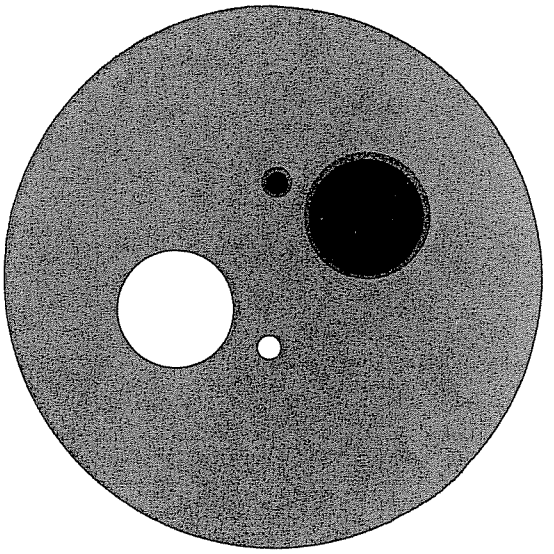
COMPUTER SCIENCES
DEPARTMENT

University of Wisconsin-
Madison



A PYRAMIDAL APPROACH FOR THE RECOGNITION OF NEURONS USING KEY FEATURES

by

Ze-nian Li
and
Leonard Uhr

Computer Sciences Technical Report #591

April 1985

A Pyramidal Approach for the Recognition of Neurons Using Key Features

Ze-nian Li and Leonard Uhr
Department of Computer Sciences
University of Wisconsin-Madison

ABSTRACT

Pyramid-like hierarchical structures have been shown to be suitable for many computer vision tasks. In this paper a pyramidal approach for recognition of neurons from an electron micrograph is described. Key features for object recognition are defined. Transforms for extracting features are performed hierarchically in a simulated pyramid, cascading local parallel operations. The algorithms described in this paper are expected to be fairly expandable to general purpose pyramidal vision systems.

key feature object recognition parallel processing pyramid transform

1. Introduction.

The recognition of objects using a *recognition cone*, by extracting and combining features at levels with different resolutions was proposed by Uhr [1][2]. The layered hierarchical structure of this 'recognition cone' approach resembles the human vision system. It can be realized by parallel processors, hence it is well suited for fast and efficient VLSI multicomputers.

The pyramid structure introduced by Tanimoto and Pavlidis [3] has the same concept of multi-resolution and hierarchical representation of image data. A prototype pyramid has n levels, each level k ($0 \leq k < n$) has $2^k * 2^k$ nodes (simple processors). Each node at level k is hard-wired to its 13

neighbors, i.e. 1 parent, 8 siblings and 4 children. See [5] for more descriptions of pyramids.

It has been shown [4][5][6] that, using local windows and parallel computation, pyramid-like structures are very efficient for low level image processing, e.g. averaging, histograming, edge detection, median filtering, image segmentation, etc. Some earlier systems [2][4] also used pyramid-like structures in their object recognition process.

In this paper we demonstrate a pyramidal approach for the recognition of nerve cells.

The raw data come from an electron micrograph which depicts a cross section of neurons from the visual system of the larval corn borer (lepidoptera). Neurologists have great interest in understanding how the neurons project from eye to brain, and how they functionally and structurally communicate with each other. Techniques for Computer-Aided Reconstruction by Tracing Of serial Sections (CARTOS) were reviewed by [7][8]. The task is to realize 3D reconstructions by using a series of cross sections (2D images), then to obtain 3D displays on graphics terminals. One way to procure traces of cell boundaries is to record a finite number of coordinates of boundary points. This can be done manually using some sort of coordinate input device, e.g. tablet. But it is time consuming and tedious. Our intention is to achieve automatic recognition of the boundaries, i.e. to derive the necessary shape and coordinate information to enable recognition and computerized 3D reconstruction.

In order to recognize objects, we must extract features first. In many biomedical images, complete recovery of broken edges is not always possible. Generally the pieces of evidence one obtains first are local features. One has to accumulate partial evidence to approach the recognition goal. A good description of 'Local-Feature-Focus' method is given by [9], where one local feature in an image is found, referred to as *focus feature*, and is thereafter used to predict a few nearby features to look for.

