INTERNATIONAL COMPUTER SCIENCE INSTITUTE
1947 Center Street • Suite 600 • Berkeley, California 94704 • (415) 643–4274 • FAX (415) 643-7684

A Network for Extracting the Locations of Point Clusters Using Selective Attention[1]

Subutai Ahmad
International Computer Science Institute
and
University of Illinois at Urbana-Champaign
ahmad@icsi.berkeley.edu

Stephen Omohundro
International Computer Science Institute
om@icsi.berkeley.edu

Technical Report #90-011

May 30, 1990

ABSTRACT

This report explores the problem of dynamically computing visual relations in connectionist systems. It concentrates on the task of learning whether three clumps of points in a 256x256 image form an equilateral triangle. We argue that feed-forward networks for solving this task would not scale well to images of this size. One reason for this is that local information does not contribute to the solution: it is necessary to compute relational information such as the distances between points. Our solution implements a mechanism for dynamically extracting the locations of the point clusters. It consists of an efficient *focus of attention* mechanism and a cluster detection scheme. The focus of attention mechanism allows the system to select any circular portion of the image in constant time. The cluster detector directs the focus of attention to clusters in the image. These two mechanisms are used to sequentially extract the relevant coordinates. With this new representation (locations of the points) very few training examples are required to learn the correct function. The resulting network is also very compact: the number of required weights is proportional to the number of input pixels.

---

1. A version of this report is to be presented as a talk at the 12th Annual Conference of the Cognitive Science Society at MIT, July 25-28th, 1990.

1. Introduction

Consider the visual task of determining whether a set of three point clusters form an equilateral triangle (Figure 1). People are very good at solving this kind of problem, but a connectionist implementation for images with reasonable resolution is not obvious. The difficulties posed by this problem are common to a wide variety of visual tasks and so we have used it as a touchstone against which to test visual neural architectures. In this paper we describe some of the fundamental operations the problem seems to require, biologically inspired implementations of those operations, and a computer simulation which combines them into a complete system.

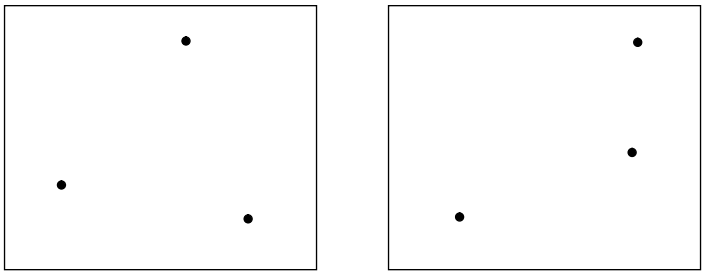


Figure 1 Left triangle is equilateral.The right one is not.

The most straightforward visual neural representations assign a distinct unit to each visual pattern that must be classified. Unfortunately, the space of possible triangles is much too large for this kind of approach to be biologically possible. The optic nerve consists of about a million fibers from each eye, and so it is reasonable to consider square images which are a thousand pixels on a side. Since each of the three vertices can occupy any of these pixels, the total number of possible triangles in such an image is about 1000^6=10^{18}. A brute force representation would require about a million times as many neurons as we have in our entire brain for just this one task. Restricting the units to represent just the set of equilateral triangles would still require about 10^{12} units. If coarse coded representations are used, these numbers can be reduced somewhat, but many more examples will still be needed to learn to properly classify equilateral triangles than is biologically possible. The spatial relationships which define equilateralness will have to be discovered for each position, scale, and orientation of the triangle. Techniques have been proposed for introducing translation and rotational invariance into networks (Giles et. al., 1987) which eliminate the need for independent feature detectors at every location. Unfortunately these methods require that every unit have a large (quadratic) number of connections with complicated weight linkages between them. Furthermore, positional information is lost in these representations - one cannot retrieve the location and orientation of the objects in the image.

These difficulties would disappear if we could directly extract the real valued coordinates of the cluster centers, say in the activations of 6 units. Using this representation it is easy to construct units which compute the distance between a pair of points. The network would then only need to learn a simple classification function - recognizing when 3 numbers are equal. This is the kind of task that simple backpropagation networks have been successful at. The main difficulty, of course, is to transform the representation from a set of pixel values to a set of vertex coordinates.

Ullman (1987) has argued that many high-level visual tasks may be implemented as “visual routines”. The idea is that a high-level system solves a visual task by choosing and combining a set of primitive visual operations. He has suggested some general purpose sequential primitives which might be useful. Among them

are primitives for deciding which portions of the image are relevant, selecting out these sections, and storing their locations for later processing. Given these primitives, a natural solution to the triangle problem is a system which uses a focus of attention to sequentially select the vertices and store their locations.

The visual routines framework is well-suited to our task, but a connectionist solution is non-trivial. This report describes our implementation of primitives for detecting and remembering the locations of clusters of points based on an efficient focus of attention mechanism.

2. Evidence for Sequential Visual Processing in the Brain

There is a large body of psychology literature which supports the idea of sequential visual routines. One of the best examples is the work by Treisman and her colleagues (Treisman & Gormican, 1988). Their work suggests that certain simple visual tasks are performed by people in parallel (response time is independent of the number of objects in the image) whereas other tasks require serial processing (response time is linear in the number of objects). Jolicoeur et al. (1986) have provided further evidence. They find that the time to report whether two stimuli lie on the same curve increases linearly with the distance between them along the curve. In both cases saccadic eye movements are ruled out suggesting that internal processes are responsible for the sequential behavior.

An attentional mechanism which can selectively inhibit or excite regions of the visual input is central to the explanations of the above results (Ullman, 1987, Treisman & Gormican, 1988). Various psychophysical experiments have attempted to pin down the precise characteristics of this mechanism. It was shown in (Posner *et. al*, 1982) that the mechanism was very fast with noticeable effects beginning as soon as 50 msecs after presentations of a cue. They also reported a very flexible scheme for directing the attention to different loci. A shift of attention can be induced by the appearance of a cue, the absence of a cue, or even the *expected onset* of a cue. This suggests that a combination of bottom-up and top-down information is used to decide where to focus attention. (Shulman *et. al*, 1979) have provided evidence that the region of influence moves continuously from one point to another. (Eriksen & Yeh, 1985) have shown that attention is allocated at only one contiguous region in any given instance but suggest that the area of influence can vary continuously in size. There have also been neurophysiological studies showing that some neurons can dynamically change their response properties. Moran & Desimone (1985) report evidence that the sizes and locations of the receptive fields of certain neurons in the monkey visual cortex (areas V4 and IT) change with the task that the animal is trying to accomplish.

