International Journal of Computer Vision 25(2), 109–143 (1997)

Learning Recognition and Segmentation Using the Cresceptron

JOHN (JUYANG) WENG
Department of Computer Science, Michigan State University, East Lansing, MI 48824 USA

NARENDRA AHUJA AND THOMAS S. HUANG
Beckman Institute, 405 N. Mathews Avenue, University of Illinois, Urbana, IL 61801 USA

Received March, 1996; Accepted March, 1996

Abstract. This paper presents a framework called Cresceptron for view-based learning, recognition and segmentation. Specifically, it recognizes and segments image patterns that are similar to those learned, using a stochastic distortion model and view-based interpolation, allowing other view points that are moderately different from those used in learning. The learning phase is interactive. The user trains the system using a collection of training images. For each training image, the user manually draws a polygon outlining the region of interest and types in the label of its class. Then, from the directional edges of each of the segmented regions, the Cresceptron uses a hierarchical self-organization scheme to grow a sparsely connected network automatically, adaptively and incrementally during the learning phase. At each level, the system detects new image structures that need to be learned and assigns a new neural plane for each new feature. The network grows by creating new nodes and connections which memorize the new image structures and their context as they are detected. Thus, the structure of the network is a function of the training exemplars. The Cresceptron incorporates both individual learning and class learning; with the former, each training example is treated as a different individual while with the latter, each example is a sample of a class. In the performance phase, segmentation and recognition are tightly coupled. No foreground extraction is necessary, which is achieved by backtracking the response of the network down the hierarchy to the image parts contributing to recognition. Several stochastic shape distortion models are analyzed to show why multilevel matching such as that in the Cresceptron can deal with more general stochastic distortions that a single-level matching scheme cannot. The system is demonstrated using images from broadcast television and other video segments to learn faces and other objects, and then later to locate and to recognize similar, but possibly distorted, views of the same objects.

Keywords: visual learning, face recognition, face detection, object recognition, object segmentation, feature selection, feature extraction, shape representation, self-organization, associative memory

1. Introduction

The image appearance of a real-world scene depends on a series of factors: illumination, object shape, surface reflectance, viewing geometry, and sensor type. In real-world situations, all these factors change frequently and most of them are unknown and uncontrollable, making computer vision difficult (Pavlidis, 1992), especially in general real-world settings.

The work was supported by the Defense Advanced Research Projects Agency and the National Science Foundation under grant IRI-8902728, NSF under IRI-9410741, Office of Naval Research under N00014-95-1-0637 and N00014-96-1-0129.

1.1. Approaches to Vision

Currently prevailing approaches to computer vision rely on human designers to hand-craft a set of rules for a specific task and then to explicitly code these rules into a program. The task of recognizing 3-D objects from 2-D electro-optical images has been found very difficult in a general setting. The approaches,

using either 2-D or 3-D data, include model-based geometric reasoning (e.g., Brooks, 1981), model-based specific-feature grouping (e.g., Lowe, 1985; Chen and Kak, 1989; Sato and Binford, 1992) or alignment (e.g., Faugeras and Hebert, 1986; Huttenlocher and Ullman, 1987), constrained search (e.g., Grimson and Lozano-Perez, 1984), feature based hashing (e.g., Lamdan and Wolfson, 1988; Stein and Medioni, 1992), weights based evidence accumulation (e.g., Jain and Hoffman, 1988), invariants based techniques (e.g., Forsyth et al., 1991; Gool et al., 1991), combination of different techniques (e.g., Bichsel, 1991). Some work has been done in automatic generation of model-based search trees (e.g., Ikeuchi and Kanade, 1988; Hansen and Henderson, 1989; Arman and Aggarwal, 1991).

In the development of such an approach, the human designer is responsible for designing detailed searching strategies and matching criteria. Restricted by the tractability of manual algorithm design, various conditions must be imposed, including type of features to be used, type of surface or shape to be seen, lighting condition and viewing geometry. Although the resulting algorithms can be efficient under the required conditions, it appears very difficult to apply this type of approach to complex real world scenes with virtually unlimited object types and background. Among others, it is difficult to *automatically* verify whether the required conditions have been satisfied.

In contrast, human vision capability is so versatile. For example, as long as the visual image of an object bears a large degree of resemblance to what one has seen, one can recognize the object. It is also known that learning plays a central role in the development of such a capability in humans and it takes place over a long period. Human vision appears to be more a process of learning and recalling rather than one relying on understanding the physical processes of image formation and object-modeling. If what has been learned by seeing is very different visually from what is currently being presented, the human recognition becomes very difficult. For example, "Thatcher's illusion" (Thompson, 1980) indicates that facial expression is very difficult to recognize from an upside-down face, although it would be quickly revealed by a simple "mental" rotation if the brain actually performed such a rotation. The evidence for appearance learning in vision includes even low-level vision. For instance, it has been demonstrated that a common visual experience, overhead light source, is learned and used to perceive shape from shading (Ramachandran, 1990), although the solution to the problem is not unique from the image formation point of view.

The work presented in this paper is motivated by the need to recognize objects from real world images. It seems intractable to hand-craft a set of rules that are sufficient to deal with general vision problems in our complex real world. Thus, we emphasize learning ability *in general settings*—learning visual appearance of objects from examples. Humans as designers need only to provide a good structure for learning, but they are relieved of most design details. With a good system structure for learning, we can use the abundance of visual information available in the real world to develop a vision system through training, instead of attempting developing a system based on human designer's capability to convert his knowledge about visual information processing into vision rules.

The idea of learning for vision is not new. It is the message that comes through most clearly from the work in psychology, cognitive science and neurophysiology (e.g., Hubel, 1988; Anderson, 1990; Ramachandran, 1990; Martinez and Kessner, 1991). The question is how to do computational learning. This paper presents a new framework which has several desirable characteristics for machine learning from general real world images. We first briefly overview some past works on learning.

1.2. Learning Techniques

Most traditional learning techniques are devoted to data classification in that an instance of object to be classified is already represented by a feature vector. The problem of learning there is to determine how to assign a label to a vector, based on training data in which each vector has a given label. The major techniques include parameter estimation based on Bayes decision theory (e.g., Keehn, 1965), non-parametric k-nearest-neighbor rule (e.g., Loftsgaarden and Quesenberry, 1965; Cover and Hart, 1967), linear discriminant functions (e.g., Highleyman, 1962), clustering techniques (e.g., Cover, 1969; Jain and Dubes, 1988), and syntactic methods (e.g., Fu, 1968; Pavlidis, 1977). Although symbolic learning in the machine learning community aims at constructing a description of rules from train ing samples, feature vectors are also predefined by human experts (ID3, (Quinlan, 1986), CART (Breiman et al., 1984), AQ15 (Michalski et al., 1986)).

However, extraction of objects from images of a cluttered scene is a major task. If a human is available to segment the objects of interest from images, then why not let her or him do the entire recognition! Segmenting an object from the background is not necessarily easier than recognizing it, and the two are not independent. If feature vectors are provided for the entire image without identifying which features belong to a single object, no traditional learning technique will work. A recognition method that also deals with segmentation must work directly from the retinotopic data (i.e., each item represents a sensory position in the retina).

The Neocognitron system developed by Fukushima and his colleagues since the early 70's (Fukushima, 1975; Fukushima et al., 1983) was designed for recognizing a small number of segmented patterns such as numerals and alphabets, directly from binary images. Their idea of grouping low-level features to form high-level features in a multi-level structure of retinotopic planes of neurons is very useful for visual learning. In the field of computer vision, a few published works performed learning directly from images, although the task of segmentation has not been directly addressed. Pomerleau's work on ALVINN (Autonomous Land Vehicle in a Neural Network) (Pomerleau, 1989) used a neural network to learn, directly from intensity images and range images, the mapping from the sensory input to the heading direction. The performance of this neural-network-controlled CMU NAVLAB in road following is at least comparable to that achieved by the best traditional vision-based autonomous navigation algorithm at CMU. Turk and Pentland (1991) applied the method of principal component analysis directly to normalized intensity images for face recognition. The work reported here, started in 1990, aims at learning to recognize and segment of wide variety of objects directly from cluttered natural images. Some partial preliminary results of the work reported here were presented during 1992–1993 at a few conferences (Weng et al., 1992, 1993; Weng, 1993).

In dealing with learning and segmenting directly from images, the predictability and efficiency are two central issues. Recently, there has been a surge in interest in learning using models of artificial neural networks (or connectionist models of computation). However, the performance of a neural network trained by existing methods (e.g., the back-propagation method Rumelhart, 1986) are not predicatable, due to the problem with local minima. Poggio and Edelman (1990) used a generalized Gaussian radial basis function to approximate the mapping from a 2-D view of a set of points to a standard view, assuming that the points are from a single object. Such a flat network improves the predictability of the system at trained points. However, the existing neural network methods lack a capability of dealing with complex and large-size problems, since sharing of knowledge among different objects is limited by the fixed network structure.

1.3. The Challenges

Although the use of neural networks has shown encouraging results in some pattern recognition and vision problems, it is not clear whether this approach can handle complex real-world recognition-and-segmentation problems for which a retinotopic network is needed. There is a lack of systematic treatment of the retinotopic network structure, and the theory for such neural networks is missing. A neural network is often treated as an opaque box (instead of local to global analysis) and its learning is often formulated as an optimization problem with a huge number of parameters, which leads to unpredictable system performance. For example, if a pattern is learned at one position on the retina, it is not guaranteed that the exact same pattern can be recognized in all the other positions. It is not clear how to systematically deal with generalization from training exemplars. In order to handle the complexity of vision problems, we have identified the following requirements:

- The system must be able to learn many detailed low-level rules that humans (practically) cannot manually specify. Learning should not be limited to the parameters of hand-crafted rules, because a fixed set of rules is not scalable to complex problems.
- Feature representation must be automatic. It is intractable to manually define the feature represented by every neuron. Significant image structures, or features, must be automatically identified, and their breakdown and mapping to the framework must be automatic.
- Learning result must be reliable. The unpredicatable performance as with back-propagation must be avoided.
- Learning must be predictable. The size of a network for a non-trivial vision task has got to be large. Repeated modification of all weights such as done in a typical back-propagation algorithm is impractical.

- Learning must be incremental. An addition of a new object to be learned should not require the entire network to be re-trained. This is a key towards a self-improving complex vision system.
- The method must work for unsegmented input. It is often impractical to require presegmented data.
- The dimension of feature space should not be fixed. Due to the complexity of the problems we are dealing with, a fixed number of features will lead to a failure of recognition when the number of objects becomes very large. The system should be able to memorize and utilize different features for different objects.

In this work, we are not addressing higher-level visual learning issues, such as learning to infer 3D shapes from 2-D images, learning from mistakes, learning to identify discriminant object parts and ignore irrelevant parts, etc. Our goal here is to recognize and segment image parts based on similarity of visual appearance.

1.4. The Cresceptron

Our framework is called *Cresceptron*, coined from Latin *cresco* (grow) and *perceptio* (perception). Like the Neocognitron, this framework uses a multi-level retinotopic layers of neurons. However, it is fundamentally different from the Neocognitron in that the actual network configuration of the Cresceptron is *automatically* determined during learning, among other structural differences. The following are some salient features of the Cresceptron which contribute to the satisfaction of the above mentioned requirements.

1. The Cresceptron performs retinotopic learning through hierarchical image *analysis* based on hierarchical structural features derived therefrom (see Fig. 1). The learning phase requires human to provide segmented images and to provide a class label for each. Unlike conventional learning, learning in the Cresceptron is incremental. New features are detected and the network structure appropriately created to relate new features with previously learned features. Feature-grouping sharing occurs automatically at every level of the network which keeps the network size limited.
2. Tolerance to deviation is made hierarchical, smaller at a lower level and larger at a higher level. This makes it possible to handle many perceptually similar objects based on a relatively small set of training exemplars.
3. Learning in the Cresceptron is based on hierarchical (i.e., local-to-global) *analysis* instead of back-propagation. The structure of the object is analized in a hierarchical, bottom-up fashion before a decision is made. Therefore, the network is not an opaque box[1]. This local-to-global analysis scheme might have some positive implication to dealing with local minima, but its proof has not yet been established.
4. Segmentation and recognition are tightly coupled. No foreground extraction is necessary, which is achieved by backtracking the response of the network through the hierarchy to the image parts contributing to the recognition.
5. The network is locally and sparsely connected. This is a crucial restriction one must impose for computational tractability with a large network as well as hardware implementation of large networks.
6. The system is able to detect, memorize, and utilize different features for different objects, and thus is quite different from conventional pattern recognition schemes in which the classification decision is based on a single and fixed feature parameter space.

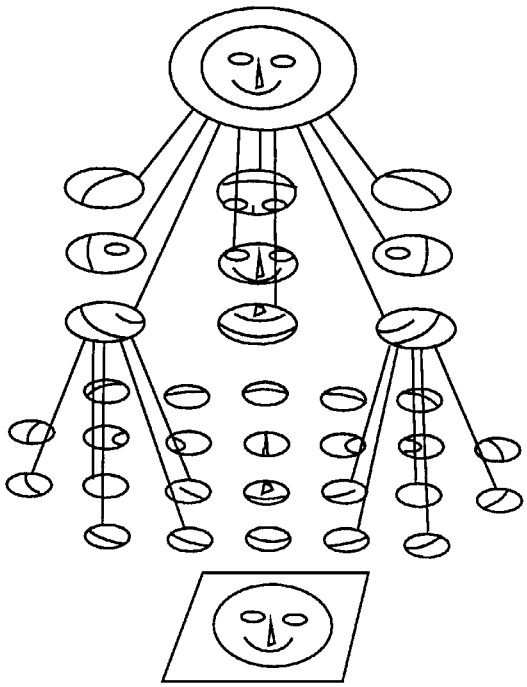

Figure 1. A schematic illustration of hierarchical feature grouping in the Cresceptron. In the figure, not all the connections are shown.

7. Several structures for the network have been developed and evaluated with respect to desired network properties.

The remainder of this paper is organized as follows. Section 2 presents a system overview. Section 3 introduces the stochastic hierarchical distortion model that the Cresceptron is based upon. Section 4 introduces components of the network and presents their properties. These components serve as building blocks of an automatically generated network. Section 5 explains the Cresceptron structure and its working. Section 6 presents some results of experiments. Section 7 gives some concluding remarks.

2. System Overview

The Cresceptron is designed to have two phases: learning and recognition. During the learning phase, a series of images is presented to the Cresceptron which learns the objects specified by a human operator. In the recognition phase, a new image is presented to the Cresceptron which finds recognizable objects in the image and reports the names and then, segments every recognized object from the input image.

2.1. Attention Pyramid

Human eye movements occur in examining a scene so that the object of interest falls on the retina (Iarbus, 1967; Lévy-Schoen, 1981; Treisman, 1983). In the current version of the Cresceptron, a simple version of attention scan is used to examine the input image effectively.

Each attention fixation defines a square attention window. Its objective is to scale the part of image covered by the square attention window down to the size of the *attention image* (i.e., with a fixed number of pixels) as input to the neural network. In our experiment, the attention image is a square of 64×64 pixels.

In order to deal with object of different sizes, the attention window has different sizes. A series of attention window sizes are defined: $W_1, W_2, \ldots, W_k$, where $W_{j+1} = \alpha W_j$. The value of α is determined by the fact that a rate of $1 - \sqrt{\alpha}$ size difference is well within the system tolerance, which is closely related to the system vigilance v to be explained in Section 4.1. (In the experiment, we used an empirical relation $\alpha = \lfloor 5 * v \rfloor / 5$.) If several v values have been used in the learning phase, the largest corresponding α value is used in the recognition phase.