There is evidence for 'Focus of Attention' in human vision [10][11]. Human perceivers usually do not stare at the entire area with the same intensity. Rather, after a rough scan they focus on the 'interesting' portion of a scene. This act is one of the factors which makes the human vision

system so effective. In the event that the observed object is partially occluded, humans focus on the visible features which are important in the recognition of the object. Once enough evidence is accumulated, humans will confidently complete the recognition process.

We will name those features, which appear frequently and are important to the object recognition, as *key features*. The existence of key features often strongly implies the existence of the object in question. As examples, window (or door) is a key feature for a house, glass pane is a key feature for a window. Oftentimes we can ascertain a hierarchy of key features. For instance, in shape analysis key features will usually be corners, long lines, long curves, or locations with dramatic curvature changes.

This paper describes methods for the recognition of neurons using key features in pyramids. It will be shown that the local parallel processing ability of pyramidal processors can be exploited to accomplish many desired computations.

2. The Recognition of Neurons in an Electron Micrograph.

Fig. 2.1 is an electron micrograph which is a cross section of neurons in a neuropil region of the insect brain (supraesophageal ganglion).

Our pyramid is a software simulation on the VAX 11-750. The structure is highly reconfigurable. The pyramid used in this example has 10 levels (0 - 9). Each node at level 5 and above has its own 2k bit memory, whereas nodes at the larger lower levels have less local memory space. There are mainly two reasons to design lower level nodes with less local memory space: (1) it is necessary to limit the size of our simulation package, (2) there are more combinations of features at higher levels, therefore more need for memory space. The following table shows the configuration of this simulated pyramid machine.

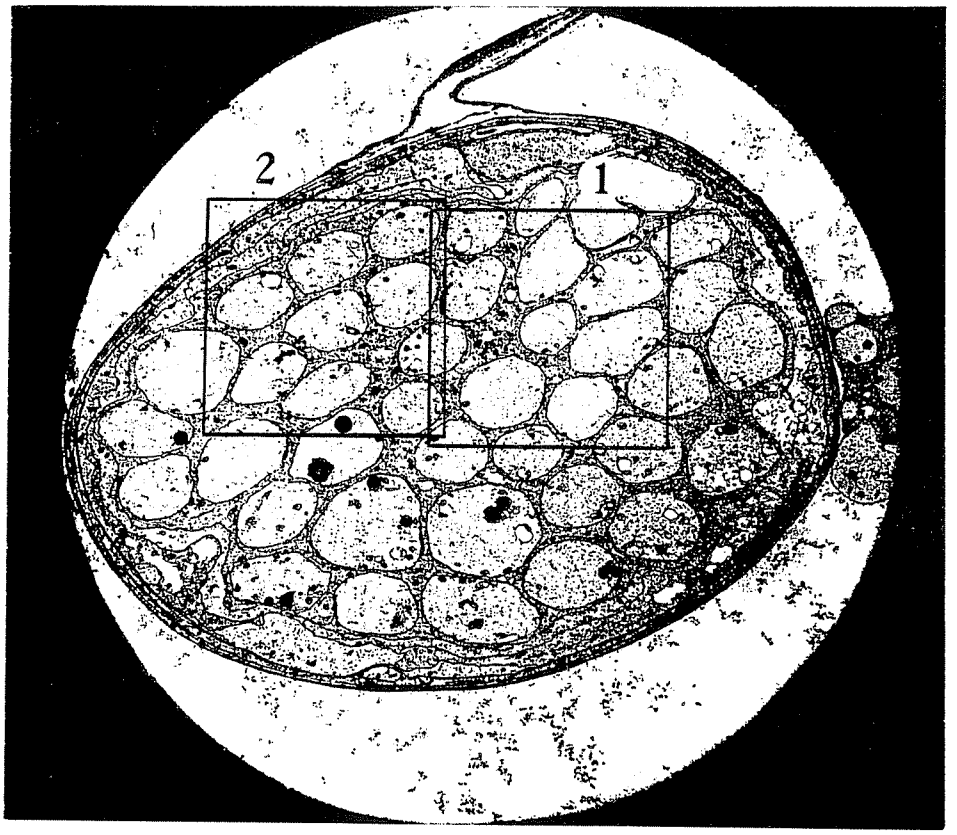


Fig. 2.1 Electron Micrograph of Neurons

Level	Size	Local Memory (bit)
0	1 x 1	2048
1	2 x 2	2048
2	4 x 4	2048
3	8 x 8	2048
4	16 x 16	2048
5	32 x 32	2048
6	64 x 64	1024
7	128 x 128	256
8	256 x 256	64
9	512 x 512	16