3. Implementing a Focus of Attention

Along with these discoveries a number of detailed models have been proposed for implementing selective attention. In the next section we briefly review the most relevant ones and then describe our mechanism.

3.1 Previous work

(Koch and Ullman, 1985) have suggested a model based on a saliency map combined with a pyramidal winner-take-all network. The saliency map consists of a retinotopic grid of neurons which accumulate bottom-up information from various feature maps. Activity of a neuron in this map is proportional to the relevance of the corresponding location. The model then uses a parallel log-depth tree to compute the most salient val-

ue. Interior nodes at each level compute the max of their incoming signals and transmit it to the next level. The time to focus on a spot is thus logarithmic in the number of pixels. Chapman (1990) has extended the model to allow "pointer addressing". To move the focus to an arbitrary location an address is routed from the top of the pyramid down to the appropriate leaf. The value at that leaf is then passed up the tree to the root node.

(Mozer, 1988) has described a model of selective attention which eliminates the need for a log-depth tree. His "attentional mechanism" (AM) consists of a layer of retinotopic units. The layer receives input from all the feature maps. Units in the AM gate the outputs of corresponding units in the feature maps. Thus a single region of active units in the AM corresponds to a single region of activity in the feature maps. A locally competitive rule is used to decide which units in the AM should be active. At each step the current activity of each unit is averaged with the activity of its neighbors minus the activity of all other units. After some iterations the network stabilizes to form a roughly circular spotlight representing the most active region. One advantage of this scheme is that the focus of attention is a circular region instead of a single point[1]. In addition, there is a parameter which can roughly influence the size of the circle. However, since region selection is based on local competition, the time required before the network stabilizes on a single region is quite sensitive to different images. It could be quite large if there are several similar regions of activity which are widely separated in the image.

(Fukushima, 1986) describes a model of selective attention which combines aspects of the above two systems. His system consists of a log-depth hierarchical network in which the output level contains one unit per pattern. Given an image with two or more patterns the output nodes corresponding to the different patterns will respond. The unit with the largest response is selected and a signal is transmitted back through the pathways activated by this particular pattern. This creates a positive feedback loop which sharpens the detection of this pattern. The pathways corresponding to the other patterns gradually attenuate in the absence of this facilitation. Thus the network attends to one of the input patterns. One advantage of this scheme is that the shape of the focus of attention depends only on the shape of the input patterns and thus can be arbitrary. However the process is quite slow since many iterations may be required for the network to settle, and for each iteration the activity has to flow up and down the hierarchy.

3.2 A Constant Time Mechanism for Selective Attention

In all of the above models two fundamentally different operations are bundled together: the mechanism for selecting the location of the focus and the mechanism for suppressing irrelevant regions of the image and allowing relevant portions through. This affects the time to actually shift to a new location as well as limiting the flexibility of the models. A response time of 50 msecs (Posner *et. al*, 1982) only allows 5 to 10 neurons to fire in sequence. Considering the fact that activity may have to flow through several pre-processing layers before reaching the attention system, there doesn't seem to be much time for a relaxation model (such as Mozer's and Fukushima's) to settle or for activity to flow up hierarchies.

In our model we separate these two operations. Our attention mechanism is able to select any circular portion of the input space in two time steps. So given a new location the system is able to efficiently disengage from the previous location and focus on the new one. The locations of interest are determined by external networks and can include top-down as well as bottom up information. (In our current implementation we use a cluster detector to detect the vertices. See Section 4.) In the next few sections we describe the details of our system.

---

1. See (Chapman, 1990) for an implementation which removes this restriction somewhat.

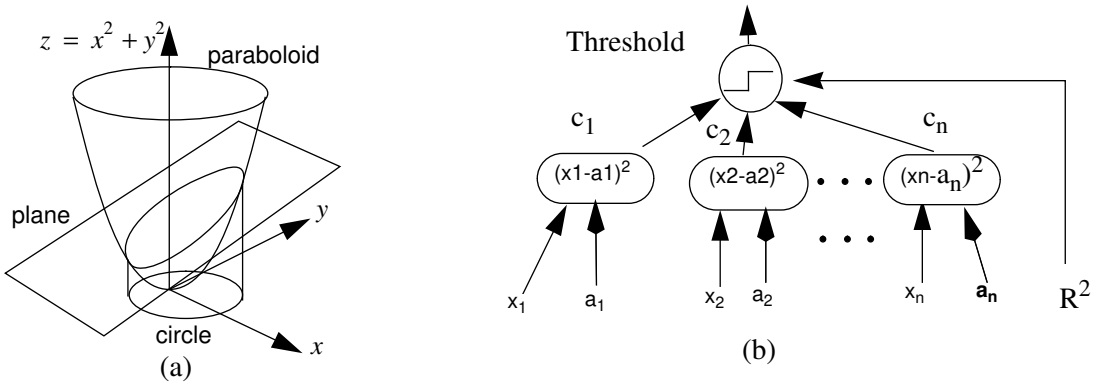


Figure 2 (a) The plane intersects the paraboloid in a curve which projects to a circle. (b) Architecture of threshold unit computing the intersection in n dimensions.

