the scale and rotation invariance properties of these retina-like sensors, some techniques for object recognition have been studied [9]. These same properties are also interesting for solutions to optic flow estimation, time-to-collision determination, and stereo disparity measurement [10, 11, 12]. Nonuniform data reduction camera systems can also be used for tracking and spacecraft docking applications [4, 13].

This paper discusses research to develop a biologically motivated nonuniform resolution sensor. Its intended application is for autonomous mobile robot navigation. To design a prototype motivated by the primate retina, we need to choose a model for data reduction, an efficient algorithm to perform the input-to-output mapping, and a hardware platform to implement the algorithm. Section 2 introduces a variety of retinal mapping models, their implementations, and details of the overlapping circular RF model used in our prototype. Section 3 describes the LUT algorithm and discusses how an adaptation of a scan-line algorithm can reduce computational time given certain assumptions about the processing architecture. Section 4 presents an implementation of the scan-line algorithm based on a parallel network of TMS320C40's. Also presented are preliminary performance results and a short discussion of certain technical issues yet to be resolved.

2 Retinal Mapping Models

The models for placing RF's and determining pixel-to-RF correspondences can be divided into two categories: conformal mapping models and overlapping circular receptive fields models. Both types provide scale and rotation invariance in their peripheral outputs [14, 15, 2], and assume square input pixels.

Two conformal retinal mapping models are based on the complex logarithm function: $w = \log z$ [16] and $w = \log(z + a)$[15]. Complex variables z and w represent pixel coordinates in the input and output images, respectively. For the $log(z)$ model, the input image mapping template consists of a *log-polar grid*: a set of rays emanating from the centre of the image at regular angular increments, and a set of circles where the ratio of the radii of adjacent circles is constant. Adjacent circles and adjacent rays bound rings (annuli) and sectors, respectively, in the input image. Pixels within a given ring and sector make up a RF,and are uniformly averaged to produce an output value. These values are arranged in a rectangular periphery output grid where one axis corresponds to the index of the sector in which the RF lies (angular component), and the other, to the index of the ring in which the RF lies (log-radial component). The resulting coordinate system is a *log-polar coordinate system.*

[2]The displacement in radians from the fovea.

Since the $\log(z)$ function has a singularity at $z = 0$, a second output image is produced: the fovea. It consists of a disc containing pixels from the centre of original input image. As for the $\log(z + a)$ input image template, it can be created by cutting a vertical slice of width $2a$ from the middle of the $log(z)$ template and bringing the two remaining pieces together. The shape of the RF's is the same as those in the $log(z)$ template except along the vertical midline, and the pixels within each RF are also uniformly averaged. The output values are arranged in a *butterfly* image: each of the left and right halves of the template corresponds to a horizontal parabolic shape, and the vertices of the two parabolas touch. The horizontal axis corresponds to the radial (ring) component of the RF, and the vertical axis, to the angular (ray) component. Adjacent RF's on each side of the vertical midline are no longer adjacent in the output image. No foveal image is produced with this version of the mapping since the singularity problem no longer exists. For both of these conformal mapping models the RF's generated do not overlap, their contents are uniformly averaged, and each input image pixel contributes to only one RF value.

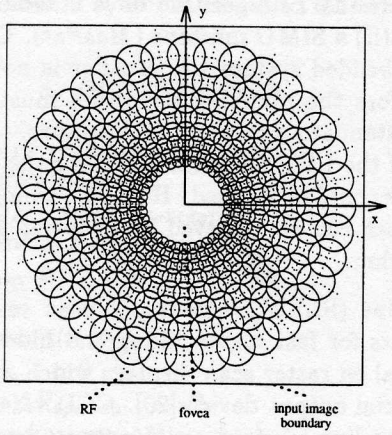

Figure 1: Overlapping circular RF mapping template using our variation of the model ($c_{d/e}$=.25, o_v=.3). The dashed lines represent the log-polar grid on which the RF's are centered.