With the Cresceptron, there are two attention modes, manual and automatic. In the manual attention mode, which is mainly designed for the learning phase, the user interactively selects a location and a legal size of the attention window so that the object to be recognized can be directly mapped to the attention image. In the automatic attention mode, which is designed for the reco gnition phase, the system automatically scans the entire image. The scan window size W starts from the maximum legal attention window size, and passes through all the legal window sizes. For each window size W, the attention window scans the entire image from left to right, from top to bottom, by a step size p. (In our experiment $p = W/5$.) After an attention image is obtained, learning or recognition is applied to the attention image.

2.2. Position, Scale, Orientation and Other Variations

In the Cresceptron, attention image recognition—applying the network to the attention image—can tolerate a moderate amount of shape distortion based on the stochastic distortion models to be presented in Section 3. Large variations in object's scale and position are dealt with by visual attention mechanism described above. In other words, attention image recognition can tolerate size and positional differences between two consecutive legal attention window sizes and between two consecutive attention scan positions. It also tolerates other types of variation, such as object orientation. Large orientational variations should be learned individually, as indicated by Fig. 2, which

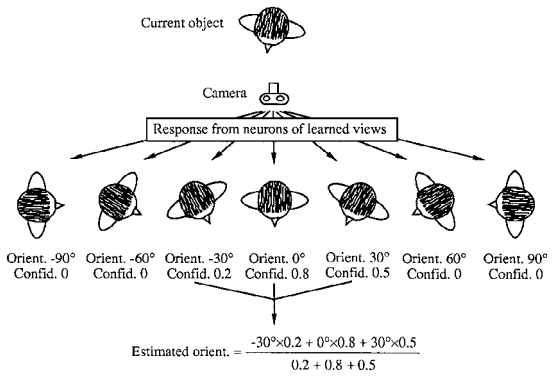

Figure 2. Recognize different orientations by learning several exemplars. Different top views of a human body are drawn in the figure to indicate corresponding learned camera views.

shows how the system estimates an object's orientation of an unknown view based on the orientations of several learned views.

Some studies have demonstrated that the human vision system does not have a perfect invariance in either translation (Nazir, 1990), scale (Kolers et al., 1985), or orientation (Thompson, 1980). These studies seem to suggest that at least human vision does not solely rely on invariant features, even if it extracts them. It appears that a human's ability in recognizing objects with distortion due to size, relative orientation between the object and the viewer, lighting conditions, etc., could be explained by learning under various variations. This not only makes it possible to recognize object while identifying orientation, but also makes the system more efficient by allocating less memory for cases that rarely occur (e.g., up-side-down human faces).

In the Cresceptron, we do not use predefined invariants (see, (Weiss, 1993) for a survey) because existing invariants require that the objects be pre-segmented, have well extracted object contour, belong to a special class, and that feature-model correspondence has been established. These requirements are not suitable for general real-world settings and the methods tend to be *ad hoc* in nature.

2.3. *Image Primitives*

The system is designed in such a way that any image primitive can be easily used for learning and recognition. In order to reduce the system sensitivity to absolute image intensity values, the current version of the Cresceptron uses directional edges as image primitives. From each attention image, the system computes edge images at two scales (Gaussian blurring with templates of 5 Pixels and 9 Pixels, respectively), each of which records the zero-crossings of the second directional derivative of the Gaussian-smoothed image along one of the 8 discretized directions. We add this larger scale Gaussian smoothing because we want to emphasize global edges. Therefore, there are a total of 16 edge images as input to the network, 8 for each scale. In the edge image, a pixel is equal to 1 if a directional edge is present and 0 otherwise. Since the contrast of every attention image is automatically adjusted to the full range of the 8-bit intensity representation, the edge threshold does not need to be adaptive. On the other hand, the method does not require connected edge contours. The pixels in the input edge images receive real numbers in $[0, 255]$ once the edge images are blurred by the tail probability profile.

The use of directional edges, instead of letting the system discover them, was motivated by efficiency considerations. On the other hand, studies show that edge detection occurs very earlier in biological visual passway, e.g., in ganglion cells on the retina, and thus is likely "hard wired" at birth or shortly after birth (Dreher and Sanderson, 1973; Wilson and Giese, 1977; Wilson and Bergen, 1979).

2.4. *What to Learn*

With the Cresceptron, the human operator, as a teacher, indicates to the system what area in the image should be learned. How does segmentation occur in human's visual learning? In the course of children's early visual learning (Carey, 1985; Martinez and Kessner, 1991), touching, handling and movements of objects allow the objects to be visually segmented from the background so that each object is mentally linked with its segmented visual appearances from various views. As reported by neurologist Oliver Sacks (1993), in order to learn to see (i.e., to understand what an image means), the necessity of touching, manipulation and segmentation of object (via, e.g., segmentation of face from background through a face motion) is strikingly evident in a case where an adult who has been blind since childhood suddenly has his vision restored. In the case of Cresceptron, a human teacher provides segmented image areas directly, and due to this, learning with Cresceptron is not fully automatic and of course, very different from human visual learning.

In order for the system to learn many objects, a user-friendly user interface is essential. We have developed a window-based interactive interface shown in Fig. 3. During the learning phase, the user selects the object to learn by interactively drawing a polygon in the attention image to outline the object, as shown in Fig. 4. Then a click on the button "mask" tells the system to remove the background. A click on the button "learn" triggers the learning process which incrementally grow the network according to a framework outlined below.

2.5. *The Network Framework*

The network consists of several levels $\{i; i = 0, 1, 2, \ldots, N\}$ ($N = 6$ in the current experiment). The number

Figure 3. Interface console of the cresceptron.

Figure 4. Specifying the object to learn. (a) User draws a polygon in the attention image to outline the object to learn. (b) A click on the button "mask" removes the background.

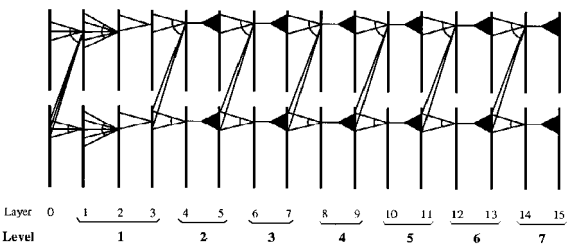

Figure 5. A schematic illustration of the selected framework for a multi-level network. A thick vertical line represents an edge-on view of a neural plane and thus, each neural plane is indicated by a line segment. In the illustration, a plane being connected to two lower-layer planes means that every plane in this layer can be connected to several planes during learning. Otherwise, it accepts input from only one lower-layer plane.

of levels N is to guarantee that the receptive field of each top-level node covers the entire attention image. The receptive field of a node at a layer l is defined as the spatial extent of the layer-0 input pixels it connects to either directly, or indirectly through other intermediate levels.

Each level has 2 or 3 layers. Thus, totally, the network has several layers that are numbered collectively by l, $l = 0, 1, 2, \ldots, L$. The output of a lower layer l is the input for the next higher layer $l + 1$. Figure 5 shows a framework that has been implemented. In the figure, only one set of input connection is shown. Such connections are also used to indicate the type of neural plane to be explained in Section 4. At each layer l, there are many neural planes of the same type. Each neural plane consists of a square of $k(l) \times k(l)$ nodes. That is, all the neural planes in a layer have the same

number of nodes. Each neural plane represents a particular feature and the response at a certain location of the neural plane indicates the presence of the feature. Therefore, all the nodes in a neural plane use the same mapping function that maps from its input nodes to its output. This structure allows us to keep only one set of neurons and connections for the entire neural plane. The nodes at layer $l = 0$ correspond to pixels in the input attention image.

2.6. Growth, Recognition and Segmentation

Initially, the network does not exist: no neural plane or neurons exists at any layer. Given each input image to learn, the learning process invoked by a click on the button "learn" automatically grows the network recursively from the lowest layer to the top layer. The way in which the network grows will be explained in Subsection 5.4. There are two types of learning with the Cresceptron, individual and class. With the former, each input is dis tinguishable from other inputs. With the latter, each input is an example of a class and it is not to be distinguished from other examples of the same class. An individual learning will always add, at the top level, a new neural plane which is assigned with a new label (e.g., name of the object, with attributes if applicable) supplied by the user. A class learning will cause some incremental growth at some levels of the network; but no new neural plane will be added at the top level, since the top-level neural plane of an existing class is used instead. Learning of a new class always starts with individual learning using the first example in the class in order to add a new label.

In the recognition phase, an unknown image is supplied as input. The system automatically applies the aforementioned simple attention mechanism to extract attention images which are then fed into the learned network. If a neural plane at the top level of the network has a high response (i.e., confidence value is higher than 0.5), a report is generated to the user. If the user wants to locate the region in the input that corresponds to the recognized object, he or she clicks the button "segment" which invokes a segmentation process which back-track the network response to the input image.

3. Stochastic Distortion Models and Feature Matching Schemes

Generalization is a key issue in learning. A system for recognition must be able to recognize objects that look like, but not necessarily the same as, a learned exemplar. From the view point of computational tractability, the number of images that the Cresceptron can practically learn is much less than those available on a child's retina during his/her early development. Thus, while human visual learning can afford to continuously learn and generalize from an astronomical number of image instances, our system must generalize from a relatively small number of images. The question is how. This section addresses the issues related to generalization based on stochastic pattern distortion models. We will see the limit of a few matching schemes in the type of stochastic pattern distortion they can incorporate. Particularly, it is shown that a single-level template matching is functionally inferior to multi-level matching. The properties derived in this section are utilized in the design of the Cresceptron framework.

3.1. Feature, Its Shift and Expected Distance Distortion

In recognizing an image, a discrepancy between the location of a feature in the image and that of the corresponding learned image can be considered as distortion. Absence of a feature is treated as a location distortion beyond the allowed range, as far as the distortion-based feature matching is concerned.

First, consider the 1-D case. A 1-D image is a function $f(x)$: $\Re_+ \mapsto \Re$, where $\Re_+ = [0, \infty)$ and $\Re = (-\infty, \infty)$. A *randomly distorted image* $\tilde{f}$ from image f is a new image defined by

$$\tilde{f}(x) = f(x + \alpha(x)) = f(u(x)) \tag{1}$$

where $\alpha(x)$ is a realization of a random process $A = \{\alpha(x); x \in \Re\}$, and

$$u(x) = x + \alpha(x) \tag{2}$$

x is any point in the image. In particular, x is the position of a feature. $u(x)$ is the distorted position of the point at x, and $\alpha(x)$ is the amount of random distortion from x. Now, consider the distribution of α (note that for notational simplicity, we drop the parameter x in $\alpha(x)$ here).

Definition 1. The *tail probability* of a random variable α is defined as

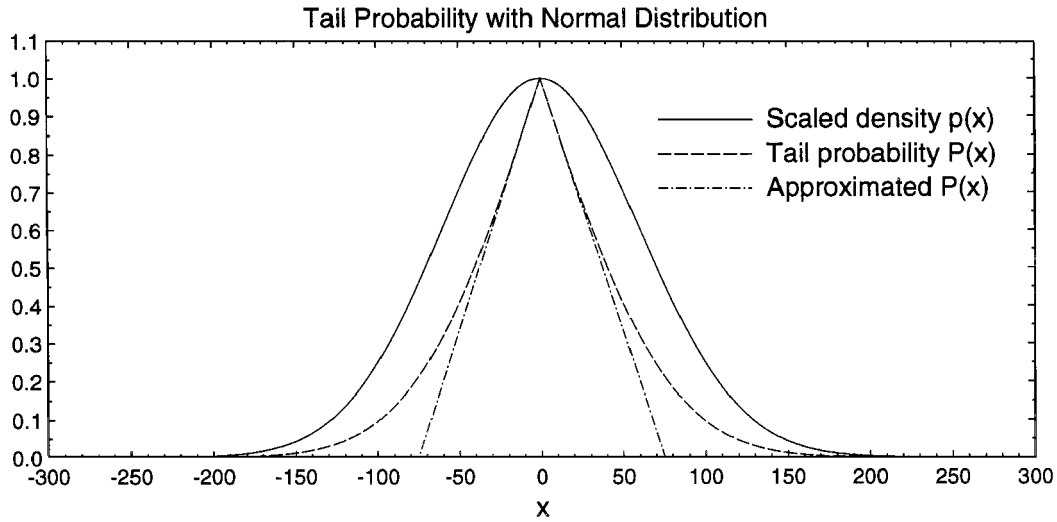

Figure 6. Tail probability of Gaussian distribution and its triangular approximation. The scaled probability density is $p(x) = e^{-x^2/(2\sigma^2)}$. The approximated $P(x)$ is a triangular function $t(x)$ so that $t(x)$ and $P(x)$ have the same slope at $x = 0$.

$$P(s) = P(|\alpha| \geq |s|) = 1 - \int_{-|s|}^{|s|} p(x)\,dx,$$

where $p(x)$ is the probability density function of α.

The tail probability $P(s)$ indicates the probability for the deformation α to have a magnitude not less than that of s. For example, supposing α has a Gaussian distribution $N(0, \sigma^2)$, the tail probability $P(s)$ is shown in Fig. 6. As we can see, the tail probability can be reasonably approximated by a triangular profile, especially in the central part. Thus, for implementational simplicity, one may use a triangular tail probability profile $t(s)$:

$$t(s) = P(|\alpha| \geq |s|) = \begin{cases} \frac{1}{T}(T - |s|) & \text{if } |s| \leq T \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

A simple way to determine the cut-off position T is such that $P(s)$ and $t(s)$ have the same left and right derivatives, respectively, at the center $x = 0$, which yields

$$T = \sqrt{\frac{\pi}{2}}\sigma. \tag{4}$$

As shown in Fig. 6, a side effect of this approximation is a lowered probability measure for large distortions. It is easy to show that the underlying distribution of α that corresponds to the above triangular tail probability function $t(s)$ is a uniform distribution in $[-T, T]$, if the density is symmetrical.

Now, we consider the distance between two features.

Definition 2. Assume that two feature points at x and x', respectively, are distorted to appear at $u(x)$ and $u(x')$, respectively. The distance distortion between these two features

$$r(x, x') = (u(x') - u(x)) - (x' - x) = \alpha(x') - \alpha(x) \tag{5}$$

is called *distance distortion* between x and x'. Its root variance

$$v(x, x') = \sqrt{\text{var}[r(x, x')]}$$

is called their *expected distance deviation* (EDD).

We will use EDD to characterize the amount of pattern deformation when we consider pattern matching schemes.

3.2. *The White Distortion Model and Single-Level Template Matching*

The extent of a feature is the size of the pixel region, in the input image, that defines the feature. In the

Cresceptron, the higher the level, the larger the extent of the feature it detects. The extent of a feature detected by a node cannot be larger than the receptive field of the node, but they are typically close. A visual feature arises from a physical object surface and its appearance changes with the surface. Therefore, we can consider a deformation of a visual features as a result of that of the surface.

Consider inter-feature distance between two neighboring features at x and x' respectively. The extent of features to be detected increases with the level number of the network. The higher the level, the larger the extent becomes. Therefore, inter-feature distance $|x-x'|$ varies with the extent of the feature extent or the level number. The higher the level, the larger inter-feature distance $|x - x'|$ we are interested in. Therefore, we need to investigate how the variation of inter-feature distance $|x - x'|$ changes with the value $|x - x'|$ itself.

Definition 3. If $\alpha(x)$ in (2) is such that $\alpha(x)$ and $\alpha(x')$ are mutually uncorrelated for any $x, x' \in \Re_+$, then the distortion model is called a *white* distortion model.

If, in a white distortion model, $\alpha(x)$ and $\alpha(x')$ have the same distribution for any $x, x' \in \Re_+$, then the distortion model is called *homogeneous*. For example, Let A be a white homogeneous process and $\alpha(x)$ has a Gaussian distribution $N(0, \sigma^2)$, independent of x (homogeneity). Then, the tail probability $P(x, s)$ of $\alpha(x)$ indicates the probability for the distortion at x to have a magnitude larger than s. With a homogeneous model, $P(x, s) = P(x', s)$ for any $x, x' \in R_+$, and thus, we can drop x in $P(x, s)$ to write $P(s)$ instead. Similarly, the EDD of a homogeneous model, $v(x, x')$, is a function of $x' - x$ only and we write $v(d)$ with $d = x' - x$.