Locally Tuned Receptive Fields

We first describe a mechanism by which linear threshold units can give a localized response in a feature space. The following fact is exploited: if one maps the points in $\Re^{n-1}$ onto the paraboloid defined by $z = \sum_{i=1}^{n-1} x_i^2$, then the intersection of a hyperplane in $\Re^n$ with this paraboloid projects onto a sphere in $\Re^{n-1}$. Thus there is a mapping between planes in $\Re^n$ and spheres in $\Re^{n-1}$. To select a set of points which lie within a sphere in some space one just has to project the points onto the paraboloid and slice it with the plane corresponding to the sphere. Points which lie "beneath" the plane are within the sphere. Figure 2 illustrates this for $\Re^2$. Notice that the computation of a threshold unit is exactly that of deciding on which side of a hyperplane an input point lies. To encode circular receptive fields with threshold units, you just need to include an extra input: the sum of the squares of all the other inputs. An equation of the form:

$$-(\sum w_i x_i + \sum x_i^2 + const) > 0 \tag{1}$$

will be positive if $\vec{x}$ lies within a spherical volume determined by the weights and constant.

The above method creates circular receptive fields with hard boundaries. A smooth boundary with a flat top may be obtained by using a sigmoid instead of a threshold function. The steepness of the sigmoid (its gain) will then control the steepness of the receptive field boundary. Non-circular receptive fields are also possible by changing the nature of the non-linearity. Elliptical receptive fields may be obtained by using the paraboloid $z = \sum_{i=1}^{n-1} c_i x_i^2$ where c_i denotes the amount of stretching along each axis. In principle arbitrary shapes can be obtained by appropriately choosing the non-linearity.

Dynamic Receptive Fields

In addition to being able to select a portion of the input space, we need the ability to shift the location and size of the receptive field around quickly in response to changing demands. In our model there are two ways to do this. The first method involves changing the slope of the hyperplane. In Figure 2 (a) note that as the slope increases the center of the projected circle will shift to the right. For any sphere it is possible to compute the coefficients of the hyperplane which produces that sphere. Given a plane $\vec{m} \cdot \vec{x} = \vec{m} \cdot \vec{c}$ where $\vec{m}$ and $\vec{c}$ are real-valued vectors, the projection of the intersection of the plane with the paraboloid is a sphere whose center is:

$$(a_1, a_2, \ldots, a_{n-1}) = \left(-\frac{m_1}{m_n}, \ldots, -\frac{m_{n-1}}{m_n}\right) \tag{2}$$

and whose radius is:

$$R = \frac{\sqrt{m_1^2 + m_2^2 + \ldots + m_{n-1}^2 + 4m_n(\vec{m} \cdot \vec{c})}}{2m_n} \quad (3)$$

In a threshold unit, changing the slope of the hyperplane corresponds to changing the weights of the inputs. One of these units could eventually "learn" the correct position of its receptive field, however the time scale for weight changes is too slow for dynamic computations.

Another way to alter the sphere is to fix the plane but shift the paraboloid, by computing $z = \sum_{i=1}^{n-1} c_i (x_i - a_i)^2 + r^2$. This moves it a distance a_i along dimension i (changing the location of the sphere) and a distance r^2 along the z-axis (changing the radius of the sphere). If the a_i's and r^2 are available as input then the receptive field can be changed an arbitrary amount in one time step. Figure 2(b) shows how such a unit would be configured. For each input dimension there is a sub-unit which computes the square. ((Suarez & Koch, 1989) present a neurally plausible mechanism for computing a quadratic.) The outputs of these units are fed into a threshold (or sigmoid) unit. The net effect is that the threshold unit will respond only when the input vector $\vec{x}$ lies within the spherical receptive field determined by $\vec{a}$ and r.

Focus Of Attention With Value Coded Units

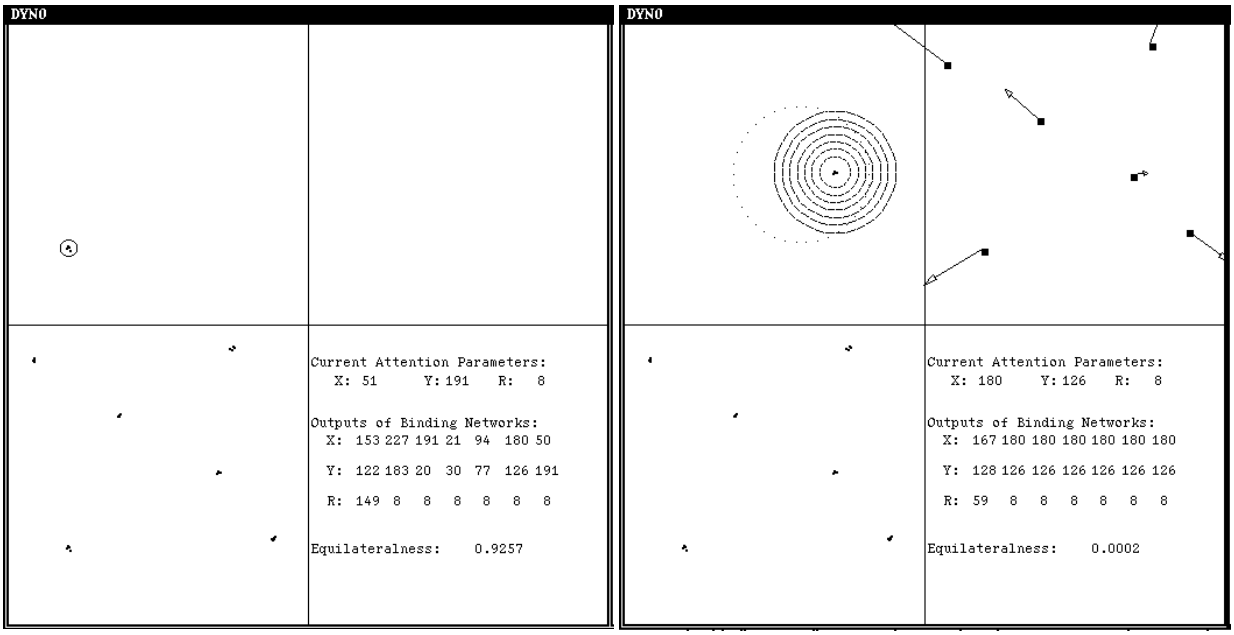
So far we have assumed an n-dimensional input space that is encoded as n analog signals. In our triangle task however, we have to implement a circular patch in a 2-dimensional retina. The units in this representation are laid out on a flat sheet, with each unit explicitly encoding a point in the space.