In contrast, overlapping circular RF models permit each pixel to contribute to more than one RF, and not all pixels contribute to the same number of RF's. Notwithstanding this increase in mapping complexity, overlapping circular RF models are attractive for a mobile robot visual system because a variety of RF profiles (e.g., Gaussian, difference-of-Gaussians)[1] can be easily implemented. Hence we selected this type of model for our prototype. Wilson [2] has proposed such a model and defined biologically-based parameters to constrain the placement of the RF's on the image template. These are the foveal radius r_f, the RF overlap factor o_v, and the RF size-to-eccentricity ratio $c_{d/e}$. In this context, RF size is the diameter of the RF circle, and the eccentricity of an RF is the radial distance from the center of the RF to the center the image. The RF overlap factor indicates the fraction of the diameter of a receptive field that is overlapped by one of its neighbors. Based on neurophysiological data, Wilson fixed the RF overlap o_v to 50%, and the RF diameter to radial distance ratio $c_{d/e}$ to 0.25. Yamamoto et al. [17] have extended Wilson's model to permit variable $c_{d/e}$ and o_v. The mathematical expressions for positioning the circular RF's based on the parameters in both Wilson's and Yamamoto's models contain a small-angle approximation. We have investigated another set of RF positioning expressions, also with variable $c_{d/e}$ and o_v, where this approximation has been removed.

Figure 1 illustrates a mapping template using our version of the RF positioning equations. The RF centers are arranged along rays originating at the centre of the image, and along circles also placed at the centre of the image. In fact, the RF centres are at the intersection of the circles and rays of a log-polar grid. To constrain the placement of these centre points, we need to derive two positioning parameters from the three model parameters, namely, the angle θ between adjacent rays and the ratio k of radii of adjacent circles. The ray spacing angle is given by

$$\theta = \cos^{-1}(1 - c_{d/e}^2/2(1 - o_v)^2) \qquad (1)$$

As for the radii ratio, we used an adjacent circle spacing constraint which ensures that the average of the RF overlap of a RF on a circle i by its neighbors on circles $i-1$ and $i+1$ is the desired overlap o_v. Given that the first ring of RF's is at the foveal radius r_f (where $i = 0$), this constraint forces the circles to have radii e_i, given by

$$e_i = r_f k^i, \qquad (2)$$

where

$$k = \frac{\left(\begin{array}{l} -c_{d/e}(1-2o_v)- \\ \quad (4+c_{d/e}^2(1-2o_v)^2 - c_{d/e}^2)^{1/2} \end{array}\right)}{(c_{d/e}-1)}. \quad (3)$$

The difference between these positioning parameters and Yamamoto's version is very small (see [18] for details of this comparison and the derivations for k and θ).

The three versions of the overlapping RF model provide similar templates, but they differ on how pixels within the RF's are averaged, how the central area of the input image (the fovea) is treated, and how the output values are organized into output images. In Wilson's model, the central area of the template (the area within the first RF circle) also contains RF's, and the size of all foveal RF's is the same as those found at the fovea boundary; this produces a fixed number of foveal output pixels regardless of the foveal radius. Uniform averaging of pixels within RF's is used, and the output values are organized in a manner similar to Schwartz's butterfly image. Yamamoto et al. implemented their version of Wilson's model in a system called *Fovia* [17] which uses a *MasPar* (SIMD) machine. Each RF is assigned to a processor, and RF values are the (Gaussian) averaged intensities of a fixed number of input pixels regardless of the RF size[3]. Like Wilson's model, the fovea contains RF's and the foveal output size is independent of the foveal diameter. Unlike Wilson's model, and much like the $\log(z)$ model, two output images are produced: a fovea with a Cartesian coordinate system and a (rectangular) periphery with a log-polar coordinate system. Based on Yamamoto's implementation, Baron and Levine have recently redesigned this retinal mapping to make use of all the pixels within the RF circles, while employing a circularly symmetric Gaussian profile to weight the pixels within the RF's [19]. Also, the fovea no longer contains RF's, and the foveal output is a copy of the central disc of the input image. As for the prototype system developed here, it was decided to produce two output images as in Baron and Levine. Also, the RF averaging profile is left as a selectable system parameter. This permits the computation of a variety of filter templates.