At different steps in the recognition process, pyramidal operations with different window sizes (2x2, 3x3, 4x4) are used. In other words, we are using both overlapped and non-overlapped pyramids as needed [5].

The digitized image of Fig. 2.1 has a resolution of 256x256. Two portions (64x64) of it are used as our test data (Image1 and Image2 as indicated by '1' and '2' respectively in Fig. 2.1). The recognition consists of following steps:

(1) Getting Micro-edges.

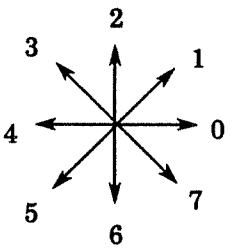
The pyramidal median filtering technique is first applied to preserve edges while reducing noise. The input image (64x64) is stored at level 6, each node on level 5 takes the median intensity value of its 3x3 child set as its intensity value. The result is a filtered image with reduced resolution.

After filtering, the well-known Prewitt edge operator is applied to the filtered image array which is now at level 5. There are eight edge masks encoded by 0,1,...,7. Fig. 2.2.

On each node of the array, convolutions are computed using these eight masks. The maximum magnitude of the eight convolution results is retained as the weight of the micro-edge, and the corresponding encoded direction is kept as the direction of the micro-edge. The result of this step is a micro-edge map of the filtered image.

(2) Getting Short Curves.

The concept of transforms in [2] is used for feature extraction in the recognition of neurons. Transforms are procedures which compute values or search for features in a set of cells in one level,



(a) Edge coding

-1	0	1	0	1	1
-1	0	1	-1	0	1
-1	0	1	-1	-1	0

(b) Masks for Edge0 and Edge1

Fig. 2.2 Edge Coding and Edge Masks

and output the values or implied features (or objects) into the same or the next level.

It turns out that the extraction of features can be directed by the micro-edge map. In case there is a long vertical edge in the input image, it is apparent that there should be many micro-edges (direction 0 or 4) lining up vertically at the location where the long edge lies. Thus at one level above the micro-edge map, micro-edges (direction 0 or 4) can be combined into short-edges. At another level above, short-edges are combined into longer edges, and so on.

In our electron micrograph images, curves are dealt with instead of straight edges. Fig. 2.3 shows a typical micro-edge map of a cell. With a pyramid structure it is desirable to locate short curves at a lower level, then combine them into longer curves and eventually take in whole cell boundaries at higher levels. Cells in our images are all nearly round or elliptical, hence mainly convex curves are under consideration.

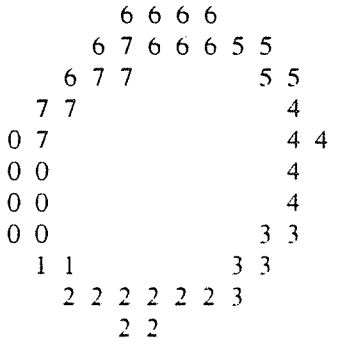


Fig. 2.3 Micro-edge map of a cell

In order to recognize objects efficiently in pyramids, every effort should be made to use only local transforms that examine small windows surrounding each node. If some feature is too global to detect efficiently at a lower level, it should be assessed at a higher level.

To cope with the small windows, we further studied the micro-maps. It was found that the edge map of any ellipse-shaped object can be divided into eight segments (Fig. 2.4(a)). The places where

micro-edges change directions, e.g. 0 → 1, 1 → 2, etc, are very important. For nearly round or elliptical objects, there are always eight such places. This phenomenon is invariant to size, orientation and geometry (elongate nature) of the cell, moreover it can always be observed in a very small window, e.g. 3x3.