To create dynamic receptive fields here, we construct a gating layer, with one "gate unit" per input unit. Each gate unit receives three global inputs: A_x, A_y, and A_r representing the parameters of the focus of attention. The gate units are basically the same as the localized units described above with one small modification: x_1 and x_2 are fixed for each gate unit. To encode this we have two separate connections from the input unit to the gate unit. The weights of these connections are x_1 and x_2. If the input unit fires, the threshold unit will fire only when its input unit is within the circle determined by A_x, A_y, and A_r. The effect is a layer of units which filters the input image according to a global control signal. The system can select any circular portion of the image in one time step. The hardware required to implement this is minimal: 8 extra connections per input unit. It is also fairly easy to extend our mechanism to allow foci of different shapes once the parameters are available.

Figure 3 shows the graphical output of our simulator for two different images. In each display, the lower left quadrant displays the current input image (3 point clusters). The upper left quadrant shows the output of the gate units. The circle shows the implicit focus of attention represented by the 3 parameters. In Figure 3 (a) note that only the activity within the focus is allowed to propagate.

4. Deciding Where To Focus ... And How To Get There.

A focus of attention system is useless without a mechanism to select the interesting locations. In general, many different criteria can be used to define "interesting". One possibility is a bottom up method which depends on the characteristics of the current image (i.e. choosing the brightest image point). The location could also be chosen in a top-down fashion (i.e. as a result of prior expectations). In our domain, the natural criterion is the cluster with the largest number of points. Note that our attention mechanism itself is independent of the particular criterion used - all it requires are the coordinates of the location.



(a) (b)

Figure 3 Examples of the system behavior for a 256x256 image with seven clusters of points. In both displays, the lower left quadrant shows the image; the upper left quadrant shows the output of the gate units. The error vectors are displayed in the upper right and the outputs of selected units are shown in the lower right. (a) shows a snapshot of the system with the focus of attention near one of the clusters. (b) shows the dynamic scaling behavior as the focus tries to fit the cluster of points within it.

Our method uses a coarse grid of "error" units. Encoded in each location in this grid is a value representing its importance and values encoding the discrepancy between the current focus of attention and the cluster within its receptive field. The most active grid location thus provides a rough estimate of the next location to visit. Once attention has been directed to that location, it is fine tuned to settle exactly on the center of mass of the cluster. These two systems are described in detail in the following two sections.

4.1 Fine Tuning the Focus of Attention

We first describe a mechanism for fine tuning A_x, A_y, and A_r to settle on the center and size of the cluster of points within the current focus of attention. The center of mass, (C_x, C_y), is defined as the average of the x and y coordinates of the active points:

$$C_x = \frac{\sum_i X(i)\, a_i}{\sum_i a_i} \text{ and } C_y = \frac{\sum_i Y(i)\, a_i}{\sum_i a_i} \tag{4}$$

where $X(i)$ and $Y(i)$ denote the x and y coordinates of the i'th unit and a_i denotes its activity. $\sum a_i$ can be computed by a unit which receives input from all gate units with a weight of 1 (unit 1 in Figure 4). To compute the numerators we include two units with links to every gate unit (units 2 and 3). The weights from the i'th gate unit to each of these two units are $X(i)$ and $Y(i)$, respectively. A weighted sum of their incoming activity thus computes $\sum X(i)\, a_i$ and $\sum Y(i)\, a_i$. Two units (units 4 and 5) perform the division to obtain C_x and C_y. These values are fed into two more units which compute the difference between the center of mass and the attention parameters: $(C_x - A_x, C_y - A_y)$. The units representing $A_x$and A_y receive as input this difference as well as their own output. By computing the sum of the two inputs, they continually update the focus to the center of mass of the points within it.

To get an estimate of the size of the cluster we also include a unit which continually adjusts the size of the

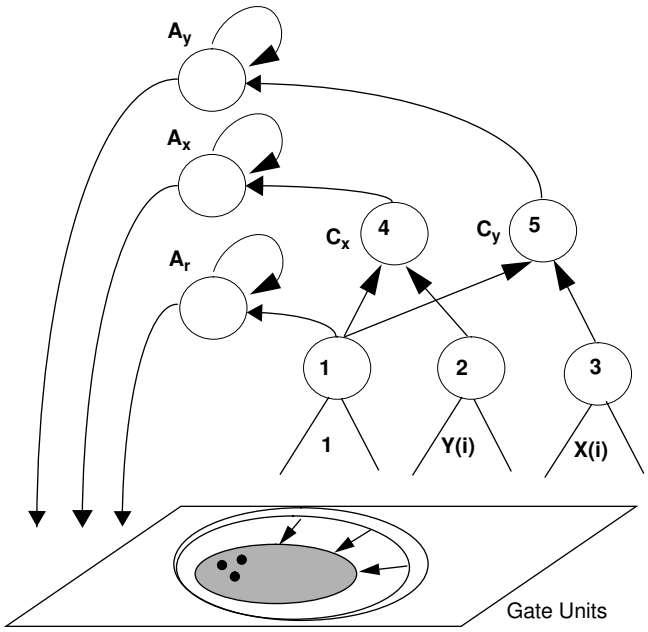


Figure 4 The units which continually fine tune the focus of attention. See text for explanation.

focus of attention to match the size of the set of points within it. As long as $\sum a_i$ remains constant, the scaling unit decreases A_r by a small amount. If the sum decreases, indicating that the scale has become too small, the unit increases A_r slightly and stops.

Figure 3 (b) illustrates this scheme on our simulator. The focus of attention is initialized to contain all three clusters and be slightly off center (dotted circle). The set of concentric bands show successive steps as the focus of attention decreases in size and shifts its location to fit the cluster inside.

4.2 Error Units

The mechanism described above does not give us a way to fixate on clusters outside the focus. To do this we include a coarse grid of units each of which receives input from a small patch in the image. At each grid location there are three outputs. The first two outputs encode the difference between the center of mass of the cluster of points within their receptive fields and the point (A_x, A_y). In this way each grid location encodes an "error" vector for adjusting the focus of attention. These error vectors are continually updated to compensate for changes in A_x and A_y. The third unit represents a confidence value from 0 to 1, indicating the importance of its location. By adding the vector with the highest confidence value to A_x and A_y, the focus of attention can be shifted to the location of the most salient cluster. The focus can be shifted to the nearest location by selecting the location with the smallest error vector.