With a homogeneous white distortion model, let the variance of $\alpha(x)$ is σ^2, independent of x. Then, from (5), we have

$$v^2(d) = \text{var}[\alpha(x')] + \text{var}[\alpha(x)] = 2\sigma^2 \qquad (6)$$

and thus, $v(d) = \sqrt{2}\sigma$. Thus, we have the following property:

Property 1. *In a white homogeneous distortion model, the EDD is constant, independent of the distance d.*

This is not a desirable property for generalization. If two features are far apart in the input, it is more likely that the distance variation between them is also larger (e.g., consider the distance deviation due to a slight change in the viewing orientation).

Now, consider a single-level template correlation-based matching methods. Such a pixel-to-pixel correlation method does not explicitly take into account the positional deviation and thus, a one-pixel shift of a pixel-wide pulse results in a bad value in the pixel-to-pixel correlation. If either the matching template or the input is blurred by a blurring function $h(x)$, the amount of deformation is characterized by the variance of $h(x)$. However, the Property 1 indicates that the EDD is still a constant, which does not change with the extent of the feature. This is counter-intuitive: a larger template should be allowed to deform more in matching. Therefore, a single-level template correlation-based matching method is not very effective for general distortion-based matching problems.

3.3. *Markovian Distortion Model and Single-Level Deformable Matching*

The following investigation of Markov distortion models provides insight into why we need a hierarchical network.

Still using the distortion definition in (1), a Markov random process A is such that for any x, $s \geq 0$ and $z \in \Re$

$$P\{\alpha(x+s) < z \mid \alpha(x'), x' \leq x\} = P\{\alpha(x+s) < z \mid \alpha(x)\}. \qquad (7)$$

In other words, the amount of future $\alpha(x+s)$, is independent of the past $\{\alpha(x'); x' < x\}$ if the present $\alpha(x)$ is given. A Markov distortion model is defined as in (1) where $A = \{\alpha(x); x \in \Re\}$ is a Markov random process A. If

$$P\{\alpha(x+s) < z \mid \alpha(x)\} = P\{\alpha(s) < z \mid \alpha(0)\} \qquad (8)$$

holds for all z, $x \geq 0$ and $s > 0$, the process A is said to be a *homogeneous* Markov process.

Next, we consider a homogeneous Markov process, since in the absence of *a priori* information about the distortion variation at different locations, it is natural to assume that the statistic nature of distortion distribution does not change from one image location to another.

Let us consider the EDD of a homogeneous Markov distortion model. For any $x, x' \in \Re_+$,

$$v^2(x, x') = E[(r(x, x') - E[r(x, x')])^2]$$

Without loss of generality, assume $x \leq x'$. Using a well-known equation for conditional expectation

$$E[E[y \mid x_1, \ldots, x_n]] = E[y] \tag{9}$$

we have

$$v^2(x, x') = E[E[(r(x, x') - E[r(x, x')])^2 \mid \alpha(x)]]$$

Due to the homogeneity, it follows that

$$\begin{aligned} &E[(r(x, x') - E[r(x, x')])^2 \mid \alpha(x)] \\ &\quad = E[(r(0, x' - x) - E[r(0, x' - x)])^2 \mid \alpha(0)] \end{aligned}$$

Thus,

$$\begin{aligned} v^2(x, x') &= E[E[(r(x' - x) - E[r(x' - x)])^2 \mid \alpha(0)]] \\ &= v^2(0, x' - x) \end{aligned}$$

Therefore, $v(x, x')$ is shift-invariant for homogeneous process and we can write $v(d)$, where $d = x' - x$.

Theorem 1. *The EDD $v(d)$ of a homogeneous Markov random distortion model is proportional to the square root of the distance $\sqrt{d}$:*

$$v(d) = \sqrt{d}\, v(1). \tag{10}$$

The proof is presented in Appendix A.

The above result can be extended to any kth central moment

$$w(x, x') = E[(\alpha(x') - \alpha(x) - E[\alpha(x') - \alpha(x)])^k]$$

where $k > 0$ is any positive integer, of a homogeneous Markov random distortion model. Note, $v(x, x') = \sqrt{w(x, x')}$ when $k = 2$. Since we have $w(x, x') = (-1)^k w(x', x)$, without loss of generality, suppose $x' \geq x$. Due to homogeneity, we have $w(x, x') = w(0, x' - x) = w(0, d)$, where $d = x' - x$.

Corollary 1. *Let $w(x, x') = E[(\alpha(x') - \alpha(x) - E[\alpha(x') - \alpha(x)])^k]$ be the kth central moments of a homogeneous Markov random distortion model, where $k > 0$ is any positive integer. If $w(0, d)$ is continuous in d, then*

$$w(x, x') = (x' - x) w(0, 1). \tag{11}$$

if $x \leq x'$, and

$$w(x, x') = (x - x') w(1, 0). \tag{12}$$

if $x > x'$.

The proof is relegated to Appendix B.

Next, consider the 2-D case. A 2-D image (i.e., pattern) is defined by a two dimensional function $f(x, y)$: $\Re^2 \mapsto \Re$. The image can be considered as a local feature map (e.g., edge map) of an intensity image. A randomly distorted image $\tilde{f}$ from f is a random field defined by

$$\tilde{f}(x, y) = f(x + \alpha(x, y), y + \beta(x, y))$$

where $(\alpha(x, y), \beta(x, y))$ is a realization of a 2-dimensional random field $A = \{(\alpha(x, y), \beta(x, y)); (x, y) \in \Re^2\}$. We may also define a Markov random distortion field in the same way as we define causal, semicausal and noncausal Markov random fields (Jain, 1989). In a conventional random field, the value at (x, y) is random, while in our random distortion model, the distortion is random and two-dimensional.

Let us consider an example of 2-D distorted image. Figure 7(a) shows a binary edge image $f(x, y)$, computed from an intensity image. Suppose that the distribution of the distortion components $\alpha(x, y)$ and $\beta(x, y)$ is determined by

$$\begin{aligned} \alpha(x, y) &= \int_0^x a(s)\, ds + \int_0^y b(s)\, ds \\ \beta(x, y) &= \int_0^x c(s)\, ds + \int_0^y d(s)\, ds \end{aligned} \tag{13}$$

where $a(s), b(s), c(s)$ and $d(s)$ are distinct realizations of white Gaussian random processes of the form $W = \{w(t); t \in \Re\}$. In other words, W is such that for any x and $s > 0$, $\int_x^{x+s} w(t)\, dt$ is a random variable of Gaussian distribution $N(0, s\sigma^2)$. For integer x and y, we can express the random distortion in two sums of identically and independently distributed random variables. Figure 7 shows several examples of such distorted images.

The above results about 1-D Markov distortion models can be applied to each of the two directions x, y in a 2-D model. However, the result may not be directly applicable to any other directions. The distance d between two points (x_1, y_1) and (x_2, y_2) may not necessarily be directly extended to Euclidian distance either. For example, in the above 2-D example, the expected square Euclidian distance $E[(x_1 - x_2)^2 + (y_1 - y_2)^2]$ between any two points (x_1, y_1) and (x_2, y_2) is proportional to the square root of the city-block distance $|x_1 - x_2| + |y_1 - y_2|$ between the two points.

Recalling Section 3.2, the EDD in a white homogeneous distortion model is constant, independent of the

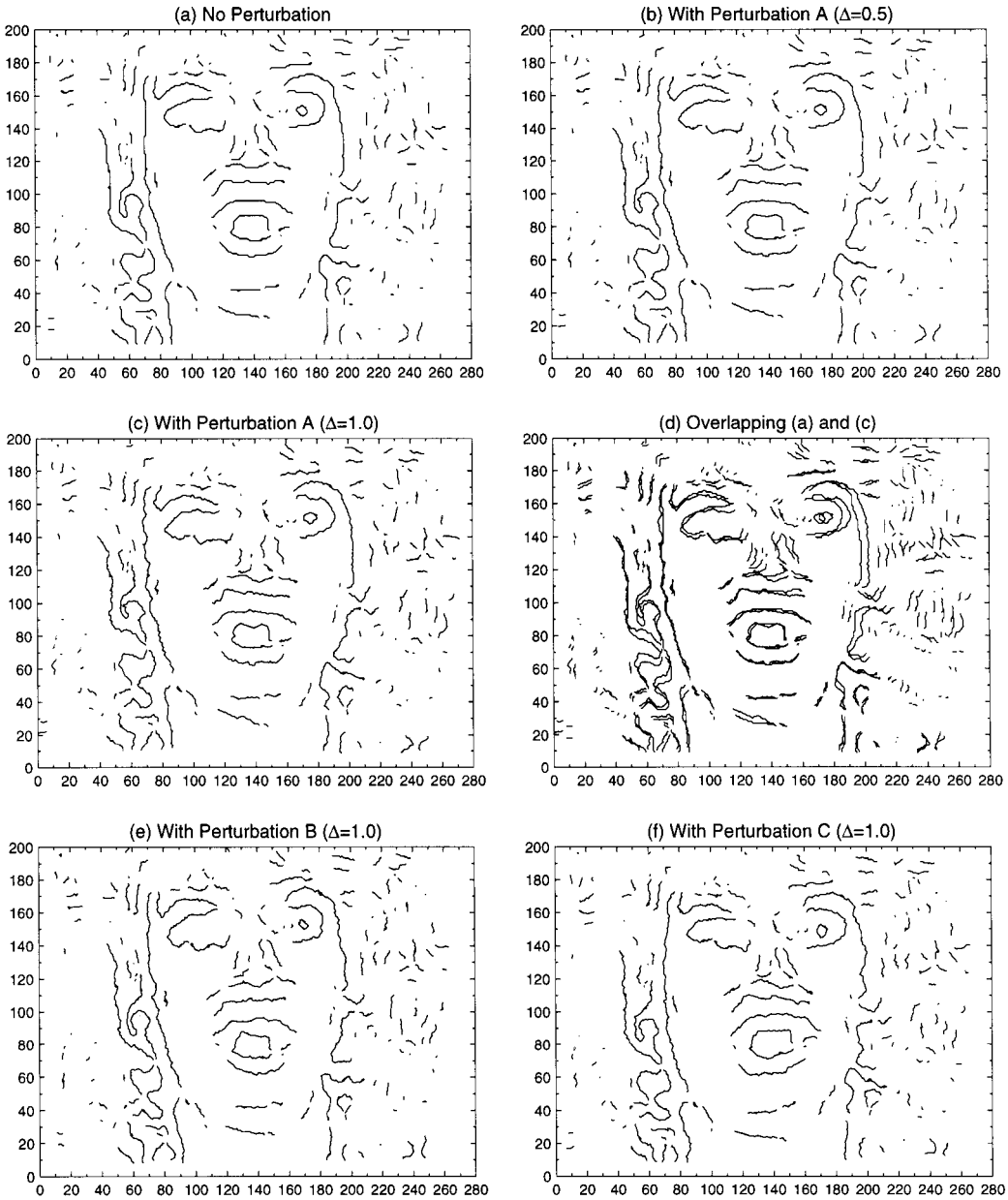

Figure 7. Examples of 2-D homogeneous Markov distortions. (a): Original image with distortion. (b): A distorted image. (c): Same distortion as in (b) except that the magnitude of distortion is doubled. (d): Superimposition of (a) and (b) to show the distortion of (b) from (a). (e) and (f): Two differently distorted images. $\Delta^2 = 12\sigma^2$.

distance between the two feature points. This white distortion model corresponds to a single-level correlation-type of template matching coupled with some blurring. This type of template matching method has a very limited generalization power because the allowed distortion is constant.

Markov distortion model cannot be implemented by matching with a fixed template, because, as indicated by Theorem 1, the amount of distortion in a Markov distortion model is not constant, but instead, it depends on the distance between two features that are under consideration.

In fact, a deformable template can be used to match a Markov distortion model. We consider the 1-D case. Suppose that a 1-D template consists of $n+1$ features: $f(x_i)$ at location x_i, $i = 0, 1, \ldots, n$, where $x_0 < x_1 < \cdots < x_n$. Suppose that in the input we have observed $n+1$ features, $f(x_i)$ at distorted location $\tilde{x}_i = x_i + \alpha_i$, $i = 0, 1, 2, \ldots, n$. If the input feature locations are distorted with a homogeneous Markov random distortion model, then the inter-feature distortion $\delta_i = \alpha_i - \alpha_{i-1}$, given α_{i-1}, is mutually independent.

The Markov model based deformable template matching can be conducted as follows. Identify the first feature $f(x_0)$. Suppose that the $(i-1)$th feature $f(x_{i-1})$ has been identified. Thus, the observed distortion at this feature is observed as

$$\alpha_{i-1} = \tilde{x}_{i-1} - x_{i-1}$$

Then, given the observed distortion α_{i-1}, "stretch" and "compress" (i.e., deform) the template between two feature points x_{i-1} and x_i to identify the next feature $f(x_i)$ and its location $\tilde{x}_i$ in the input, according to the conditionally independent distribution of δ_i, at location

$$\tilde{x}_i = x_i + \alpha_i = x_i + \alpha_{i-1} + \delta_i$$

Suppose the feature is observed at position $\tilde{x}_i = s_i$, the confidence related to this position is indicated by the conditional tail probability of $\delta_i = \tilde{x}_i - x_i - \alpha_{i-1}$:

$$\begin{aligned} &P(\text{abs}(\delta_i) > \text{abs}(s_i - x_i - \alpha_{i-1}) \mid \alpha_{i-1}) \\ &\quad = P(\text{abs}(\alpha_i - \alpha_{i-1}) > \text{abs}(s_i - x_i - \alpha_{i-1}) \mid \alpha_{i-1}) \end{aligned} \tag{14}$$

where $\text{abs}(x)$ is the absolute value of x. Note that in a homogeneous Markov system, the probability in (14) is conditionally independent of j, for all $j < i-1$. Once we have the first feature at $\tilde{x}_0$, recursively using the Markovianity, the overall matching confidence can be determined by the overall tail probability for observing $\tilde{x}_i = s_i$, $i = 1, 2, \ldots, n$, as follows

$$\prod_{i=1}^{n} P(\text{abs}(\alpha_i - \alpha_{i-1}) > \text{abs}(s_i - x_i - \alpha_{i-1}) \mid \alpha_{i-1})$$

As we proved in Theorem 1, the expected amount of template *shift* at x_i is proportional to square root of the distance between x_0 and x_i: $\sqrt{x_i - x_0}$. As can be seen, a single-level matching using a fixed template cannot even deal with this type of Markov random distortion, because the amount of shift allowed is constant with a fixed template.

3.4. *Non-Markovian Distortion Model and Multi-Level Matching*

Single-level deformable template matching relies on the Markovianity. Once a feature is located, the location of the next feature is independent of all previous feature locations. Thus, the distance deformation between the current feature and the next feature can be checked by a certain statistic measure, such as variance.

However, if the distortion process is not Markovian, a single-level neighboring feature distance checking is not sufficient, because the probability depends on near and far-away nodes (i.e., at different scales). In a multi-level distortion checking, each level can be responsible to a certain scale. The scale here implies both extent of features and inter-feature distance, because distance between two larger features is also larger.

In order to deal with distortions that are not necessarily proportional to interfeature distance, the Cresceptron has adopted a multi-level, multi-scale matching scheme, as follows. There are L levels of template matching, $l = 1, 2, \ldots, L$. The receptive field of level-l node is a square of $(2^l + 1) \times (2^l + 1)$ pixels. An odd size is used so that the receptive field is centered at a pixel. The response of the mth neural plane at a layer l at position (x, y) is called confidence value $n(l, m, x, y)$, which ranges between 0 and 1. $n(l, m, x, y) \approx 1$ implies that the feature represented by neural plane m is centered at (x, y) with a high confidence. $n(l, m, x, y) \approx 0$ means either there is no feature at (x, y) or the feature at (x, y) is very different from the feature that the mth neural plan represents.