Based on this phenomenon, a transform was built for short curves. At level 4, each node examines its 3x3 child set. It will claim that it finds SHORTCURVE0 if there are 'micro-edge0' at upper left and 'micro-edge1' at lower right of its window with considerable weights. In the same way, it finds SHORTCURVE1,..., SHORTCURVE7. Fig. 2.4(b) is the coding for short curves.

The coordinates of the two end points of the 'short curve' are recorded. In case of Fig. 2.5(a) and (b), the location of the most upper left 0-point and the location of the most lower right 1-point are taken.

The weight of the short curve is calculated by $W = \sum W_i * \sum W_j$, where W_i is the weight of micro-edge i, and W_j is the weight of micro-edge j. The sum is taken within the window. If the curve is salient, the local edge map could look similar to Fig. 2.5(a) and the resulting weight is relatively high. If the curve is slightly broken as in Fig. 2.5(b), it still has a good chance of being detected, usually with lower weight. The noisy spots rarely survive as curves; even if they are combined into 'short curve's, they carry very small weights.

(a) Segments of Micro-edge Map (b) Coding of Short Curves

Fig. 2.4 Edge Segments and Curve Coding

(a) Salient Curve (b) Slightly Broken Curve

Fig. 2.5 Two Micro-edge Maps of SHORTCURVE0

(3) Recognizing Cells.

Since we are using features of shape in recognizing cells close to an ellipse, a longer curve, such as a 'quarter-ellipse', is a key feature to utilize. Our transform detects such long curves much as it detects short curves. Every node at level 3 examines its 3x3 window at level 4. If, for example, SHORTCURVE0 is at upper left and SHORTCURVE1 at lower right of the window and their weights exceed certain threshold, they get combined to LONGCURVE0 (SW) at level 3.

In pyramids we are able to detect and store features hierarchically. What we have actually built is a multi-level description of the image in terms of features with locations and implicit geometrical relations.

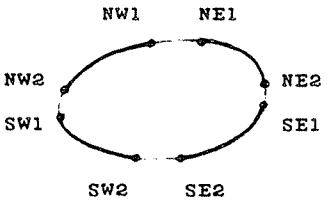
In the present terminology there could be two sets of long curves: SW, SE, NE, NW; or W, S, E, N. Either one of these two sets can be combined into a whole cell. We show the results using the first set. The transform, that combines long curves to cells, first finds the pairs of matched LONGCURVEs SW-NE and SE-NW, and then tries to combine two such corresponding pairs to a whole cell at level 2. Fig. 2.6 (a) and Fig. 2.7 (a) show the initial results for Image1 and Image2. The data shown are coordinates of the endpoints of long curves, e.g. the data in SW1 and SW2 columns are the coordinates for two endpoints of LONGCURVE0 (SW).

As one can see, not all the combinations are complete. For example, in Fig. 2.6 (a) only four cells get complete combination of SW-NE and SE-NW. The remaining three matched pairs of SW-NE did not get their corresponding SE-NW. There could be many reasons for missing pairs, e.g. some of the LONGCURVEs never got combined from short curves. The reason for this is that we

started to combine features with very small windows (3x3) and relatively high thresholds to avoid mistakes. To improve the initial results, the program looks for missing curves at the surrounding locations of the extracted key features (LONGCURVEs). In this second pass of search, bigger widows and lower weights are used. Fig. 2.6 (b) and Fig. 2.7 (b) show the results after this second search. In Fig. 2.6 (b) all seven cells in Image1 are completely combined. (Cells marked with * are those found in the second search.)

As shown in Fig. 2.7 (b) for Image2, sometimes the program still cannot attain the 'perfect' combination of four long curves for all cells (e.g. due to the inflexibility of the weights, limitation of the features used, severely broken edges or artifacts on the micrograph). But with the remaining matched pairs of LONGCURVEs (say SW-NE), the program will still predict these additional cells, though with lower confidence weight.

Following the described steps of prediction is a top-down verification process which gives the final description of cell boundaries. Up to now only a very small number of shape features have been used. Additional shapes and other important features (color, texture ...) will be incorporated in our later experiments.