[3]This is done for computational reasons and involves a subsampling strategy.

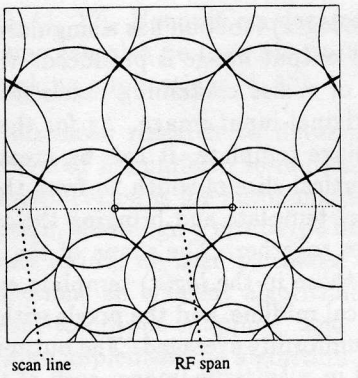

Figure 2: The intersection of a scan line (dotted) with a particular receptive field template. The solid line represents the span of the middle receptive field on the scan line.

3 Mapping Algorithm

Overlapping RF models are interesting, but since each input image pixel contributes to many RF's, mapping an entire input image is much more expensive computationally than conformal mapping models. The computer implementations of the models described previously make use of (i) a look-up-table (LUT) algorithm on a standard processor or (ii) a SIMD machine (*MasPar*). Clearly, for an embedded system the *MasPar* is not suitable. Therefore the mapping algorithm must run on a more standard processor architecture. An extension of the LUT algorithm used for the conformal models could be devised. However, a more efficient approach can be achieved by adapting a scan-line algorithm.

Scan-line (SL) algorithms are used in computer graphics for fast region filling and hidden surface removal on raster scan displays which are line-by-line serial output devices[20]. CCD video cameras are also line-oriented serial output devices. The original motivation for using this approach was to determine whether it was possible to perform the retino-cortical mapping as fast as the video data exited the camera. Although the high camera output rate makes this very difficult, using the scan-line approach does have some benefits.

Figure 2 shows a span of pixels within a receptive field (RF span) intersected by a scan line. Given the availability of (i) pixel data for a given line, (ii) a span's endpoints, and (iii) the appropriate pixel weights, we can add all pixel contributions for a RF span before retrieving from memory and updating the RF output value. If we are provided with the

information required for every RF span on a given line, updating the values of RF's which intersect the line will occur only once on each line. The general idea behind the SL algorithm is to maintain RF span information from one line to the next by updating the span endpoints, eliminating list entries for RF's which no longer intersect the new line, and adding new list entries for RF's which begin on the next line. Updating the RF span information for every line is a fair amount of work, and at the cost of additional memory, we can instead precompute the span information for every scan line in the input image, thereby creating a *span information table* (SIT).

To formally define algorithm ASL, the assumption made about the computational architecture is that calculation (ALU/FPU) operations take only register arguments. This means that only memory read/writes have special addressing modes. Value registers are represented by the variable R and the subscripts indicate the contents of the register. The address register is given by the variable AR, and dereferencing of the address (indirect addressing mode) is indicated by the $*$ prefix. The formal steps of the algorithm ASL are explained below; a table of symbols used also follows (Table 1):

Algorithm ASL

$\forall$ span s

1. $*AR_{SIT} \rightarrow AR_{spandata}$
2. increment(AR_{SIT})
3. $*AR_{SIT} \rightarrow AR_{weightdata}$
4. increment(AR_{SIT})
5. $R_{accum} = 0$
6. $*AR_{SIT} \rightarrow R_{count}$
7. increment(AR_{SIT})
8. If $R_{count} > 0$
   (a) $*AR_{spandata} \rightarrow R_I$
   (b) increment($AR_{spandata}$)
   (c) $*AR_{weightdata} \rightarrow R_w$
   (d) increment($AR_{weightdata}$)
   (e) $R_w R_I \rightarrow R_{contribution}$
   (f) $R_{accum} + R_{contribution} \rightarrow R_{accum}$
   (g) $R_{count} - 1 \rightarrow R_{count}$
9. $*AR_{SIT} \rightarrow AR_{RFvalue}$
10. increment(AR_{SIT})
11. $*AR_{RFvalue} \rightarrow R_{RFvalue}$
12. $R_{RFvalue} + R_{accum} \rightarrow R_{RFvalue}$
13. $R_{RFvalue} \rightarrow *AR_{RFvalue}$