Since the receptive field of level-l node is a square of $(2^l + 1) \times (2^l + 1)$, features at a higher level have a larger receptive field. Thus, to determine the inter-feature distance between two features of a size $(2^l + 1) \times (2^l + 1)$, we jointly examine positions that are roughly $(2^l + 1)$ apart, i.e., a subsampled grid of nodes $\{(x_0+ir, y_0+jr); r = 2^l, i, j$ are any integers$\}$. r is the subsample spacing (one sample every r nodes). In the learning phase, the examination results in a memory of the response at these grid points. In the recognition phase, the task is to detect the presence of the recorded pattern on the subsampled grid while allowing distortion.

Two neighboring features on the grid at level l are 2^l-pixels apart. If the EDD of these two pixels is proportional to $\sqrt{2^l}$, it coincides with the property of the Markov model, as stated in Theorem 1. Otherwise, it implies the model is Non-Markovian. The larger the EDD, the more distortion is allowed at this level. Since our framework allows arbitrary specification of EDD at every level, and thus, is not restricted by Markovianity. We will get back to this issue in Section 4.

The result we derived so far is summarized in Table 1.

In summary, this section has analyzed why Cresceptron uses a multi-level stochastic distortion model instead of a single-level one used by, e.g., a single-level template correlation method. The multi-level stochastic distortion model used by the Cresceptron can handle more general distortions than those that a single-level scheme can deal with.

Table 1. Stochastic distortion models and the corresponding matching schemes.

Distortion model	EED	Matching schemes
White	Constant	Single-level template
Markovian	Proportional to distance	Single-level deformable template
Non-Markovian	Any	Multi-level multi-scale

4. Network Components

The network components to be presented here are used to implement, in a digital form, a Non-Markovian stochastic distortion model as discussed in Section 3. They are three types of neural layer: pattern-detection layer, node-reduction layer, and blurring layer. A module is a combination of layers. The modules are used to construct a complete framework. For simplicity, we will first consider 1-D networks which is then extended to 2-D networks in a straightforward way.

4.1. Pattern-Detection Layer

As explained in Section 3, pattern matching with deformation consists of two subtasks, detection of feature and checking the interfeature distance. The purpose of the pattern-detection layer is to accomplish the former. Two types of pattern-detection layer are useful: regular pattern-detection layer and subsampled pattern-detection layer.

The regular pattern-detection layer is illustrated in Fig. 8. For a regular pattern-detection layer at layer l, there are a number of input neural planes at layer $l - 1$. Let $n(l, m, i, j)$ denote the value of response at position (i, j) in the mth neural plane at layer l. A feature at position (i_0, j_0) is a 2-D pattern $\{n(l, m, i_0 + i, j_0 + j); -h \le i, j \le h\}$, where $2h + 1$ is defined as the size of the pattern.

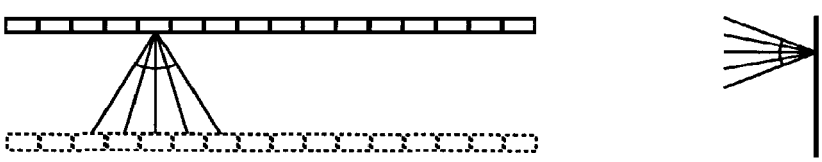

Figure 8. Regular pattern-detection layer. (a) A schematic illustration in which only the connections to one node are drawn and only one input plane is shown. The arc across the connections represents an AND-like condition. (b) The symbol of the regular pattern-detection layer used in Fig. 5. The number of connections in the symbol indicates the size value $2h + 1$. But, the case of 2 connections is reserved for the sybol for the subsampled pattern-detection layer.

In the learning phase, once a new feature is detected at (i_0, j_0) at layer l, a new neural plane k is created at layer $l+1$, devoted to this feature. The new feature is memorized by a new node whose synapses are assigned with the observed values

$$w(l, k, m, i, j) = n(l, m, i_0 + i, j_0 + j),$$

$-h \le i, j \le h$. Let P denote the set of all the indices of input planes where the new feature is detected. In the recognition phase, the response in the kth new neural plane at (i_0, j_0) of layer $l+1$ is

$$n(l+1, k, i_0, j_0) = f[s(l+1, k)z(l, k, i_0, j_0) - T(l+1, k)]$$

where

$$z(l, k, i_0, j_0) = \sum_{m \in P} \sum_{-h \le i, j \le h} w(l, k, m, i, j) n(l, m, i_0 + i, j_0 + j)$$

and $f(x)$ is a (monotonic) sigmoidal function (or a soft limiter) (Lippmann, 1987) that maps x to a normalized range [0, 1], and the values $s(l+1, k)$ and $T(l+1, k)$ are automatically determined in the learning phase so that

$$f[v\, s(l+1, k)z(l, k, i_0, j_0) - T(l+1, k)] \approx 1 \qquad (15)$$

and

$$f\left[\frac{v}{2} s(l+1, k)z(l, k, i_0, j_0) - T(l+1, k)\right] \approx 0 \qquad (16)$$

where v is the only user-specified parameter, called *system vigilance* parameter, which indicates the desired discrimination power of the system. Intuitively speaking, the output is over-saturated if the exact pattern is presented and is under-saturated if about a half of the response is presented, depending, of course, on the system vigilance. Therefore, the pattern detection layer can be viewed as a cross-correlation with the learned pattern followed by a sigmoidal nonlinear mapping on to a normalized range. Such a simple computation can be implemented by the simple processing elements in a digital or analogue neurocomputer.

Another type, the subsampled pattern-detection layer, is similar to the regular one except that the input nodes are not consecutive. Instead, the input nodes are from the subsampled grid as discussed in Section 3.4.

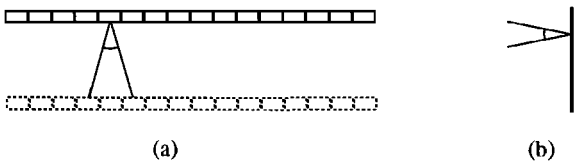

Figure 9. Subsampled pattern-detection layer. (a) A schematic illustration in which only the connections to one node are drawn and only one input plane is shown. The arc across the connections represents an AND-like condition. (b) The symbol of the subsampled pattern-detection layer. In this paper, if the number of connections in the symbol is more than 2, the symbol represents a regular pattern-detection layer.

We will use a type of subsampled pattern-detection layer in which each node accepts four subsamples from the lower-layer neural plane and the subsample spacing r is such that the receptive fields of these four subsamples correspond to the four quadrants of a large square, respectively. Thus, the subsampled pattern-detection layer can be used to increase the receptive field with a minimal overlap in the receptive filed.

4.2. *Node-Reduction Layer*

If the number of nodes in each neural plane is reduced from layer l to $l+1$, then we say that $l+1$ is a *node-reduction layer*, as shown in Fig. 10. Node-Reduction layer is primarily for reducing the spatial resolution of the neural plane, and thus reducing the computational complexity.

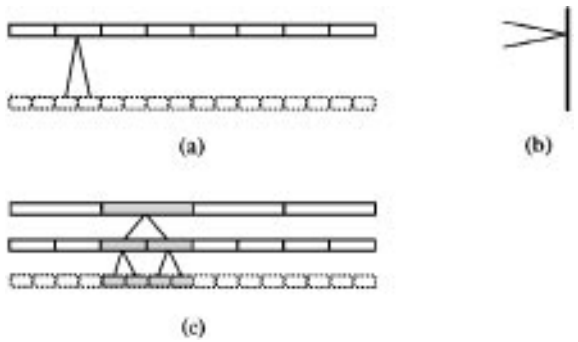

Figure 10. Node reduction layer. (a) A schematic illustration in which only the connections to one node are shown. Every neural plane at the node reduction layer has only one input plane. No arc across the connections is present, which represents an OR-like condition. Notice that the number of nodes is reduced at the upper layer. (b) The symbol of the node reduction layer. (c) A large connectivity (or receptive field) is achieved by the node reduction layer in which each node connects to just local nodes.

Node-reduction implies that one node at layer $l+1$ corresponds to more than one node at layer l. We use a node-reduction rate of two, i.e., one node at layer $l+1$ corresponds to, at layer l, two nodes in the 1-D case and 2×2 nodes in the 2-D case.

Node-reduction may cause difficulties in multi-level pattern matching. We first define recallability with respect to translation:

Definition 4. If a network learns, in the learning phase, a pattern that is presented at a certain location in the input, and it can also recognize, in the later recognition phase, the same pattern but translated arbitrarily in the input image, then, this network is *recallable under translation*. A subnetwork, whose output is to be used as input to another subnetwork, is recallable under translation if the learned response is still present in the output neural plane (but translated) no matter how the input is translated in the input. Here the term "present" means that the corresponding node has a response not lower than what is learned.

At a node reduction layer, the response of a node is a function of the corresponding output from the lower-layer nodes, say, two nodes n_1 and n_2. Then, the response is $f(n_1, n_2)$. A desirable function for recallability under translation is function max: $f(n_1, n_2) = \max(n_1, n_2)$. This means that the output is active (i.e., response is high) if the pattern is presented at either n_1 or n_2. In fact, the function max corresponds to a logic OR function in multivalue logic. In the 2-D case, a node-reduction neural plane k that accepts the input from neural plane m at layer l is determined by:

$$n(l+1, k, i, j) = \max\{n(l, m, 2i+p, 2j+q);\ p, q = 0, 1\}. \quad (17)$$

However, an OR-based node reduction alone does not guarantee recallability under translation. Consider the example shown in Fig. 11. In the learning phase,

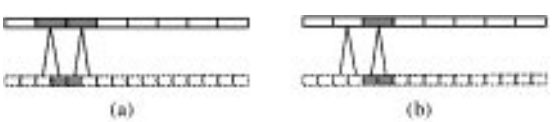

Figure 11. Node reduction causes a reduction of active nodes at the upper layer. (a) During learning, two nodes are active at the upper layer. (b) During recognition, a shift of the pattern causes only one node being active at the upper layer.

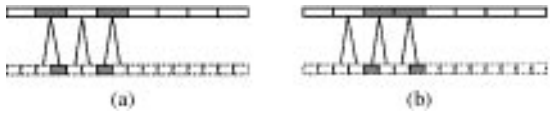

Figure 12. Node reduction causes a distortion of response pattern at the upper layer. (a) During learning, two active nodes are separate at the upper layer. (b) During recognition, a shift in the input causes two active nodes to change their distance at the upper layer.

two neighboring nodes contribute to different upper-layer nodes and thus both upper-layer nodes are active. But a shift in the same pattern in the recognition phase causes two nodes to contribute to a single upper-layer node. Thus, at the higher layer, no successful matching will be reported because only one node is active instead of the expected two.

Such an inconsistent shift at the upper layer can also result in a distorted output pattern, as illustrated in Fig. 12. Due to a shift at the lower layer, some nodes at the upper layer shift but some do not, causing a distortion of the response pattern at the upper layer. Such a distortion can be too large to ignore because a one-node distortion at a high layer corresponds to a very large distortion in the input image. Therefore, we need the following combinations of different layers to eliminate undesirable effects.

4.3. *Node-Reduction Structures Recallable under Translation*

We consider a combination of a pattern-detection layer and a node-reduction layer, the latter being on top of the former. For the node-reduction layer, the key to recallability under translation is to perform different computations for the learning phase and the recognition phase. We introduce two types of structure: (a) feature-centered feature detection; (b) grid-centered feature detection.

The term "feature-centered" means that, in the learning phase, detection of a new feature is performed for all the positions (i, j). Once a new feature is reported at (i_0, j_0), we keep a flag of offset $(o_i, o_j) = (i_0 \bmod 2, j_0 \bmod 2)$ in the new neural plane that is created for this new feature. Then, during the learning phase, we update the response at this new neural plane by computing only the response at $(i, j) = (o_i + 2k, o_j + 2m)$ for all integers k and m so that (i, j) is in the new neural plane. Every uncomputed position (i, j) is assigned a zero value. In other words, we only

compute a response at those positions that are offset from (i_0, j_0) by an even number of coordinates. Note that the new neural plane at layer $l+1$ may be connected to input from several neural planes at layer l, and thus the offset (o_i, o_j) is shared by all these input neural planes.

The term "grid-centered" means that the offset (o_i, o_j) is predetermined for the layer, where $o_i = o_j$ can be either 0 or 1. Therefore, we only detect new features at $(i, j) = (o_i + 2k, o_j + 2m)$ for integers k and m so that (i, j) is in the neural plane, and also only compute the resp onse at these positions in the learning phase. Because a new feature has a significant extent in the lower pattern-detection layer (e.g., 5×5), the grid-centered feature detection does not alter the feature very much compared to feature-centered counterpart.

In the recognition phase, the response must be computed for all positions, regardless of whether it is feature-centered or grid-centered.

Definition 5. A *feature-centered node-reduction module* (FCNR module) consists of two layers: the lower layer is a regular pattern-detection layer and the upper layer is a node-reduction layer. A *grid-centered node reduction module* (GCNR module) is the same as the FCNR module except that the lower layer performs the grid-centered feature detection.

Property 2. *The FCNR module is recallable under translation.*

Proof: We consider a 1-D network case. The 2-D case is analogous. Letting l be the lower layer and then the upper layer is $l+1$. There are only two cases (a) $o_i = 0$, (b) $o_i = 1$. Suppose (a) is true. Then, in the learning phase, only the even positions in the output of layer l can be active. Thus, if a node at the node-reduction layer $l+1$ is active, it must have gotten a response from an even position at layer l. In the recognition phase, a shift of the input at layer $l-1$ to the right by $2k$ or $2k+1$ (k is an integer) nodes will cause the learned pattern at layer l just to shift in the same direction by k nodes. See Fig. 13. Then according to (17), a shift of $2k$ or $2k+1$ nodes in layer $l-1$ makes the learned response at layer $l+1$ just shift in the same direction by k nodes. The similar case is true for a shift to the left. Obviously, a simple shift at layer $l+1$ means that the response value does not change except for the position change. Case (b) can be proved in the same way. Therefore the FCNR module is recallable under translation. □

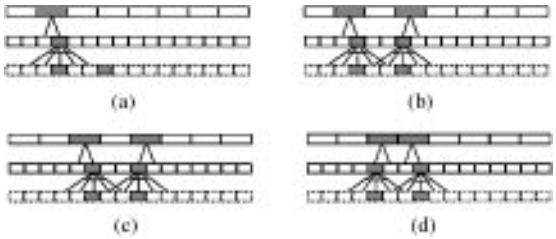

Figure 13. Feature-centered node-reduction (FCNR) module is recallable under translation. Let $o_i = 0$ and the index i start from 1. (a) In the learning phase. Only the first feature is assigned to this upper-layer neural plane because the feature is centered at an even $i = 4$. The other feature at $i = 7$ is assigned to another neural plane. (b) In the recognition phase: the input (at the bottom layer in the figure) does not shift. (c) In the recognition phase: the input shifts by 2. (d) In the recognition phase: the input shifts by 1.

From Fig. 13 we can see that the learning phase distinguishes the two features in terms of their offset, and each neural plane only learns the features with one type of offset.

Property 3. *The GCNR module is recallable under translation.*

The proof is analogous to the proof of Property 2 and is omitted.

As shown in Fig. 12, the output layer of the entire module has more active nodes than that in the learning phase, and the output still varies according to the position of the input. This is a side effect caused by a limited resolution, although recallability is guaranteed. This type of module is only used at a low level where reduction of neural plane size is needed and a pixel level shift of pattern are taken care of by multi-level blurring at high levels implemented by the blurring layers explained in the next subsection.

4.4. The Blurring Layer

The blurring layer is to generate the tail probability profile as shown in Fig. 6. From image processing operation point of view, it blurs the response from the input, but it does not reduce the number of nodes.