Matched Total Cells:

Cells	SW1	SW2	NE1	NE2	SE1	SE2	NW1	NW2
1	(13, 2)	(14, 4)	(7, 7)	(8, 9)	(11, 8)	(15, 6)	(7, 3)	(9, 2)
2	(11,19)	(14,21)	(5,27)	(9,29)	(9,29)	(12,27)	(6,24)	(9,23)
3	(16,11)	(18,13)	(13,15)	(15,18)	(15,18)	(19,15)	(13,15)	(16,11)
4	(19,18)	(21,20)	(14,27)	(15,30)	(19,28)	(21,25)	(15,22)	(19,19)

Remaining Matched Pairs of LONGCURVES:

Pairs	SW1	SW2	NE1	NE2
1	(9,11)	(12,12)	(3,17)	(7,20)
2	(25, 5)	(29, 7)	(19,11)	(23,14)
3	(27,15)	(30,19)	(23,21)	(25,23)

(a) Initial Results

Matched Total Cells:

Cells	SW1	SW2	NE1	NE2	SE1	SE2	NW1	NW2
1	(13, 2)	(14, 4)	(7, 7)	(8, 9)	(11, 8)	(15, 6)	(7, 3)	(9, 2)
2	(11,19)	(14,21)	(5,27)	(9,29)	(9,29)	(12,27)	(6,24)	(9,23)
3	(16,11)	(18,13)	(13,15)	(15,18)	(15,18)	(19,15)	(13,15)	(16,11)
4	(19,18)	(21,20)	(14,27)	(15,30)	(19,28)	(21,25)	(15,22)	(19,19)
* 5	(9,11)	(12,12)	(3,17)	(7,20)	(7,20)	(11,15)	(3,17)	(9,11)
* 6	(25, 5)	(29, 7)	(19,11)	(23,14)	(25,13)	(28,11)	(19, 7)	(25, 4)
* 7	(27,15)	(30,19)	(23,21)	(25,23)	(25,23)	(31,20)	(23,18)	(25,16)

(b) Results after Further Search

Fig. 2.6 Cell Recognition Results for Image1

Matched Total Cells:

Cells	SW1	SW2	NE1	NE2	SE1	SE2	NW1	NW2
1	(9,11)	(11,13)	(5,17)	(7,20)	(7,20)	(11,17)	(5,14)	(9,11)
2	(5,22)	(7,25)	(1,27)	(5,30)	(5,30)	(7,28)	(1,26)	(5,22)
3	(15, 1)	(16, 3)	(10, 7)	(13,10)	(13,10)	(17, 7)	(10, 6)	(15, 1)
4	(15,22)	(17,24)	(9,27)	(11,29)	(13,29)	(16,27)	(9,25)	(12,21)
5	(29,23)	(31,23)	(24,27)	(27,29)	(27,29)	(31,27)	(24,25)	(27,23)

Remaining Matched Pairs of LONGCURVES:

Pairs	SW1	SW2	NE1	NE2
1	(18,14)	(20,15)	(12,17)	(15,19)
2	(25, 4)	(27, 4)	(20,11)	(22,12)
3	(27,10)	(28,12)	(21,19)	(25,21)

(a) Initial Results

Matched Total Cells:

Cells	SW1	SW2	NE1	NE2	SE1	SE2	NW1	NW2
1	(9,11)	(11,13)	(5,17)	(7,20)	(7,20)	(11,17)	(5,14)	(9,11)
2	(5,22)	(7,25)	(1,27)	(5,30)	(5,30)	(7,28)	(1,26)	(5,22)
3	(15, 1)	(16, 3)	(10, 7)	(13,10)	(13,10)	(17, 7)	(10, 6)	(15, 1)
4	(15,22)	(17,24)	(9,27)	(11,29)	(13,29)	(16,27)	(9,25)	(12,21)
5	(29,23)	(31,23)	(24,27)	(27,29)	(27,29)	(31,27)	(24,25)	(27,23)
* 6	(27,10)	(28,12)	(21,19)	(25,21)	(23,21)	(27,17)	(21,16)	(27,11)

Remaining Matched Pairs of LONGCURVES:

Pairs	SW1	SW2	NE1	NE2
1	(18,14)	(20,15)	(12,17)	(15,19)
2	(25, 4)	(27, 4)	(20,11)	(22,12)