For each span, the references to (or address of) the input pixel at the beginning of the span, and the beginning of the span weights are obtained from the SIT (steps 1 through 4). Next, a temporary accumulation register R_{accum} is initialized (step 5). To control the completion of the pixel contribution loop (step 8), a count of the number of pixels within the span, also obtained from the SIT (step 6), is used. For each pixel in the span, the pixel intensity and the pixel weight are obtained (steps 8(a) and 8(c)), and the pointers to the input and weight sequences are incremented (steps 8(b) and 8(d)). The pixel value is then weighted and added to R_{accum} (steps 8(e) and 8(f)). Once all of the contributions for an RF span are tallied, the address for the RF value is obtained from the SIT (step 9). The RF value is retrieved (step 11), updated (step 12) and stored anew (step 13). Overall, there are eight steps performed for each RF contribution (or span pixel), two of which are memory accesses. There are six memory accesses and six calculations for each RF span, for a total of twelve steps per span. The information needed for each span from the SIT consists of a reference to the beginning of the span data, a reference to the beginning of the weight list for the span, the size of the span, and the location of the peripheral image output value. A copy of the span weights could be stored for every span of each RF, but since many averaging profiles are circularly symmetric (e.g., uniform, Gaussian, DOG), corresponding spans in RF's of the same size can use the same pixel weighting factors. Therefore the RF weight factors need only be stored once for each RF size.

AR_{SIT}	reference to span information
$AR_{spandata}$	reference to input data
$AR_{weightdata}$	reference to span weights
R_{accum}	span contribution accumulator
R_{count}	contribution loop counter
R_I	pixel intensity value
R_w	pixel contribution weight
$R_{contribution}$	pixel contribution value
$AR_{RFvalue}$	reference to RF value
$R_{RFvalue}$	retrieved RF value

Table 1: Symbol definitions for algorithm ASL.

When compared to a similar description of the LUT algorithm, analysis shows that algorithm ASL is more efficient both in terms of memory size and computation time. The details of the analysis can be found in [18]. In the table for algorithm LUT, weights are kept for every contribution of each input pixel. On the other hand, in

algorithm ASL, pixel weights are stored only once per RF size, and table entries are in terms of RF spans which are less numerous than input pixels. These two facts produce a significant reduction in required memory. For ranges of o_v of $(0, .5]$ and $c_{d/e}$ of $(0, 1]$, for a foveal radius r_f of 50 pixels, and a 512 by 512 input image, the SIT of algorithm ASL demands at most 30% the space required by the table of algorithm LUT. As for computation time, analysis indicates that if each step of the algorithm corresponds to a single CPU cycle, then for the same range of parameters as above, algorithm ASL takes at least 10% fewer cycles to complete the mapping of a single input image. The performance difference is even more significant if the assumptions about processor architecture reflect the actual DSP used in our prototype. The C40 features a four level instruction pipeline, a variety of addressing modes, a looping instruction, as well as certain parallel instructions, including a simultaneous multiply/add. Under these assumptions about the architecture, dereferencing and incrementing a pointer (e.g., steps 1 and 2 in algorithm ASL) takes a single cycle. Due to the looping feature, the steps for decrementing the loop index R_{count} and checking for loop completion vanish at the cost of an extra instruction before the pixel contribution loops in both algorithms. Also, the the pixel contribution loop of algorithm ASL corresponds to one of the DSP's parallel instructions, which means that the loop can be executed in a single cycle at the cost of adding two instructions around the loop. Therefore, these specific assumptions change the number of cycles required from 12 to 10 per span (steps 1 to 7 and 9 to 13), and from 8 to 1 per pixel contribution (steps 8(a) to 8(g)). With these reductions in the number of processing cycles taken into account in both algorithms, algorithm ASL requires at most 51% of the number of cycles necessary for algorithm LUT.