Suppose layer $l+1$ is a blurring layer. Let $n(l, i_0, j_0)$ denote the response at position (i_0, j_0) in a neural plane at input layer l. Then the output of layer

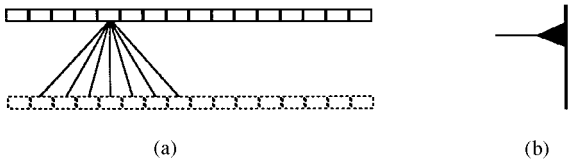

Figure 14. Blurring layer. (a) A schematic illustration in which only the connections to one node are shown. Every neural plane at the blurring layer has only one input plane. No arc across the connections is present, which represents an OR-like condition. (b) The symbol of the blurring layer. The black triangle represents the contribution from a single input node.

$l+1$ at position (i_0, j_0) is defined by

$$n(l+1, i_0, j_0) = \max_{r \le R} \left\{ \frac{R-r}{R} n(l, i_0+i, j_0+j) \;\middle|\; r^2 = i^2 + j^2 \right\} \tag{18}$$

as illustrated in Fig. 14, where R is the radius of blurring, equal to that of the receptive field. As can be seen from Eq. (18), the response at position (i_0, j_0) is the maximum among the input nodes around (i_0, j_0) weighted by a triangular profile. Therefore, an active input will contribute a triangle-type of profile to the neighboring receiving nodes, with the peak of the triangle centered at the position (i_0, j_0).

The mechanism of detecting patterns and tolerating deformation of the patterns in the blurring layer is illustrated in Fig. 15. Suppose a feature point is represented by a delta function centered at position x_0, $\delta(x_0)$, where $\delta(x)$ is the Dirac delta function, $\delta(x) = 0$ for $x \neq 0$ and $\lim_{\epsilon \to 0} \int_{-\epsilon}^{\epsilon} \delta(x)\, dx = 1$. With positional distortion, its position becomes $u(x_0) = x_0 + \alpha(x_0)$, where $\alpha(x_0)$ has a probability density $p(s)$ and a tail probability $P(s)$, independent of x_0 (homogeneity). Then, $u(x_0)$ has a probability density $p(x - x_0)$ and the value of the shifted tail probability function $P(x - x_0)$ indicates the probability for $u(x_0)$ to appear at least $|x - x_0|$ distance away from x_0. This explains the approximated probability profiles in Fig. 15, where each input triangular profile for a feature at $x = x_0$ is a shifted tail probability function $P(x - x_0)$. The extent of blurring depends on the level number.

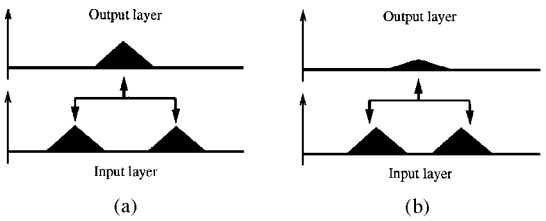

Figure 15. The mechanism of detection and measurement of geometric configuration of features from input layers. In an input layer, the position of a feature is represented by a peak. The blurring of the peak, interpreted mathematically by the tail probability, enables the output layer to measure the positional accuracy. (a) If the positions are exactly correct, two peaks are sensed and thus, the response is high at the output layer. (b) If the positions are displaced, the slope of the tail probability profile are sensed and thus, the output response is relatively low.

Definition 6. A *node-conserved blurring module* (NCB module) consists of two layers: the lower layer is a subsampled pattern-detection layer and the upper layer is a blurring layer.

The blurring does not reduce the number of nodes in the neural plane. Thus, if a pattern is shifted, the response of the blurring layer is just shifted accordingly. Therefore, we have the following property:

Property 4. *The NCB module is recallable under translation.*

5. The Hierarchical Network Framework

The component layers discussed in the preceding subsections can be used to design the framework of a hierarchical network, although the actual configuration and connections will not be determined until the learning phase is actually performed. This design is not concerned with detailed rules about vision, but rather a structure based on which the network learns and grows.

5.1. Realization

We have now described three types of modules, FCNR, GCNR and NCB. The major advantage of the FCNR and GCNR modules is the space efficiency due to their node reduction. The maximization operation in the node reduction allows a subpattern that is matched in the lower pattern-detection layer to shift by one node without affecting the recognition outcome. The direction of this allowed shift depends on whether the feature is detected at an odd or even position (row and column numbers). Although this is highly random for any feature before learning, this oddness is fixed once

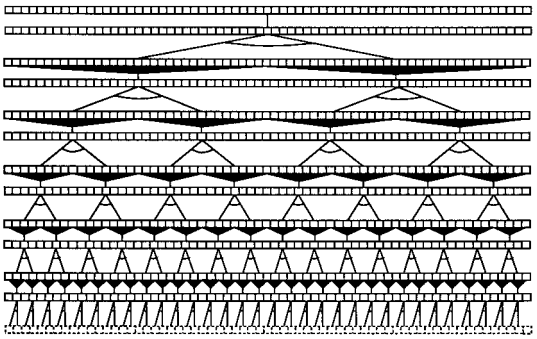

Figure 16. A 1-D illustration of a hierarchical network which consists of the NCB modules. Note how the subsample spacing r and the amount of blurring change from a low layer to high layers.

the feature is learned. This fixed direction of allowed shift does not cause much harm at a low layer. However, it is not desirable at high layers where a shift by one node corresponds to a large distance in the input image. It will overly tolerate large deviations in feature position if the multi-level network consists exclusively of the FCNR and GCNR modules. Therefore, except for the applications where a large tolerance is appropriate such as recognition of simple patterns with a small number of classes, we should use the FCNR and GCNR modules only at low layers.

The NCB module does not reduce the number of nodes, and the amount of blurring can be controlled easily. It is also possible to learn the profile of tail probability by accumulating the population distribution of the learned features. The amount of blurring should be related to the receptive field of the node, as shown by the example in Fig. 16. The principle of blurring based on tail probability is also applied to 16 edge maps (with $T = 2$) that are used as input to the network.

Based on the above observations, we designed a framework for Cresceptron, as illustrated in Fig. 5. The first level consists of three layers. The first layer is a regular pattern-detection layer followed by a GCNR module which has two layers. Next, the structure above layer 3 is similar to what is shown in Fig. 16: 6 levels of the NCB modules. As we noted, the number of levels should be such that the receptive field of the top-level node covers the entire attention image. Since $2^6 = 64$, 6 levels are enough to satisfy this requirement. An additional top level is used for high-level operation such as incorporating samples in "class" learning and future development of inter-class excitation and inhibition in high-level learning (e.g., learning from mistakes). In total, the framework has $3 + 2 \times 6 = 15$ layers.

How is the distortion model related to such a framework? Level 1 acts as a feature detector with some tolerance in feature detection. The following levels are all NCB modules whose blurring level generates a tail probability measure within the corresponding level. The sigmoidal function at each layer acts as an intermediate decision maker which maps the measurements related to the tail probability to a normalized range. The under-saturation and over-saturation points of each sigmoidal function re-normalizes the confidence measure so that a very low confidence is not further transmitted and a very high confidence is considered as an actually occurred event. Due to such an intermediate decision making using sigmoidal functions, the top-level response from the Cresceptron is not exactly a tail probability. However, it can be considered as a confidence measure.

5.2. *Non-Markovianity of the Cresceptron*

Property 5. *The variance of the multi-level blurring function in the Cresceptron implies a statistical distortion model that is not limited by a homogeneous Markov distortion model.*

Proof: Recall that the tail probability was defined as $P(x, s) = P(|\alpha(x)| > |s|)$, and the one used in the Cresceptron is in (3). Suppose that the Cresceptron is limited by (i.e., satisfies) a homogeneous Markov distortion model. Thus, the EDD between two features at $u(x) = x + \alpha(x)$ and $u(x') = x' + \alpha(x')$ of d apart, $d = x' - x$, is

$$\begin{aligned} v^2(d) &= \mathrm{var}[\alpha(x') - \alpha(x)] \\ &= E[\mathrm{var}[\alpha(x') - \alpha(x) \mid \alpha(x)]] \end{aligned} \tag{19}$$

We know that with the triangular tail probability, the distribution of $\alpha(x') - \alpha(x)$, given $\alpha(x)$, is uniform in $[-T, T]$. whose variance is $(2T)^2/12 = T^2/3$. Thus,

$$\mathrm{var}[\alpha(x') - \alpha(x) \mid \alpha(x)] = T^2/3$$

Thus, (19) becomes

$$\begin{aligned} v^2(d) &= E[\mathrm{var}[\alpha(x') - \alpha(x) \mid \alpha(x)]] \\ &= E[T^2/3] = T^2/3 \end{aligned}$$

In Fig. 16 we can see that the distance between two feature points is $d = 2T$. Substituting $T = d/2$ into the above equation yields $v^2(d) = d^2/12$ or $v(d) = d/\sqrt{12}$, which is in conflict with Theorem 1 which concludes that $v(d)$ is proportional to $\sqrt{d}$. This completes the proof.

If we consider the triangular tail probability as an approximation for that of Gaussian density. Then, $v^2(d) = \sigma^2$. From (4), $\sigma^2 = 2T^2/\pi = d^2/(2\pi)$, or $v(d) = d/\sqrt{2\pi}$, also contrary to Theorem 1. □

As we can see, the key point used in the proof is that the EDD between two feature points can be arbitrarily chosen in the Cresceptron over all the levels and scales. This capability is made possible by a hierarchical network.

A couple of points are in order here. The first point is on homogeneity. We used homogeneity in Theorem 1 primarily because of the actual application in the Cresceptron and the mathematical simplicity. If the Markov distortion model is not homogeneous, $v^2(d)$ is an integration of infinitely many infinitesimal conditional deviations in $\alpha(x)$ along the x-axis, a consequence of Markovianity.

The second point is about the selection of T in the Cresceptron. If T is selected as proportional to the square root of d, the distance between the grid points at each level, then, this special case will result in a $v(d)$ that is proportional to the square root of d. But, even in this special case, the Cresceptron still does not necessarily follow a Markov distortion model, because the receptive field of a node at levels other than the first cover more than immediate neighbors. At a high-level, a node at position x depends on a large area of receptive field centered at x in the input plane. That is why a hierarchical network with many levels can deal with statistical distortions that are more complex than Markov ones. We can also see that in a network with a small number of levels, a node at x may only be connected to a small number of neighbors around x. Thus, the corresponding distortion model is restricted by a higher order Markov random field. A hierarchical network in which the receptive field of nodes varies in a full range—from a few pixels at the lowest level to the entire attention image at the highest level—enables the measurement of statistic distortion based on not only a small neighborhood of input, but also a large context information in the input. Thus, its distortion-based generalization is not limited by Markovianity.

5.3. Learning: Detection of New Concepts

The detection of new features is performed at layer 1, the pattern detection layer, and the subsampled pattern-detection layers in all the NCB modules, i.e., layers 4, 6, 8, 10, 12, 14 (see Fig. 5). The only pattern-detection layer that does not need explicit new-feature detection is layer 2 where the objective is to perform an interlayer 5×5 pattern-detection before the following node-reduction layer. Without layer 2, the following node-reduction via maximization will cause excessive tolerance due to the node-reduction layers. In other words, addition of layer 2 allows only a matched 5×5 pattern to shift a node, but does not allow each single node response in layer 1 to shift a node individually relative to its neighboring active node, because the latter will cause, e.g., a diagonal line being recognized as a horizontal line.

An active pattern is significant if the values of the pattern is high. Suppose that the response at position (i, j) in the neural plane m of layer l is denoted by $n(l, m, i, j)$. The feature (i.e., pattern) at (i_0, j_0) in the input edge image is significant if

$$\sum_{-h \le i \le h} \sum_{-h \le j \le h} |n(l, m, i_0 + i, j_0 + j)| \qquad (20)$$

is higher than a predetermined value $s = 3$, such that 3 out of 9 pixels are active so that any line segment through the 3×3 window can be detected. Note that we used $h = 1$ at layer 0 to form a 3×3 window.

A new feature at (i_0, j_0) consists of all the significant patterns centered at location (i_0, j_0) in all the neural planes of the layer. In the subsampled pattern-detection layer, Eq. (20) should be modified accordingly to reflect the fact that each neural plane has four subsample nodes instead of a window of 3×3 consecutive nodes.

When a significant feature appears at a position (i_0, j_0) in the neural planes of layer $l - 1$, there exist three cases according to the response of the neurons at (i_0, j_0) in the neural planes of the next layer l: (1) None of the neurons at (i_0, j_0) is active. Thus, the feature is new. (2) One or more neurons at (i_0, j_0) of layer l are active. But none of them is connected to all the significant patterns at (i_0, j_0) of layer $l - 1$. In other words, although a node or more have responded to the current pattern, but each only covers (or represents) a subset of the active pattern currently present at layer $l - 1$. This implies that the input feature is more complex than the features represented by the existing active neurons, and therefore, it is also a new feature.

(3) Otherwise, the feature is not new. The above conditions are determined by a procedure that examines the response at level l. Once a new feature is detected, the growth of the network occurs as explained in the following.

5.4. *Learning: Growth*

If the structure of a system (not just the value of parameters) adaptively changes according to the input training data, such a system is called self-organizing system (Kohonen, 1988; Fukushima, 1980). In this sense, the Cresceptron is a self-organizing system.

Growth of the network during learning starts at layer 1. Once a significant new feature is detected at a position (i_0, j_0) in some neural planes at layer 0, a new neuron with synaptic connections is created together with a new neural plane at level 1 and its following planes at layers 2 and 3, as illustrated in Fig. 17, all are devoted to this new feature (see also Fig. 5). The synaptic connections from layer 0 to layer 1 take the values from the corresponding neural planes that have a significantly response at the location (i_0, j_0) as defined in (20). This selective connection results in a sparsely connected network. There is also an upper bound (25 in our experimentation) on the number of preceding planes a new neural plane can connect to, with priority assigned to newer input planes. Our experiments showed that this upper bound is rarely reached, because not many different features can appear in the same image location. Thus, although the number of neural planes may grow large after learning many objects, the entire network is always sparsely connected, which is a very important point for efficiency. After creation of each neural plane, the response of this new plane is updated by computing its response.

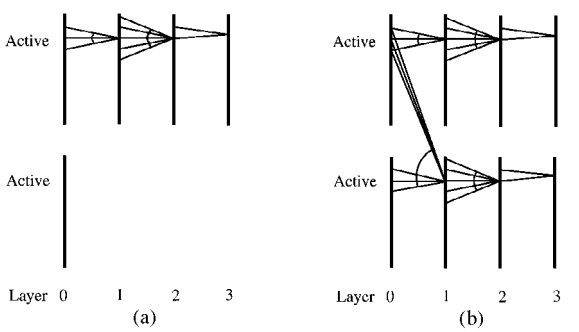

Figure 17. A 1-D illustration of growth at layers 1, 2, and 3—the first level. (a) Existing network. (b) The network after a growth.

Such a growth at layers 1, 2, 3 continues until no new feature can be detected at layer 0. Then, similar growth is performed for layer pairs 4–5, 6–7, 8–9, 10–11, 12–13, 14–15. The difference is that the first layer in each pair is now a subsampled pattern-detection layer and thus, only four subsample points at the corresponding positions of the preceding neural plane are considered. Each new neural plane created at the subsampled pattern-detection layer is followed by a new neural plane at the following node-conserved blurring layer which accepts its output.

Finally, at the top layer, if the exemplar is not recognized, a new plane is created at the top layer with a default label. The user assigns a meaningful name of the object to the label. Later in the recognition phase if this new neural plane is active at position (i, j), then the label tells the name of the object being recognized at this position.

Over the entire network, knowledge sharing occurs naturally among different positions and among different objects, since the same fea ture may appear repeatedly in various cases. The network grows only to memorize the innovation that it needs to learn.