(b) Results after Further Search

Fig. 2.7 Cell Recognition Results for Image2

Our experiments have shown that the transforms for extracting key features are good building blocks for the purpose of the pyramid vision. Such transforms can be accomplished by using only small windows and highly parallel computations in pyramids. In the present nerve cell example, the recognition is not affected by the size, orientation and elongate nature of the cells. This recognition is reasonably tolerant to noise and distorted edges. Moreover, starting with the steps shown in this paper, we can actually get micro-edge maps, and short edges or short curves, for any images at lower levels of the pyramid. The important step is, of course, the key feature extraction. For more complicated objects we also need more sophisticated way to represent knowledge and to control the search in recognition. There is reason to think that the described method can be generalized to detect other features and objects. Similar transforms have already given good result in one liver tissue cell image (which is used by Preston in [12]) and two simple house images.

3. Discussion.

3.1. How to assess complex features efficiently, using pyramids.

All of this program's operations, as just described, are executed in the local window surrounding each and every one of the cells at that operation's level. In a hardware pyramid multi-computer they would all be executed at the same time, in parallel. The sequence of transformations moving up through the pyramid, from larger to smaller arrays, makes possible the execution of successively more global operations. Thus a very complex operation over a very large region can be decomposed into a tree-like sequence of local window operations.

3.2. How to identify objects over magnification.

One essential characteristic of any object recognizer is that it be able to identify the same set of objects even though they are of different size. This requirement is not a trivial task for many computer vision systems, especially for those that expect to match a rigid template or an exact model. The reason is that two facts are usually not known in advance to a general purpose vision system: (1)

the size of objects (the object size itself and the distance from viewpoint to scene); (2) the resolution of the input digital image data. As an example take a simple rectangular object (e.g. door). According to the previous description, our system would use corners as key features to recognize a rectangle. But in a given image, the rectangle could have a different size due to the actual object size or the resolution of the image. Pyramids with layers of different resolutions offer a hierarchical structure for extracting features. It appears that there are three different approaches in solving this problem:

(1) Start to detect micro-edges at the original image level. Find 4 right angles at lower levels. If they are too far apart to see in a local window, build bigger right angles (with longer legs) at the next higher level, and so on, until they gather together and can be combined into a rectangle within a local window. This method always takes log(*objsize*) steps. The problem is that the input images could be noisy and hence not so perfect for building up 'big' features.

(2) Use pyramidal median filtering to get multi-resolution images. Try the same recognition processes on images of different resolution simultaneously. With this approach, usually fewer transforms are performed at fewer levels in each process. The idea is to work on the right resolution images with only small window operation. The keys to success of this method are, (a) the edges should be well preserved even in low resolution images; (b) the key features should be relatively size invariant (e.g. the right angle, or the curves that we choose do not disappear easily in low resolution images.)

(3) Start with the original image; extract right angles as key features at lower levels as in (1). If they are too far apart, pass them to the corresponding parent nodes (when passing features up, simply use non-overlapped pyramid structure, each child has its unique parent to communicate with.) until they are visible within a small window. This approach has the advantage of always fully using the information at the original higher resolution. The problem is how to make a decision as to the number of levels required to pass without knowing the object size in advance.

It seems that the combination of approach (2) and limited passing are plausible, especially for elongated objects. The control strategy part remains to be studied.

3.3. How to choose appropriate window sizes.

The pyramid, or so called multi-resolution, approach enables the effective use of only very small windows, e.g. 3x3, 4x4, in recognizing objects. Its advantage is: if the object is too large to 'see' at this level, it will be 'visible' at some higher level of the pyramid, because at the higher level the features will be pulled closer.