4 Prototype Design.

Although the comparison of the two algorithms described above uses idealized assumptions about processing architecture and memory latency, the large relative difference in processing steps required between the two algorithms is encouraging. Considering a 256 x 256 input image (65,536 pixels) at 30 frames per second, the processing rate required is 1,966,080 pixels per second. Given a 50 ns instruction cycle time, as is the case with a 40Mhz TMS320C40, the CPU can appropriate at best 2.5 instructions per pixel. This is not satisfactory, especially since pixels may contribute to more than one RF. Also, we are planning to use an input image larger than 256^2. To ensure adequate throughput, we are investigating the use of a small network of DSP's. Given the input image size and the complexity of the mapping problem, video frame rates are not possible at this time. This problem can be alleviated by using a frame buffer.

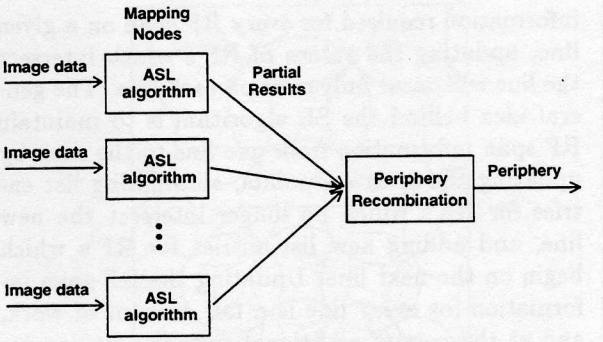

Figure 3: Distribution of the ASL algorithm on a multiprocessor network

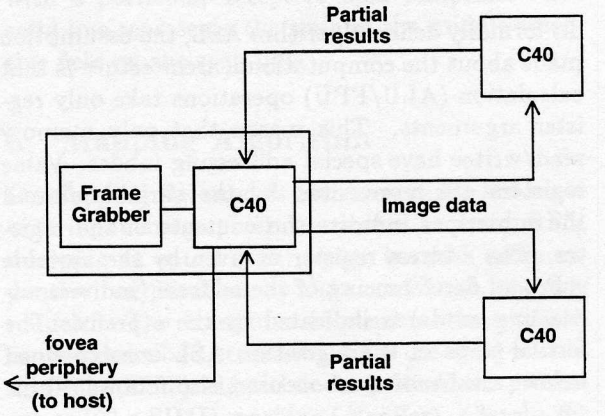

Figure 4: Current version of the prototype .

The basic principle behind the parallelization of the algorithm is to divide the image data into parts and distribute them among a set of mapping processors. Each produces a partial result which is combined with those of the other processors. This is illustrated in Figure 3.

To implement this idea, we are currently using a network of C40 nodes which consists of a frame grabber (FG) node linked to a host workstation, and two mapping nodes (see Figure 4). The worker nodes basically perform the ASL algorithm for the image data each receives from the FG node, and transmit the partial results back to it. Since the C40 has a direct memory access coprocessor

(DMAC), the data input/output can be done in parallel with the mapping computations. The FG node transfers image data to the mapping nodes, and receives the partial results from them. These are combined to create the peripheral output. It also obtains the foveal image directly from its frame buffer and transfers both images back to the host. With the help of the DMAC, some of these tasks can also be performed in parallel. Work in progress will fine-tune this scheduling scheme in order to minimize bus and memory conflicts between CPU/DMAC pairs.