The error vector representation was inspired by discoveries of a similar mechanism in the monkey superior colliculus for controlling eye saccades (Sparks, 1986). The output of the confidence units are similar in spirit to the saliency map in (Koch & Ullman, 1985). In general many factors may contribute to the saliency of a given location. In our implementation the saliency is simply the number of active points within the receptive field passed through a sigmoid.

The upper right quadrant of Figure 3 (b) show some example error vectors. (Only those locations whose confidence value is higher than 0.2 is displayed.) Each arrow represents the error vector for that location.

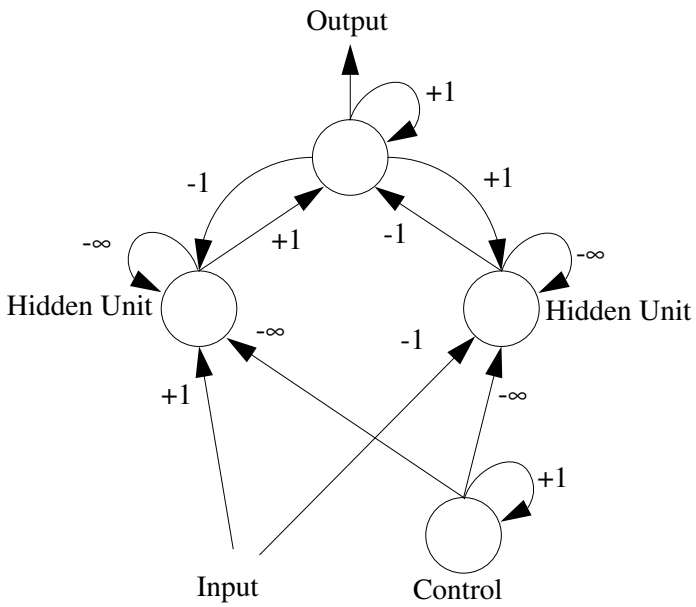


Figure 5 Schematic of a binding network. The network continually tracks its input signal until a control signal is sent, at which point the output is frozen to be the current signal.

The shaded square represents the confidence value - the darker the square the higher the confidence.

5. Storing Locations

As the network visits each vertex it should store the values of A_x, A_y, and A_r whenever the focus of attention stabilizes. We accomplish this with small recurrent networks for each value that needs to be stored. Each of these "binding" networks continually tracks a particular unit (one of A_x, A_y, and A_r) until a control signal is sent, whereupon it freezes the output to be the current value of the unit. This is done by with the network shown in Figure 5. While the control unit is off, the hidden units compute the difference between the value of the assigned unit and the current output and sends it to the output unit. The output function of the hidden units is linear while its net input is positive, zero otherwise. The left hidden unit indicates when the output should be decreased whereas the right unit indicates when it should be increased. When the control unit is turned on, the hidden units are shut off by large negative weights. An excitatory link from the control unit to itself ensures that once the control unit has fired, it stays on, preventing further adjustments. Three of these "binding networks" are used for each set of parameters that are stored.

There is one other issue to consider. As the network fixates on successive clusters, we would like different sets of binding networks to be instantiated. To do this we need some sort of a sequencing mechanism which will send control signals to successive binding networks. This is accomplished by the network shown in Figure 6. The signal unit fires when the focus of attention has stabilized for three iterations. On successive firings of the signal unit, appropriate control units are turned on. (The unit "cntrl-i" in the figure corresponds to the control signal for the i'th binding network.) Figure 8 (b) shows an example of this. The bottom left quadrant shows the output of seven sets of three binding units, each representing the scale, x and y coordinates of the focus of attention. For example the first set of units has been frozen to (161,116,114) whereas the rest of them are still free to follow the attention parameters. If a control signal were to be generated now, the second set would be frozen at (174,29,8) to represent the current parameters Figure 8 (c).