For class learning, the user identifies the top neural plane that represents this class and then clicks button "class" instead of "learn". Thus, the system will not create a new plane at the top layer. But rather, it uses a maximization operation at the top layer to feed the response of this new exemplar to the corresponding top neural plane that represents the entire class.

5.5. *Recognition and Decision Making*

Once the network has been trained, it can be presented with unknown inputs for recognition. Figure 25 shows the response of a few neural planes. At the top layer, the network reports all the response values (confidence values) higher than 0.5. The result can be one of the following: (1) No report: nothing recognized with confidence. (2) Only one object is reported: unique object is recognized with confidence. (3) Two or more objects are reported. If they belong to different types of object at the same position, the one with the highest confidence is the one recognized. Others are not as confident. If two or more belong to the same type of object, which may occur when, for example, the input contains a face that is taken at an orientation between those that have been learned, then all the related

information can be used. For example, if the recognized objects indicate several different orientations of a face, a confidence weighted orientation sum can be used to approximately estimate the actual viewing angle of the current face, as shown in Fig. 2.

5.6. Segmentation

Once an object is recognized, the network can identify the location of the recognized object in the image. This is done by back tracking the response paths of the network from the top layer down to the lowest layer.

With the Cresceptron graphic interface, the user can easily examine the network response by "walking" through the network using buttons like "up" (to the upper layer), "down" (to the lower layer), "left" (to the left-neighbor neural plane) and "right" (to the right-neighbor neural plane). At any neural plane of the network, the response of the neural plane is shown in a subwindow at the bottom right corner of the console and its label is reported in a subwindow at the top right corner, as shown in Fig. 3. To segment a recognized object, the user clicks the button "segment" from a specified node and then, the system backtracks the response from this node by marking along the way.

Suppose a node A is marked at position (i_0, j_0) at a layer l. Whether or not the input nodes at layer $l-1$ that are linked to this node A are also marked depends on the type of the layer:

1. Layer l is a node-conserved blurring layer. In the input neural plane, check every neighboring node around (i_0, j_0) from which the blurring profile can reach A: the input node is marked if the input is active.
2. Layer l is a pattern-detection layer. The input node is marked if the connection has a high value and the input is active.
3. Layer l is a node-reduction layer. For each of the four input nodes of A, the input node is marked if it is active.

The above rules are derived from the underlying AND or OR function that each layer is to perform. Blurring of response at every level is essential for the success of segmentation, since it enables the system to mark features that moderately deviate from what is learned. As shown in Fig. 16, the extent of blurring at each level approximately equals the receptive field of the feature at that level. Thus, the blurring does not cause "bleeding" in segmentation bec ause the same feature cannot appear in the same receptive field twice.

After the marking reaches input layer 0, all the major edges that have contributed to the recognition are marked in the input attention image. This information is sufficient to account for segmentation. For display purpose, we compute a convex hull of all the marked pixels in the attention image. Then, this convex hull is mapped back to the original input image and all the pixels that fall out of this back-mapped convex hull are rep laced by a white pixel. Therefore, the remaining pixels that retain original image intensities are those that fall into the back-mapped convex hull and indicate the region of the recognized object. Note that since the object is not necessarily convex, the convex hull may include pixels that do not belong to the recognized object. But, as we know, the convex hull is just a way to show the segmented edges.

6. Experiments

For the theoretical and algorithmic development, the Cresceptron system has been simulated on a SUN SPARC workstation with an interactive user interface to allow effortless training and examination of the network, as shown in Fig. 3. The program has been written in C and its source code has about 5100 lines. The system digitizes video signals into (640×480)-pixel images or directly accepts digital images of various resolutions up to 640×512. As we discussed, the first version of the system uses directional edges in the image.

Used for training and testing are a few hundreds of sample images digitized from regular cable TV programs and a video tape about a university campus. Here we show the example of a network which has learned 20 classes of objects: faces of 10 different persons and 10 other classes of objects, as indicated in Fig. 18. This network was automatically generated through learning of these objects. Before giving a summary of the performance, which is given in Table 2 at the end of this section, we explain several examples in the training set and test set to illustrate how the variation in expression, size, viewing angle, etc. are tested while all the 20 learned classes were active.

6.1. Variation in Facial Expression

We used various human face expressions to evaluate the system tolerance. In the large pool of images, there

Table 2. Summary of performance.

Type	Classes	Training set	Test set	Unique report	Top 1 correct
Faces	10 classes	14 images	52 images	100%	100%
Nonfaces	10 classes	11 images	16 images	87%	100%
All	20 classes	25 images	68 images	97%	100%

Figure 18. Representative views of 20 classes learned by the Cresceptron network, shown as images from which the object views are extracted.

is a sequence of 35 images that were randomly digitized from a regular TV interview program "ET hostess", as displayed in Fig. 19. In order to show expressions clearly, only the face part of 32 images of the 35 is shown in Fig. 19. Interestingly, these images display how a person's expression can change drastically from one time instance to the next while talking, a very challenging phenomenon for face recognition. By learning only the first three faces from the "ET Hostess", the Cresceptron successfully recognized and reported the label "ET Hostess" at the correct location in all these images, and there was no false alarm as any of the other faces or objects that have been learned, some of which are shown in Fig. 24. In other words, all the recognitions for the faces in Fig 19 are unique (unique with confidence larger than 0.5). Note that 0.5 is considered a high response as we defined before. We can regard the response value of the node at the top layer as a confidence value, although a value 1 does not mean a complete confidence. As can be seen from the response values in Fig. 19, a large change in the position of some features does not necessarily cause a much smaller response value. For example, comparing the second face in the third row and the fourth face in the last row, although the hostess made a long face in the former case, the response is still

Figure 19. Face expression variation in the training and test images of "ET Hostess". The first three images are used to train the network. All the faces are successfully recognized at the correct image location from the entire image. Only the face part is shown here for recognized images so that the expression is clearly visible. The number under each image is the response value (or confidence value) of the recognition, i.e., the response value at the corresponding node at the top layer.

Figure 20. An example of segmentation for "ET Hostess". (a) An input image of "ET Hostess" whose face part is shown in the last subimage of the first row of Fig. 19. (b) The segmentation result as a convex hull of the contributed edge segments filled with the corresponding pixel intensity.

higher than that in the latter case. This is a very desirable property of a multi-level network: the recognition is a combination of low-layer features which have a small space extent and high layer features which have a large space extent. A feature with a large space extent should be allowed to deviate more than the one with a small space extent. In the Cresceptron, this property is embedded into the network structure. Figure 20 shows the segmentation result from an image of the "ET Hostess."

6.2. Variation in Viewing Direction

In the training set, there are three views of "Student Peter", each was taken from a different orientation. An intermediate viewing angle of "Student Peter" was tested, recognized and segmented. An examination of the network indicated that the node that reported recognition was the one created when "Student Peter view 1" was learned. Figure 21 shows the three images learned, one image tested and the result of testing.

6.3. Variation in Size

In the images we digitized, the views of "Young Actress" were from different segments of a TV show and thus the size of her face appeared in the image changes considerably from one image to another (about 50% maximum). We put a view into the training set and put remaining views into the test set. This is to test whether the consecutive size difference in the hierarchy of attention window can capture the size variation while keeping discrimination power against other 19 classes of faces and objects. Figure 22 shows the corresponding training image and three test images with different face sizes.

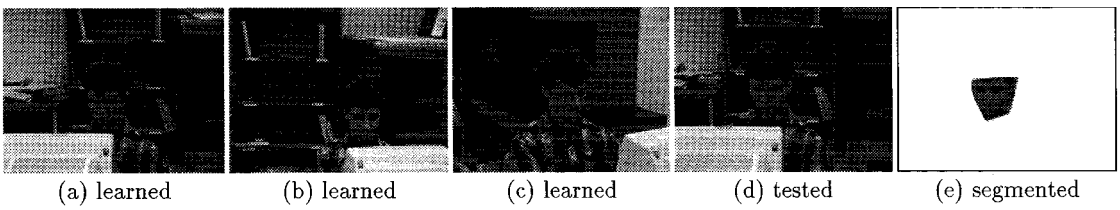

Figure 21. An example of learning different viewing angles and learning from a similar but different angle. (a) to (c) are three views that are learned. (d) is a test view recognized with a confidence value 1.00. (e) shows the segmentation result as a convex hull of the contributed edge segments filled with the corresponding pixel intensity.

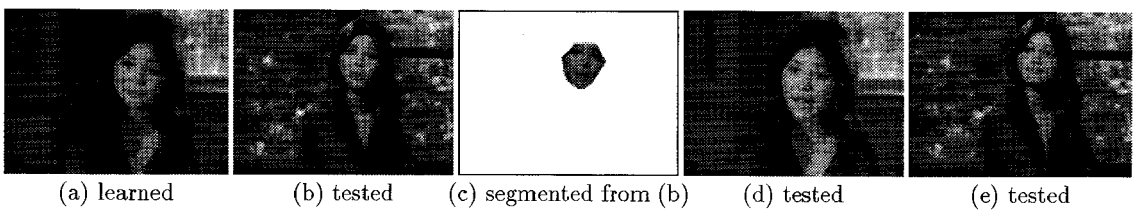

Figure 22. An example of dealing with different sizes. (a) is the single learned image for Young Actress. (b) is a test view recognized with a confidence value 0.64. (c) shows the segmentation result from (b). (d) is a test view recognized at confidence value 0.85. (e) is a test view recognized at confidence value 0.78.

Figure 23. Report of visual attention scan from the Cresceptron. The attention windows in which the object was recognized are highlighted by white squares. According to the text report from the system, there are four equal-size highlighted windows in this image (the inner small square and outer large square are a result of overlap by four equal-size windows at four different positions).

6.4. Variation in Position

To see how the attention window can find the object of interest at the right location. Figure 23 shows all the attention windows that reported a successful recognition (confidence is at least 0.5), after a complete attention scan using visual attention windows of different sizes and positions, as described in Section 2.1. Only four windows were reported as shown in the figure. This means that (1) a larger or smaller legal size of the attention window will cause a size difference that is not tolerated by the attention image recognition; (2) once a visual attention window of an appropriate size had covered the object, the recognition was successful; (3) the system did not falsely recognize any learned object in the cluttered background. Some recent studies concentrate on the detection of human faces (Yang and Huang, 1994; Sung and Poggio, 1994; Rowley et al., 1995; Swets et al., 1995).

6.5. How Other Face Models Respond?

The test set includes faces of 10 individuals, all were digitized from regular TV programs except the "Student Peter" which was from a video tape. For the cases of "ET Hostess" and "Student Peter", three images have been learned in the training of each as shown before. For other faces, only one view was learned for each case, and recognition was tested on other different views. The system successfully recognized these faces without false recognition, although some faces are not very different, such as "Young actress" and "NYSE reporter". Figure 24 shows a few examples of them. It is worth noting that the size difference between the faces is not used as a clue for discriminating different persons because the learned faces on the attention image all have a similar size. It is interesting to notice that the system tolerated the expression difference in "ET hostess" but did not tolerate the difference between, say, "Young actress" and "NYSE reporter". This behavior is brought about by the multi-level feature matching and multi-level positional tolerance.

Given any image for testing, all the 20 learned object models were active in the network. Although for every tested face image, only unique model got a confidence value higher than the threshold value 0.5 for reporting, many other models got some confidence values, as shown in the first row in Fig. 25. The figure shows an example of neural plan response at various levels when the full network learned with the 20 classes was appled to an image of "Sports reporter". The first row in the figure indicates the output level of classes. The correct class has a high response at the correct position while other classes have low responses at various positions (lower than 0.5 as observed). The increased blurring effect from low to high levels indicates the change of amount of distortion allowed by our stochastic model.

6.6. Nonface Objects

In addition to faces of 10 individuals, 10 nonface objects were learned and tested using with the same network. Each of these nonface objects was learned from one view, except for "walking dog", for which two views were learned, as shown in Fig. 26. Because we expect that these non-face objects will vary much more, a relatively lower system vigilance was used when each object was learned. In fact, only three system vigilance values were used for all the learning reported here: 0.8, 0.9, 0.99. Human faces used 0.9 or 0.99 and other objects used 0.9 or 0.8.

Figure 27 shows a few examples of learned and tested images of nonface objects and the associated segmentation result that indicates the detected location of the object in the corresponding input test image.

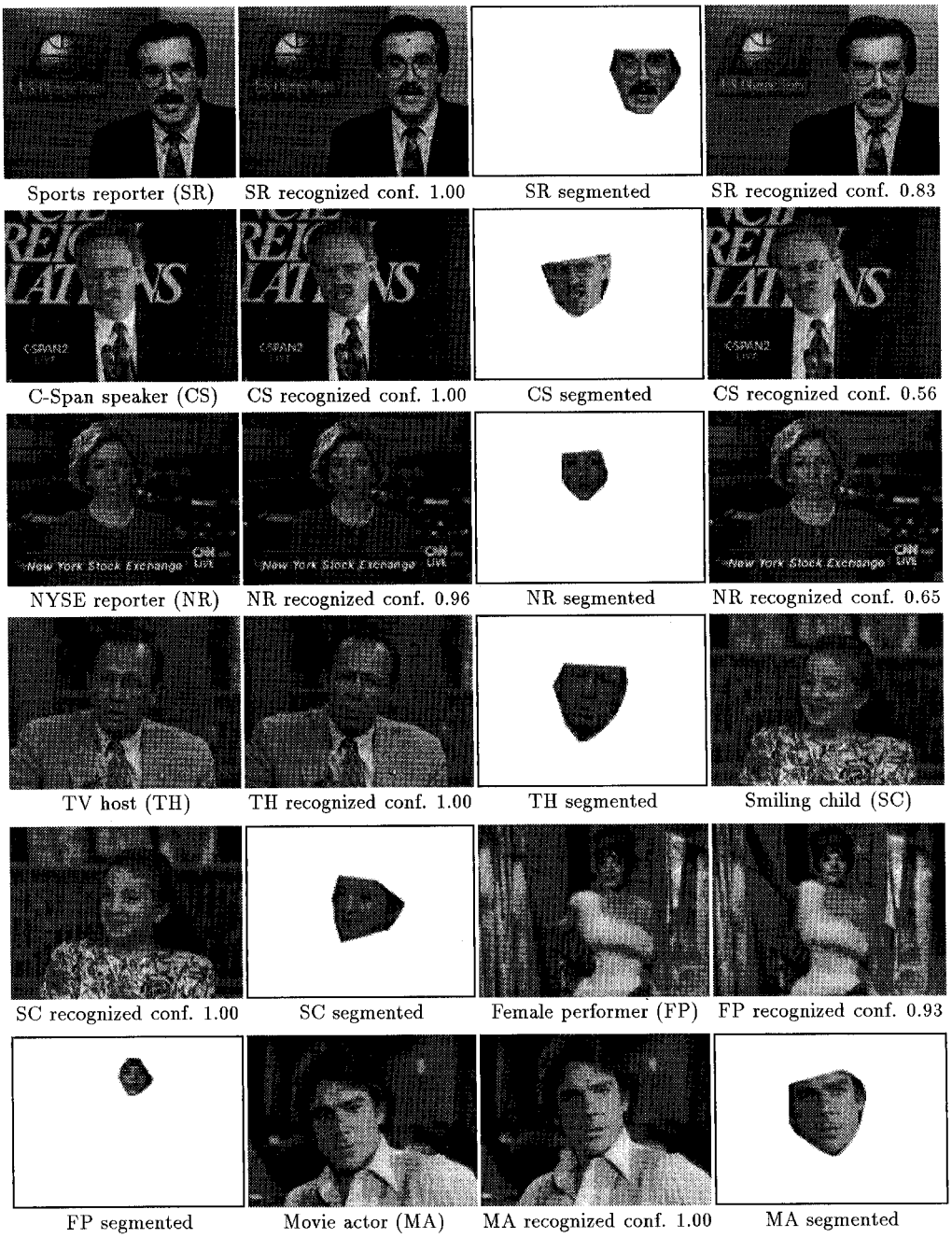

Figure 24. Examples of other trained and tested human face images with segmentation results. An image with class label only is a training image. An image with a given confidence value is a test image. An image with a word "segmented" is the result of segmentation from the previous test image.