Once we have decided the convergence of our pyramid to be 2, any window of a size bigger than 2x2 will give us a certain degree of overlapping. In the process of using the pyramid to recognize objects, the desired degree of overlapping turns out to be an important factor in determining the sizes of the windows to use. In 2x2 non-overlapped pyramid, each node p at level k ($0 \leq k < n$) has 2x2 children at level k+1, and each of those children has 2x2 children at level k+2. Therefore node p can 'see' 4x4 children at level k+2. In general, p can 'see' $(2^{n-k})^2$ nodes at level n ($n \geq k$). In overlapped pyramids the child set grows much larger. For each node at level k, while using different windows, the child set areas at level k+1, k+2, ..., n are shown below:

	k+1	k+2	k+3	...	n
2x2	2^2	4^2	8^2	...	$(2^{n-k})^2$
3x3	3^2	7^2	15^2	...	$(2*2^{n-k}-1)^2$
4x4	4^2	10^2	22^2	...	$(3*2^{n-k}-2)^2$
5x5	5^2	13^2	29^2	...	$(4*2^{n-k}-3)^2$
.					
.					
.					
MxM	M^2	...	...	...	$((M-1)*2^{n-k}-(M-2))^2$

Thus for example, in a 5x5 overlapped pyramid, at three levels below the child set size is almost 4 times larger than the child set size in a non-overlapped pyramid. In order to be able to recognize objects at the level proportional to the logarithm of the object size above the input image,

we need to use overlapped pyramids. But now we can see that big windows also bring too much overlapping. In the present scheme of recognition, low level features are combined all the way up to high level features. Too much overlapping means that the same features will be found redundantly in many nodes at the neighborhood of the expected location. Elimination of redundant information is not always an easy job, not to mention the complexity that large windows bring while combining features. Therefore our program starts with only 3x3 windows in the neuron example. Following the clue of found key features the program then expands some of its windows to 4x4.

4. Conclusion.

Key features in visual fields play important roles in both human and computer vision systems. It is feasible to develop key feature detectors within local windows in pyramids, and to use these features to accomplish the recognition goal. The computations are highly parallel. It is expected that the algorithms described here can be expanded to general purpose pyramidal vision systems.

5. Acknowledgement.

This research was partially supported by NSF Grant DCR-8302397, mod.1 to Leonard Uhr. We thank Prof. S.D. Carlson in the Department of Entomology for his support in the research of computerized 3D reconstruction of optic neuropile based on serial electron micrographs and his comments on this paper.

REFERENCES

1. L. Uhr, "Layered Recognition Cone Networks that Processes, Classify and Describe", *IEEE Trans. Computer*, 758-768, 1972

2. L. Uhr, R.J. Douglass, "A Parallel-Serial Recognition Cone System for Perception: Some Test Results", *Pattern Recognition* 11, 29-39, 1979

3. S.L. Tanimoto, T. Pavlidis, "A Hierarchical Data Structure for Picture Processing", *Computer Graphics and Image Processing* 4, 104-119, 1975

4. A.R. Hanson, E.M. Riseman, "Visions: A computer System for Interpreting Scenes", in *Computer Vision System*, ed. A.R. Hanson & E.M. Riseman, 303-334, 1978

5. P.J. Burt, T.H. Hong, A. Rosenfeld, "Segmentation and Estimation of Image Region Properties Through Cooperative Hierarchical Computation", *IEEE Trans. Systems, Man and Cybernetics* SMC-11, No.12, 802-809, 1981

6. S.L. Tanimoto, "Algorithms for Median Filtering of Images on a Pyramid Machine", Technical Report, Univ. of Washington, Dec. 1982

7. E.R. Macagno, C. Levinthal, I. Sobel, "Three-Dimensional Computer Reconstruction of Neurons and Neuronal Assemblies", *Ann. Rev. Biophys. Bioeng.*, 1979. 8:323-351

8. I. Sobel, C. Levinthal, E.R. Macagno, "Special Techniques for the Automatic Computer Reconstruction of Neuronal Structures" *Ann. Rev. Biophys. Bioeng.*, 1980. 9:347-362

9. R.C. Bolles & R.A. Cain, "Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method", *The International Journal of Robotics Research*, Vol.1, No.3, 57-82, 1982

10. A.L. Yarbus, *Eye Movements and Vision*, Plenum, NY, 1967

11. M. Nagao, "Focus of Attention in the Analysis of Complex Pictures Such As Aerial Photographs", in *Real-Time / Parallel Computing*, ed. M. Onoe, K. Preston, A. Rosenfeld, 185-191, 1981

12. K. Preston "Tissue Section Analysis: Feature Selection and Image Processing", *Pattern Recognition* 13, 17-36, 1981