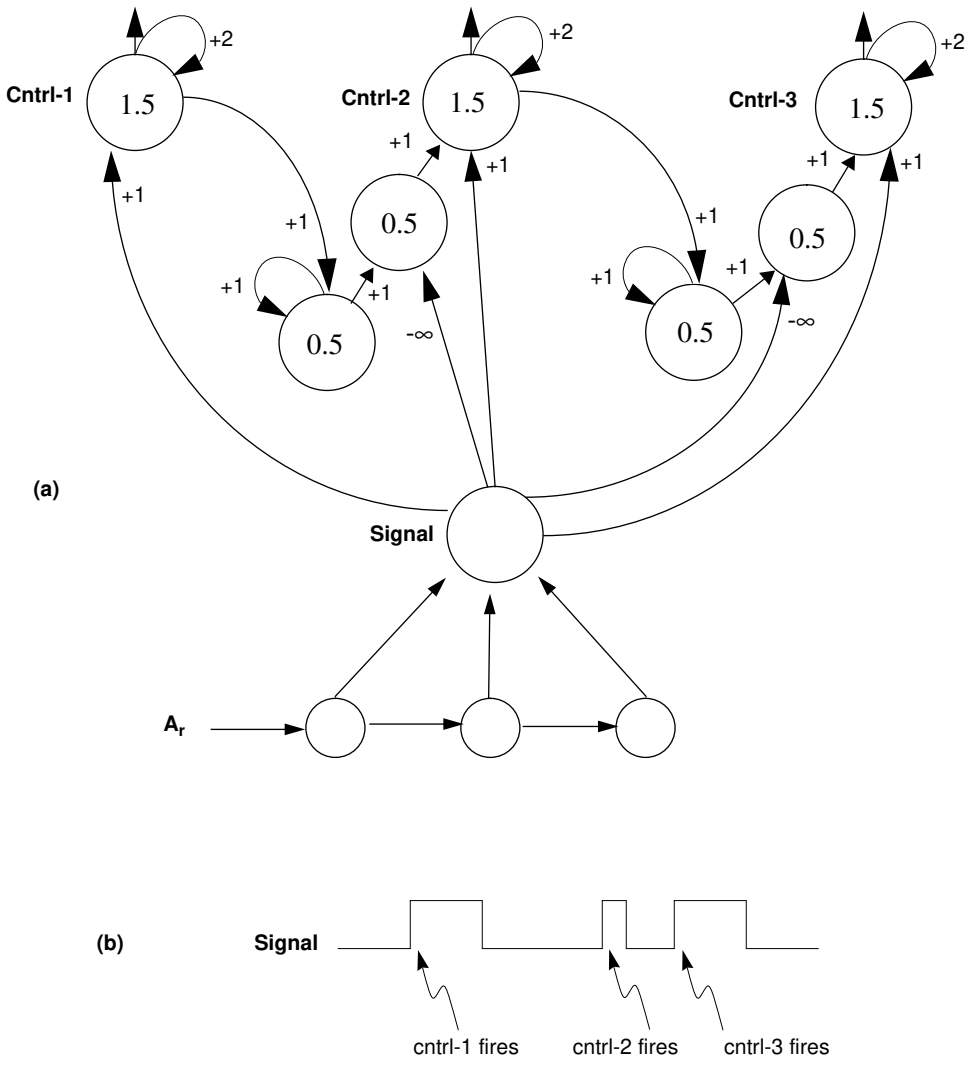


Figure 6 A detailed diagram of a portion of the sequencer. The units shown above sends control signals to the appropriate binding networks. The signal unit fires when all of its inputs are equal. The first time it fires, cntrl-1 starts firing. The next time, cntrl-2 fires, and so on. This structure is used to send control signals to successive binding networks.

6. System Architecture

Figure 7 shows a schematic of the whole architecture. The module which controls the attention field is an autonomous network that continually attempts to wrap the focus around the points within its field of view. When it has stabilized, the signal unit fires, a control signal is transmitted to the next binding network and the focus of attention is updated to the location of the next cluster.

To sequentially process each cluster in the image the system has to repeatedly select the largest confidence value, inhibit the corresponding unit, and send the error vector to the system controlling A_x and A_y. One way to do this is to construct a winner-take-all network with competing confidence units such that the system settles into a state where only one unit is active. However these networks can take a relatively long time to settle, especially when the competing values are very similar. It is also quite difficult to find the correct set of inhibitory weights to create a robust winner-take-all network. Another alternative is to construct a log-

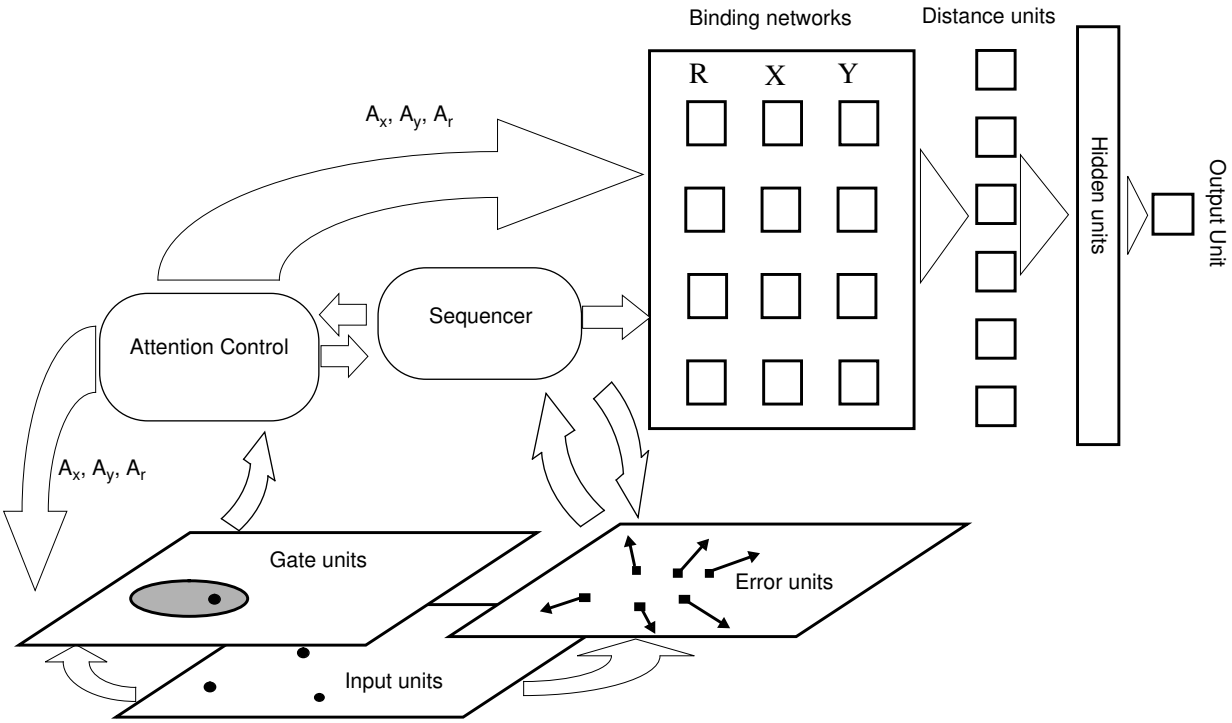


Figure 7 Basic system architecture. The module "Attention Control" continually tries to fit the focus of attention to the points within it. The "Sequencer" updates the focus of attention to visit all the clusters in sequence and also sends the control signals to the binding network to store successive locations.

depth network as in (Koch and Ullman, 1985) to explicitly compute the maximum. In our current implementation we assume a unit with a built in max function but are studying other schemes for performing this computation efficiently.

Assuming that the system starts with a focus of attention covering the entire image plane, the network first wraps the focus around the triangle and then sequentially visits and stores the locations of the three vertices. Once this is done, the first set of bindings encode the position and scale of the triangle. The next nine bindings encode the positions and scales of the three vertices in the order that they were processed. A set of distance units then explicitly computes the six possible distances between the first four stored locations. A standard feedforward network with one layer of hidden units is used to compute the final output.