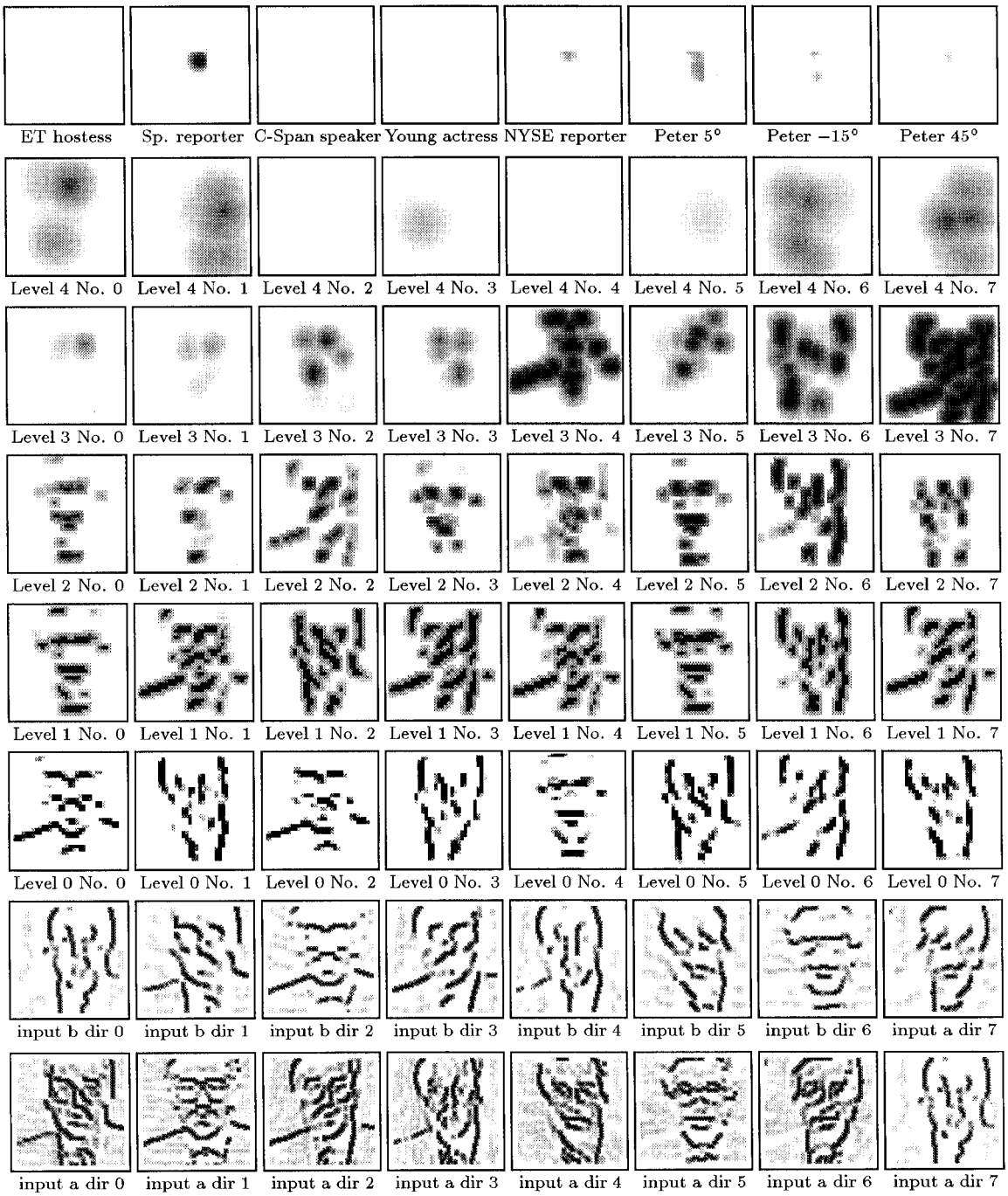

Figure 25. Response of the Cresceptron and inputs. The bottom two rows show the input edge images. The bottom row shows 8 directional edge images at the small scale and the row above it shows those with the large scale. The rows 1 to 5 from top show the first 8 neural planes in the output layers of the levels 6, 4, 3, 2, 1, respectively.

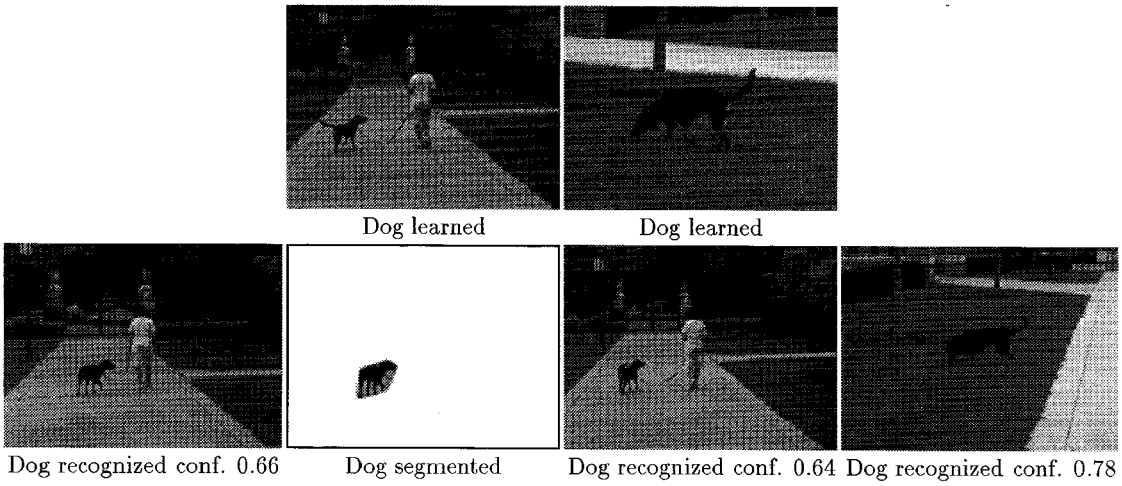

Figure 26. The "Dog" case. The first row: the learned views. The second row, test views and a segmentation result. This case involves significant changes in the shape of the object. In the learned image, the dog's tail was in sight. However, in the image to be recognized, the dog's tail was not clearly in sight (folded to its body). A low system vigilance 0.8 allows such a change, as long as other major features are still present, such as a dog's head, trunk, and legs.

6.7. Summary of the Experimental Result

In summary, 20 classes of objects shown in Fig. 18 were learned by the single automatically generated network. A total of 68 images, different from any of the training images, were used for testing. The examples from Fig. 19 to Fig. 27 indicate various examples in the training and test sets. With all the 20 classes active, the system reports all the classes with a matching confidence higher than 0.5. For nonface test images whose learning phase used a lower system vigilance, 97% got a unique recognition report. Only two nonface test images resulted in multiple reports, but the highest response (top 1) still corresponds to the correct class. One of them is the "stop sign" shown in Fig. 28. In both cases the correct class received the highest confidence value in the report.

The performance of the experiment is summarized in Table 2. The column "Unique report" in the table lists the number of cases in which one and only one class reached a confidence higher than the threshold for report (0.5). As indicated by the table, all the 20 cases were recognized correctly as top 1 confidence. However, as we know, the performance depends very much on the amount of variation between the training and test samples. The performance is expected to degrade when the number of classes increases or the difference between the learning and test samples increases. The example in Fig. 28 indicates such a degradation.

The edge segments marked by the segmentation are shown in Fig. 29 for some objects recognized. Note that all the instances here are for test views, which are different from those that are learned. These marked edge segments are the major ones that contributed to the recognition. Other minor edge segments may also have contributed to the recognition to a lesser degree. But, as displayed in Fig. 29, background edges, although they may be strong, were not marked in the attention images. It is worth noting that the edge segments shown in the figure are not completely connected and some are missing, a situation consistent with input edge images. Fortunately, the system does not rely on a close outline of the object, which is very difficult to extract, but rather uses grouping of the edge segments in an automatic and hierarchical manner. In addition to object outlines, the Cresceptron also implicitly utilized surface markings, self-occluding edges, texture, etc. An integration of visual cues seems natural in this fashion.

Figure 30 plots the growth of the number of neural planes at a few layers: 3, 5, 7, and 9, versus the the number of views learned. At layers 1 and 2, each neural plane has 64×64 nodes, and at higher layers, each neural plane has 32×32 nodes. After learning these 20 classes of objects, the network has a total of 4133 neural planes in these layers. Note that each neural plane needs to store only a single 3×3 template in the database. The size of the database of these 20 classes of objects is around 0.68 Mbytes.

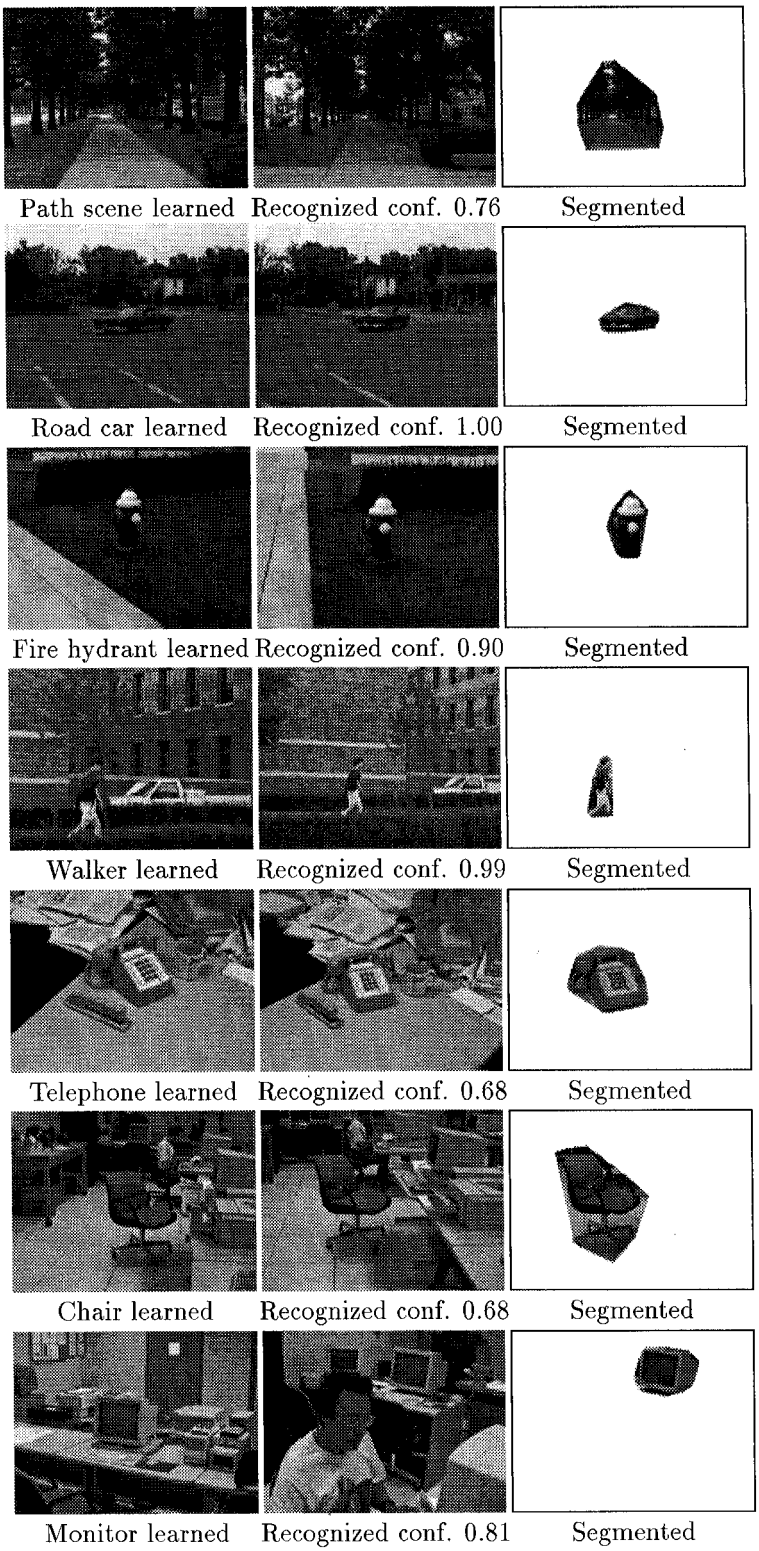

Figure 27. Some examples of trained and tested nonface images with segmentation results. The left column gives the training images; the middle column lists the corresponding test images and the right column show the corresponding segmentation result for the image in the middle column.

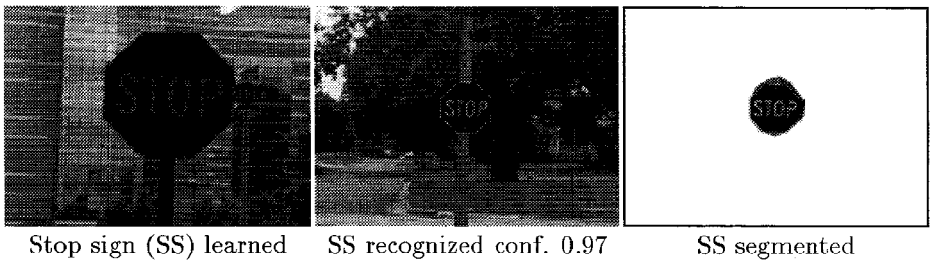

Figure 28. One of the two test images where two models were reported given the input image. The correct "stop sign" is reported with a confidence value 0.97. The second report was "dog" with a confidence value 0.66. The segmented result indicated that an "illusionary" dog was "found" at the upper half of the stop-sign board, where the vertical strokes of the word "STOP" were regarded as the dog's legs and the upper edges of the octagonal board as its back.

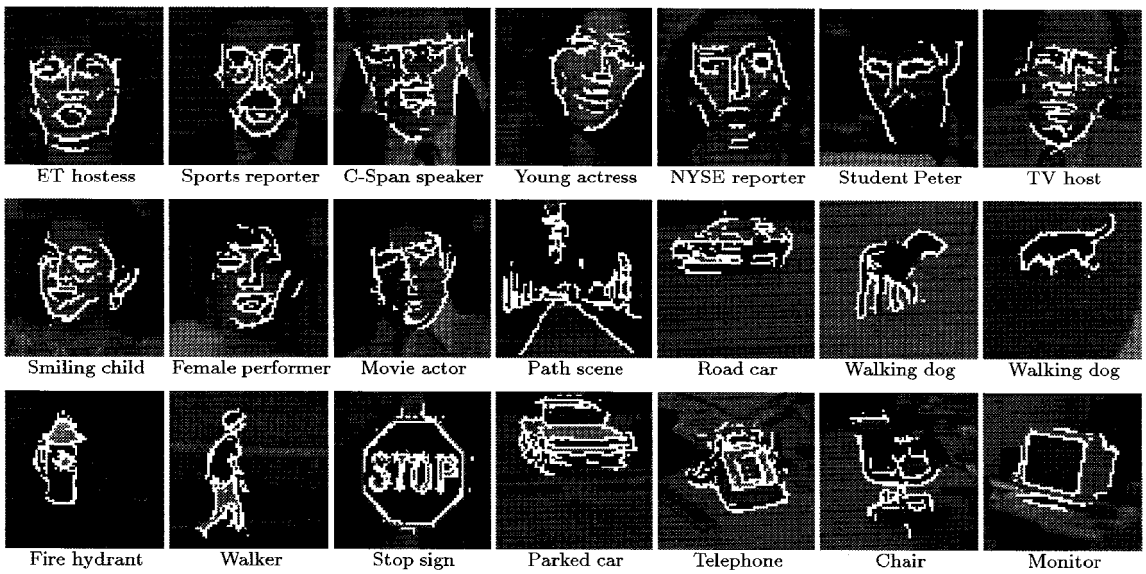

Figure 29. Edge segments marked by the segmentation process for some of the examples. These are the major edge segments contributing to the recognition.