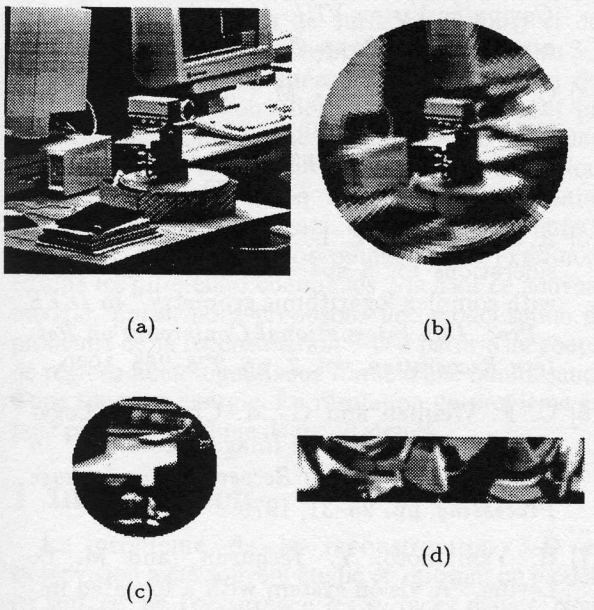

Figure 5: Sample of a typical retinal mapping with the focus of attention at the center of the image (a) input image (515 by 484), (b) inverse mapping of the output images (c) fovea (101 by 101) and (d) periphery (30 by 126). The parameters used are $r_f = 50$ pixels, $c_{d/e} = .1$, and $o_v = .5$. The input image contains 249744 pixels, whereas the foveal and peripheral images together contain 13981 pixels, giving a compression ratio of 17.86

Another issue under investigation is the splitting of the image data among the processors. Since SL algorithms are line-oriented, image data are distributed line by line in the current implementation. This distribution scheme needs to be improved in order to minimize CPU idle time and mapping latency. In other words, it is desirable to insure that the mapping CPU's are not idle for unnecessary amounts of time due to DMA transfer of the next input data set.

Figures 5(c) and 5(d) show typical output images obtained for the input image in Figure 5(a). Figure 5(b) is an inverse mapping of the output images onto the input plane, which illustrates the loss of resolution induced by the retinal mapping. The input image is 515 by 484, and the periphery computation time is approximately 0.1 seconds. This time excludes recombination of the partial results which is performed concurrently with the mapping of the next frame. Since the mapping phase is the most time-consuming task, the recombination adds latency to the overall mapping problem without affecting the throughput. Based on experimentation to date, we do not expect to obtain more than 5 to 10 frames per second with this network configuration. We are in the process of modifying the network by using a additional mapping nodes with faster memory and a dedicated recombination node.

5 Conclusions

The variable resolution feature of the primate retina is a good basis for designing a mobile robot vision system where wide field-of-view, high resolution and small output image size are required. In the prototype system being developed, an overlapping circular receptive field model with biologically-based parameters is used. Given the increase in computation required by this model, we have found that an adaptation of a scan-line algorithm is more efficient in terms of memory requirements and computational time as compared to a look-up-table algorithm. This is especially true if a pipelined DSP such as the C40 is used. Even with the improvement in efficiency, a network of processors working in parallel is needed to insure adequate performance. This entails splitting the input image, distributing it, performing the mapping on more than one processor, and then recombining the partial results. Currently the prototype system contains a small decoupled MIMD multicomputer network which yields 5 to 10 frames per second. Work in progress will help determine how to subdivide the image data more efficiently, and how to schedule the various tasks associated with the mapping in order to achieve minimum mapping latency.

References

[1] S. Shah, "Biological information processing in the primate retina," Tech. Rep. TR-CIM-92-

11, Centre for Intelligent Machines, McGill University, 1992.

[2] S. W. Wilson, "On the retino-cortical mapping," *Intl. Journal on Man-Machine Studies*, vol. 18, pp. 361–389, 1983.