7. Simulation

For the simulations in this paper we used 256x256 images. The complete network consisted of 131,912 units, 131,072 of which were the input and gate units. We generated a training set consisting of random triangles (approximately 50% of which were equilateral) with Gaussian noise added around each vertex. For each triangle the focus of attention was initialized to cover the entire image plane. The system was allowed to run until all the binding units were frozen. The outputs of the first 4 binding units were fed to 6 distance units whose outputs were then used as inputs to train the backprop network. The teacher signal was gener-

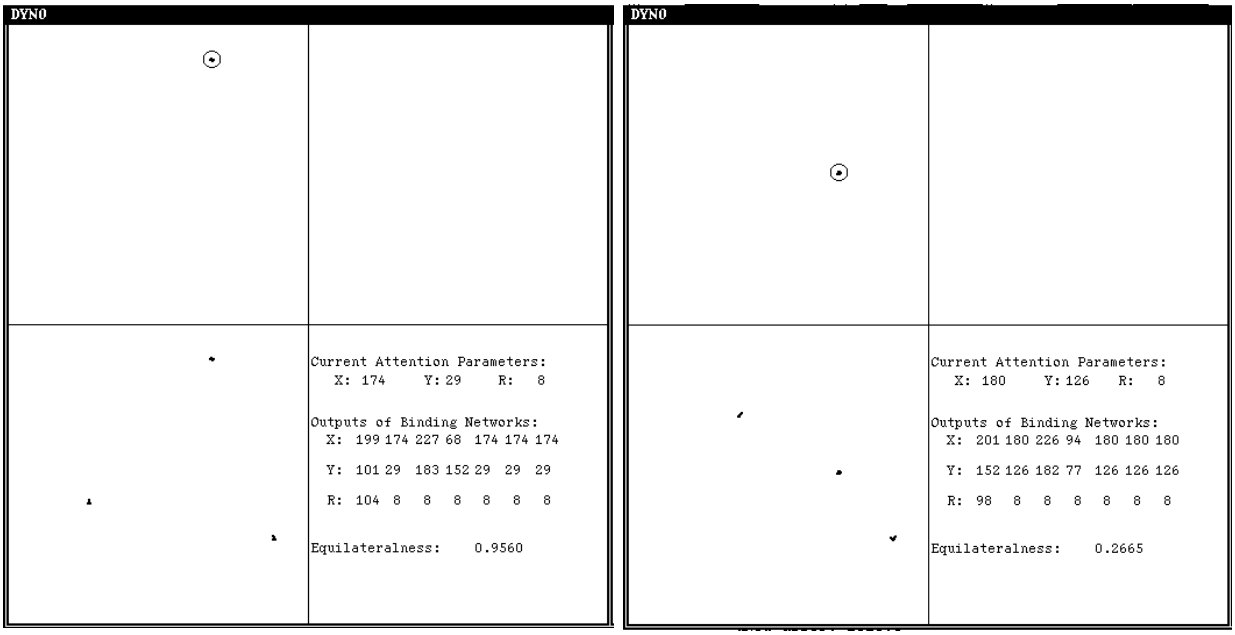


Figure 9 Results of parsing an equilateral triangle and a non-equilateral triangle. (Only the first four binding units are used here.)

ated according to:

$$1 - \frac{|l_1 - l_2| + |l_2 - l_3| + |l_1 - l_3|}{l_1 + l_2 + l_3} \tag{5}$$

where l_i is the length of the i'th side. This is a function which is 1 for equilateral triangles and degrades gradually to 0 as the triangles deviate from equilateralness. With a training set of only 100 triangles the network score was consistently greater than 0.9 for equilateral triangles on independent test sets. Note that, since the inputs to the feed-forward network are relational, the number of training examples necessary for good generalization does *not* depend on the size of the image.

Figure 8 shows a series of images from our simulator at several stages of a typical recognition sequence. Figure 9 shows the state of the network after parsing two different triangles. The system correctly classified the left triangle as being equilateral and the right one as not being equilateral. The outputs of the binding networks show the vertex coordinates that were discovered by the network.

8. Discussion

We have described a network for extracting the locations of point clusters. Although the system is geared towards a narrow class of images, it is tempting to speculate how these mechanisms would fit into a general purpose vision system. The focus of attention mechanism itself is independent of the actual image and could work in any real-valued space. The system should be easily adaptable for general visual search of the type discussed in (Treisman & Gormican, 1988) which requires the ability to isolate regions in different feature maps. Minimal hardware is required to adapt our system work with orientation selective neurons, color detectors, etc. The notion of what locations are "interesting" would also need to be modified to include clusters in arbitrary feature maps.

The above ideas would represent relatively straightforward extensions to the system, but there are some

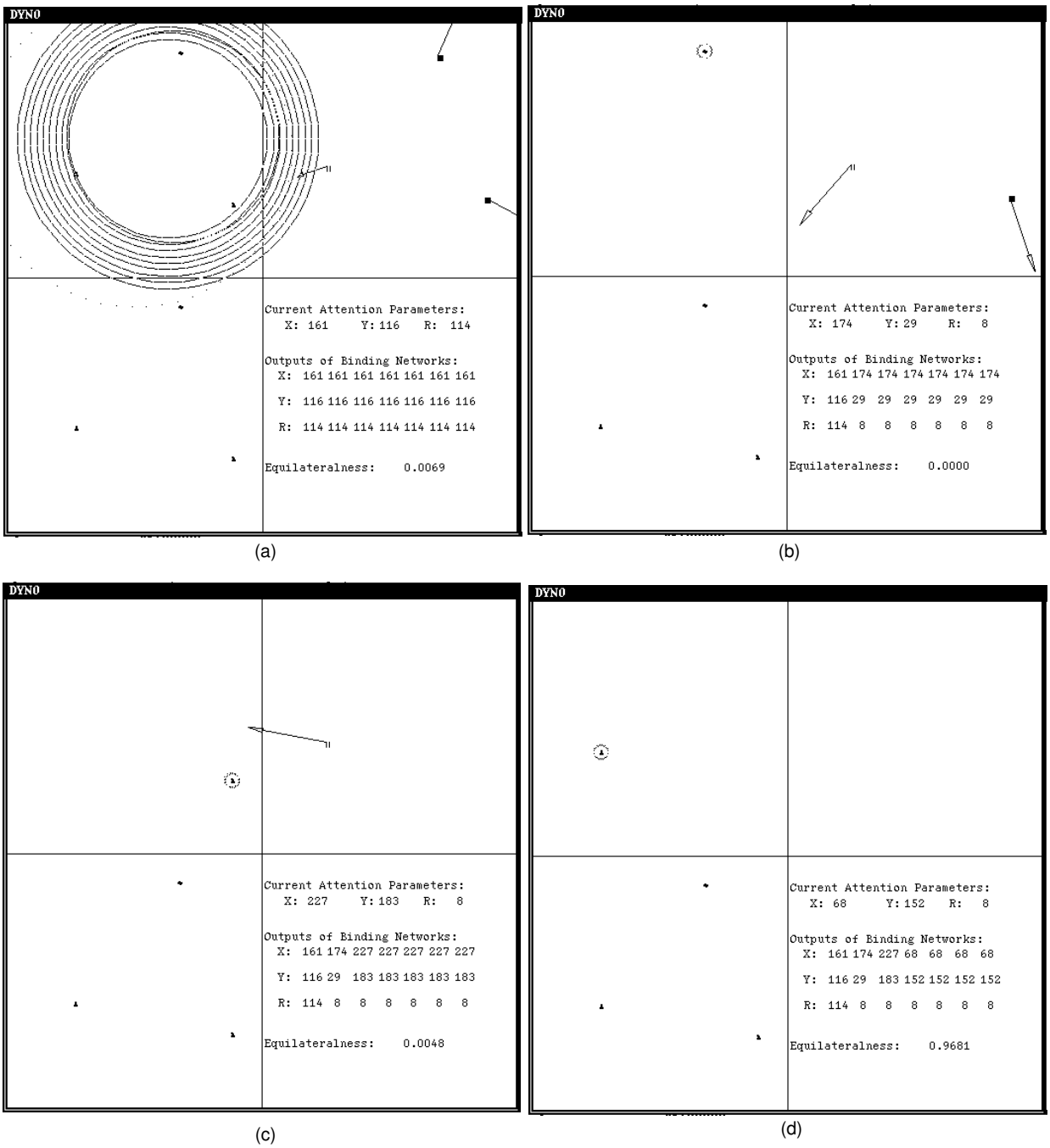


Figure 8 Four steps in a typical run. (a) Determining the size and location of the whole triangle. (b) - (d) Network after fixating on each of the three vertices. Note that the binding networks are updated correctly. Once all four positions are available, the network correctly classifies the triangle to be equilateral (bottom right of (d)).

more difficult issues which also have to be dealt with. The library of primitives must be expanded to handle more features of realistic images. We would need to extract relations based on curves, regions, shapes, etc., and so would need the appropriate primitives. For example, (Jolicoeur et. al, 1986) have provided evidence that people use a "curve-following" routine to determine whether two points lie on the same curve. Finally, a major issue that we have not addressed is the issue of compiling multiple visual primitives to accomplish a dynamically specified task.

In conclusion, a central point of this paper has been to demonstrate mechanisms for performing sequential visual computations in connectionist networks. We have described an efficient implementation of some primitives, and have used them to extract image properties that are inefficiently represented in parallel. There is evidence from psychology and neurophysiology that biological organisms actually implement similar routines at an early processing level.

9. Acknowledgments

We thank Jerome Feldman for suggesting equilateral triangles as an interesting domain to study. We would also like to acknowledge Chedsada Chinrungrueng, David Chapman, Christof Koch, Terry Regier, and Andreas Stolcke for helpful discussion.

10. References

Ahmad, S., and Omohundro, S. (1990) Equilateral Triangles: A Challenge for Connectionist Vision. To appear in: Proceedings of the 12th Annual Meeting of the Cognitive Science Society, MIT, 1990.

Chapman, D. (1990). *Vision, Instruction, and Action*. Ph.D. Thesis, Massachusetts Institute of Technology. Also MIT AI-Lab Technical Report #1204.

Eriksen, C.W. and Yeh, Y. (1985). Allocation of Attention in the Visual Field. *Journal of Experimental Psychology: Human Perception and Performance*, **11** (5), pp 583-597.

Fukushima, K. (1986) A Neural Network Model for Selective Attention in Visual Pattern Recognition. *Biological Cybernetics*, **55**, pp 5-15.

Giles, C.L., Griffin, R.D., & Maxwell, T. (1987). Encoding Geometric Invariances in Higher Order Neural Networks. In "Advances in Neural Information Processing", David Touretzky, Ed. Morgan Kaufmann.

Jolicoeur, P., Ullman, S., and Mackay, M. (1986). Curve tracing: A possible basic operation in the perception of spatial relations. *Memory and Cognition*, **14** (2), pp 129-140.

Koch, C. and Ullman, S. (1985) Shifts in selective attention: towards the underlying neural circuitry. *Human Neurobiology*, **4**, pp219-227.

Minsky, M. & Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry.* MIT Press, Cambridge, MA.

Mozer, M. (1988). A Connectionist Model of Selective Attention in Visual Perception. University of Toronto Technical Report CRG-TR-88-4

Posner, M.I., Cohen, Y., and Rafal, R.D. (1982). Neural Systems Control of Spatial Orienting. *Phil. Trans. R. Soc. Lond.* ***B*** **298**, pp287-298.

Sparks, D. L. (1986). Translation of Sensory Signals into Commands for Control of Saccadic Eye Movements: Role of Primate Superior Colliculus, *Physiological Reviews,* **66** (1).

Suarez, H., & Koch, C. (1989). Linking Linear Threshold Units with Quadratic Models of Motion Perception. *Neural Computation*, **1** (3), pp 318-320.

Treisman, Anne, and Gormican, Stephen. (1988) Feature Analysis in Early Vision: Evidence from Search Asymmetries. Psychological Review, **95** (1).

Ullman, S. (1984) Visual Routines. *Cognition*, **18**, pp 97-159.