[3] J. V. der Spiegel, G. Kreider, C. Claeys, I. Debusschere, G. Sandini, P. Dario, F. Fantini, P. Belutti, and G. Soconi, "A foveated retina-like sensor using CCD technology," in *Analog VLSI and Neural Network Implementations* (C. Mead and M. Ismail, eds.), Boston: DeKluwer, 1989.

[4] C. F. Weiman and R. D. Juday, "Tracking algorithms for log-polar mapped image coordinates," *SPIE - Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, vol. 1192, pp. 843–853, 1989.

[5] C. F. Weiman, "Video compression via log-polar mapping," in *SPIE Symposium on OE/Aerospace Sensing*, 1990.

[6] T. E. Fisher and R. D. Juday, "A programmable video image remapper," in *SPIE - Digital and Optical Shape Representation and Pattern Recognition*, vol. 938, pp. 122–128, 1988.

[7] B. B. Bederson, R. S. Wallace, and E. L. Schwartz, "A miniaturized active vision system," in *Proc. 11th IAPR International Conference on Pattern Recogntion Vol IV, Conference D: Architectures for Vision and Pattern Recognition*, pp. 58–61, 1992.

[8] R. S. Wallace, B. B. Bederson, and E. L. Schwartz, "Voice-bandwidth visual communication through logmaps: The telecortex," in *Proceedings of IEEE Workshop on Applications of Computer Vision, Palm Springs, Ca.*, pp. 4–10, IEEE Computer Society Press, 1992.

[9] J. I. Minnix, E. S. McVey, and R. M. Inigo, "Rotation and scale invariant pattern recognition using a multistaged neural network," in *SPIE - Visual Communications and Image Processing '91: Image Processing*, vol. 1606, pp. 241–251, 1991.

[10] C. F. R. Weiman, "3-d sensing with polar exponential sensor arrays," in *SPIE - Digital and Optical Shape Representation and Pattern Recognition*, vol. 938, pp. 78–87, 1988.

[11] N. C. Griswald and C. F. R. Weiman, "A modification of the fusion model for log polar coordinates," in *SPIE - Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, vol. 1192, pp. 854–866, 1989.

[12] S. L. Bartlett and R. Jain, "Motion stereo and and ego-motion complex logarithmic mapping (eclm)," in *SPIE - Digital and Optical Shape Representation and Pattern Recognition*, vol. 938, pp. 138–145, 1988.

[13] J. G. Bailey and R. A. Messner, "Docking target design and spacecraft tracking system stability," in *SPIE - Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, vol. 1192, pp. 820–831, 1990.

[14] C. F. Weiman and G. M. Chaikin, "Logarithmic spiral grids for image processing and display," *Computer Graphics and Image Processing*, vol. 11, pp. 197–226, 1979.

[15] A. S. Rojer and E. L. Schwartz, "Design considerations for a space-variant visual sensor with complex logarithmic geometry," in *IEEE - Proc. 10th International Conference on Pattern Recognition*, vol. 2, pp. 278–285, 1990.

[16] C. F. Weiman and G. M. Chaikin, "Logarithmic spiral grids for image processing," in *IEEE - Proc. Pattern Recognition and Image Processing*, pp. 25–31, 1979.

[17] H. Yamamoto, Y. Yeshurun, and M. D. Levine, "A vision system with a foveated image sensor (translated from japanese)," *Transactions of the Institute of Electronics, Information and Communication Engineers.* Accepted for publication, Sept. 1993.

[18] M. Bolduc, "A foveated sensor for robotic vision," Master's thesis, McGill University, 1994. In preparation.

[19] T. Baron, M. D. Levine, and Y. Yeshurun, "Exploring with a foveated robot eye system." Submitted to the 12th ICPR – Conference on Parallel Processing.

[20] J. D. Foley, A. V. Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, pp. 92–98. Addison-Wesley, 2nd ed., 1990.