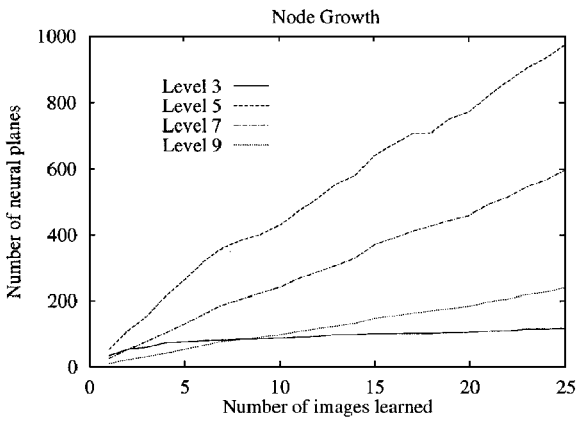

Figure 30. Network growth versus the number of images learned.

7. Discussion and Conclusions

7.1. Biological Considerations

Fukushima discussed the link between such multilayer network structures and the concept of visual cortex hierarchy: LGB (lateral geniculate body), simple cells, complex cells, lower order hyper complex cells, higher order hyper complex cells (Fukushima, 1980). If we consider all the input layers that connect to a pattern detection layer as a stack of neural planes. The region in this stack that links to a single node in the pattern detection layer constitutes a column of "cells", which contains various feature extraction cells but all are situated at the same retinotopic position. This column

closely resembles the concept of "hypercolumns" proposed by Hubel and Wiesel (1977).

In the Cresceptron, learning involves two types of network plasticity: modification of the synaptic weights (synaptic plasticity) and changes in connectivity of synapses (anatomical plasticity). Hebb (1949) proposed that selective activation of synapses in synchrony with the postsynaptic neuron could lead to long-lasting changes in the effectiveness of specific synaptic connections. Recent work by Desmond and Levy (1988) suggests that such changes in synaptic efficacy does occur with associated forms of learning in some neural systems. Anatomical plasticity, on the other hand, is not only a major characteristic in brain's fetal development (see, e.g., Shatz, 1992; Rakic, 1988), but also correlates learning in maturity (Kandel and Schwartz, 1982; Carew, 1989) and recovery to various degrees following an injury (Guth, 1975; Carbonetto and Muller, 1982).

7.2. Efficiency Issues

Efficiency is important to dealing with a large number of objects. With the Cresceptron, it is clear from Fig. 30 that the growth of layer 3 (also layers 1 and 2) saturated very soon. This is because that at these low layers, the number of possible features is very limited in a 3×3-pixel neighborhood. The system learned all the possible *significant* cases very soon. At layer 5, the number of possible features must be much larger, because it has a larger receptive field. The growth at layer 5 will slow down when the number of interested *significant* features are nearly exhausted, as far as the network is concerned. It is not clear where this will take place. It depends very much on the size and variation of the class of objects that a particular application wants to learn. When learning a small class of object with a limited variation, this exhaustion of interested significant features should occur earlier. Otherwise, it takes place later. The saturation at layer 7 will occur much later. Conceivably, the number of nodes at higher layers will be almost proportional to the number of objects that have been learned.

It is obvious that a system cannot keep learning forever without forgetting anything, due to a limited memory size. The current implementation of the Cresceptron also allows the nodes to be deleted. That is, the network "forgets" less frequently used or very "old" knowledge by deleting the corresponding nodes and thus keeps its size limited. This "forgetting" mechanism might be useful for correcting errors due to early (immature) learning. With the Cresceptron, a flag is kept in each node to record its usage, so that only those less oftenly used nodes are deleted when system capacity reaches the system limit.

7.3. Conclusions

The objective and approach of the Cresceptron are very different from other works in that it is not restricted to a particular class of objects and it works directly with natural intensity images. Due to its generality, it is applicable to human face images as well as other living and nonliving objects, including difficult things such as dogs. The segmentation scheme is based on matched edge grouping between the learned network and the in put, with hierarchical toleration for edge positions. Our analytical result has shown that a hierarchical stochastic deformation model can model stochastic behaviors that a single-level model can not.

Using a conventional method that uses hand-crafted vision rules, one typically needs to impose conditions on the object and scene environment so that the vision-rules are applicable. However, it is very difficult to provide an automatic system that is capable of verifying these conditions, given any image. The result shown in this paper seems to indicate that it is possible to approach vision with an open-ended learning system, which imposes little constraint on what it can see and can improve itself through incremental learning. One major difficulty that the Cresceptron faces is the efficiency. A later system SHOSLIF (Weng, 1996) addresses this and other issues.

Appendices

Appendix A

Proof of Theorem 1: The set of all real numbers consists of rational and irrational numbers. We first prove (10) for any rational number $d \in \Re_+$. Then, we will prove the same equation for any irrational number $d \in \Re_+$.

For any non-negative integer n, we define $\alpha_i = \alpha(id/n)$ and then divide $[0, d]$ into n segments as follows:

$$\alpha(d) - \alpha(0) = \sum_{i=1}^{n} (\alpha_i - \alpha_{i-1}) = \sum_{i=1}^{n} \delta_i(d)$$

where $\delta_i = \alpha_i - \alpha_{i-1}$. Using the Markovianity, δ_i being mutually independent between different i's given $\alpha_0, \ldots, \alpha_{n-1}$, it follows that

$$\begin{aligned} v^2(d) &= \text{var}[\alpha(d) - \alpha(0)] \\ &= E\left[E\left[\sum_{i=1}^{n}(\delta_i - E[\delta_i])^2 \mid \alpha_0, \ldots, \alpha_{n-1}\right]\right] \\ &= E\left[\sum_{i=1}^{n} E[(\delta_i - E[\delta_i])^2 \mid \alpha_0, \ldots, \alpha_{n-1}]\right] \\ &= \sum_{i=1}^{n} E[E[(\delta_i - E[\delta_i])^2 \mid \alpha_0, \ldots, \alpha_{n-1}]] \end{aligned} \tag{21}$$

It follows from (9) that

$$\begin{aligned} &E[E[(\delta_i - E[\delta_i])^2 \mid \alpha_0, \ldots, \alpha_{n-1}]] \\ &\quad = E[E[(\delta_i - E[\delta_i])^2 \mid \alpha_0, \ldots, \alpha_{i-1}]] \end{aligned}$$

Using the Markovianity and the homogeneity, the right-hand side is equal to

$$\begin{aligned} &E[E[(\delta_i - E[\delta_i])^2 \mid \alpha_{i-1}]] \\ &\quad = E[E[(\delta_1 - E[\delta_1])^2 \mid \alpha_0]] \\ &\quad = E[\text{var}[\delta_1 \mid \alpha_0]] \\ &\quad = E[\text{var}[(\alpha(d/n) - \alpha(0)) \mid \alpha_0]] \\ &\quad = \text{var}[\alpha(d/n) - \alpha(0)] \end{aligned} \tag{22}$$

Replacing the above expression for the terms under summation in (21) gives

$$\begin{aligned} v^2(d) &= \sum_{i=1}^{n} \text{var}[\alpha(d/n) - \alpha(0)] \\ &= \sum_{i=1}^{n} v^2(d/n) \\ &= n v^2(d/n) \end{aligned} \tag{23}$$

Rewriting (23), we get

$$v^2(d/n) = \frac{1}{n} v^2(d) \tag{24}$$

Given any integer $m \geq 0$, replacing n in (23) by m and letting $d = m$ gives

$$v^2(m) = m v^2(1)$$

Letting $d = m$ in (24) and using the above equation yields

$$v^2\left(\frac{m}{n}\right) = \frac{1}{n} v^2(m) = \frac{m}{n} v^2(1)$$

Because any nonnegative rational number x can be written as $x = m/n$, for some non-negative integer numbers m and n, the above equation means that (10) holds true for all rational numbers in $\Re_+$.

As we know, the set of all rational numbers is dense in $\Re$ (Royden, 1988). This implies that for any irrational number x in $\Re_+$, there exist two rational series $\{x_n^-\}$ and $\{x_n^+\}$ (i.e., all the numbers in $\{x_n^-\}$ and $\{x_n^+\}$ are rational numbers), with

$$x_n^- < x < x_n^+$$

for all integer $n > 0$, such that $x_n^- \to x$ and $x_n^+ \to x$ as $n \to \infty$.

Next, we want to prove that $v^2(x)$ is monotonically nondecreasing. That is,

$$v^2(x_n^-) \leq v^2(x) \leq v^2(x_n^+) \tag{25}$$

for all integer $n > 0$. We just need to prove that $v(x) \leq v(x')$ for any $x < x'$. In fact, using the same technique we used while proving (23) but changing the breaking points from $x_i = id/n$ to $x_0 = 0, x_1 = x, x_2 = x'$, similar to (23) we have

$$\begin{aligned} v^2(x') &= \text{var}[\alpha(x' - x) - \alpha(0)] + \text{var}[\alpha(x) - \alpha(0)] \\ &\geq \text{var}[\alpha(x) - \alpha(0)] \\ &= v^2(x) \end{aligned} \tag{26}$$

which proves (25). Using the proved result $v^2(x) = x \cdot v^2(1)$ for rational number x, (25) gives

$$x_n^- v^2(1) \leq v^2(x) \leq x_n^+ v^2(1)$$

By taking the limit for $n \to \infty$ on every side of the above inequality, we get

$$\begin{aligned} x \cdot v^2(1) &= \lim_{n\to\infty} x_n^- v^2(1) \leq v^2(x) \leq \lim_{n\to\infty} x_n^+ v^2(1) \\ &= x \cdot v^2(1) \end{aligned}$$

Therefore $v^2(x) = x \cdot v^2(1)$ holds true for any irrational number x in $\Re$ as well. Since $v(x')$ and $v(x)$ are nonnegative and $x \in \Re$, $v^2(x) = x \cdot v^2(1)$ implies $v(x) = \sqrt{x} v(1)$. □

Appendix B

Proof of Corollary 1: We suppose $x \leq x'$. The proof for $x > x'$ is analogous. If $d = x' - x$ is a rational number, the proof is analogous to that for Theorem 1, and thus, is omitted. If $d = x' - x$ is an irrational number, there is a rational series $\{x_n\} \to x' - x$ as $n \to \infty$. Because $w(0, d)$ is continuous in d, we have

$$\begin{aligned}(x' - x)w(0, 1) &= \lim_{n\to\infty} x_n w(0, 1) \\ &= \lim_{n\to\infty} w(0, x_n) \\ &= w(0, x' - x) = w(x, x') \qquad \square\end{aligned}$$

Acknowledgments

Thought-provoking discussions with Drs. M.-Y. Chiu, S. J. Hanson, C. L. Giles, A. K. Jain, S. Y. Kung, A. Singh and G. Stockman are gratefully acknowledged. The authors would like to thank Ms. S. Howden and L. Blackwood for proof-reading the manuscript.

Note

1. Take language understanding as an example. If we just remember the meaning of every sentence, we treat each sentence as a black box. But, if we know how each word is formed from letters and how each sentence is constructed from words, and different sentences may share the same words and phrases, we are not treating a sentence as an opaque box.

References

Anderson, J.R. 1990. *Cognitive Psychology and Its Implications*. 3rd edition, Freeman: New York.

Arman, F. and Aggarwal, J.K. 1991. Automatic generation of recognition strategies using CAD models. In *Proc. IEEE Workshop on Directions in Automated CAD-Based Vision*, pp. 124–133.

Bichsel, M. 1991. Strategies of robust object recognition for the automatic identification of human faces. Ph.D. thesis, Swiss Federal Institute of Technology, Zurich, Switzerland.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. 1984. *Classification and regression trees*. Wadsworth, CA.

Brooks, R.A. 1981. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence*, 17(1–3):285–348.

Carbonetto, S. and Muller, K.J. 1982. Nerve fiber growth and the cellular response to axotomy. *Current Topics in Developmental Biology*, 17:33–76.

Carew, T. J. 1989. Developmental assembly of learning in aplysia. *Trends in Neurosciences*, 12:389–394.

Carey, S. 1985. *Conceptual Change in Childhood*. The MIT Press: Cambridge, MA.

Chen, C. and Kak, A. 1989. A robot vision system for recognizing 3-D objects in low-order polynomial time. *IEEE Trans. Systems, Man, and Cybernetics*, 19(6):1535–1563.

Cover, T.M. Learning in pattern recognition. *Methodologies of Pattern Recognition*, in S. Watanabe (Ed.), Academic Press: New York, pp. 111–132.

Cover, T.M. and Hart, P.E. 1967. Nearest neighbor pattern classification. *IEEE Trans. Information Theory*, IT-13:21–27.

Desmond, N.L. and Levy, W.B. 1988. Anatomy of associative long-term synaptic modification. *Long-Term Potentiation: From Biophysics to Behavior*, in P.W. Landfield and S.A. Deadwyer (Eds.), Alan R. Liss, New York, pp. 265–305.

Dreher, B. and Sanderson, K.J. 1973. Receptive field analysis: Responses to moving visual contours by single lateral geniculate neurons in the cat. *Journal of Physiology*, London, 234:95–118.

Faugeras, O.D. and Hebert, M. 1986. The representation, recognition and location of 3-D objects. *Int'l J. Robotics Research*, 5(3):27–52.

Forsyth, D., Mundy, J.L., Zisserman, A., Coelho, C., Heller, A., and Rothwell, C. 1991. Invariant descriptors for 3-D object recognition and pose. *IEEE Trans. Pattern Anal. and Machine Intell.*, 13(10):971–992.

Fu, K.S. 1968. *Sequential methods in Pattern Recognition and Machine Learning*, Academic Press: New York.

Fukushima, K. 1975. Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, 20:121–136.

Fukushima, K. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202.

Fukushima, K., Miyake, S., and Ito, T. 1983. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Trans. Systems, Man, Cybernetics*, 13(5):826–834.

Gool, L.V., Kempenaers, P., and Oosterlinck, A. 1991. Recognition and semi-differential invariants. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*. pp. 454–460.

Grimson, W.E.L. and Lozano-Perez, T. 1984. Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35.

Guth, L. 1975. History of central nervous system regeneration research. *Experimental Neurology*, 48(3–15).

Hansen, C. and Henderson, T.C. 1989. CAGD-based computer vision. *IEEE Trans. Pattern Anal. and Machine Intell.*, 10(11):1181–1193.

Hebb, D.O. 1949. *The organization of behavior*. Wiley: New York.

Hubel, D.H. 1988. *Eye, Brain, and Vision*. Scientific American Library, 22.

Hubel, D.H. and Wiesel, T.N. 1977. Functional Architecture of macaque monkey visual cortex. *Proc. Royal Society of London*, Ser. B, Vol. 198, pp. 1–59.

Huttenlocher, D.P. and Ullman, S. 1987. Object recognition using alignment. In *Proc. Int'l Conf. Computer Vision*, London, England, pp. 102–111.

Highleyman, W.H. 1962. Linear decision functions, with application to pattern recognition. *Proc. IRE*, Vol. 50, pp. 1501–1514.

Iarbus, A.L. 1967. *Eye Movements and Vision*. Plenum Press: New York.

Ikeuchi, K. and Kanade, T. 1988. Automatic generation of object recognition programs. In *Proc. IEEE*, Vol. 76, No. 8, pp. 1016–1035.

Jain, A.K. 1989. *Fundamentals of Digital Image Processing*. Prentice Hall: New Jersey.

Jain, A.K. and Dubes, R.C. 1988. *Algorithms for Clustering Data*. Prentice-Hall: New Jersey.

Jain, A.K. and Hoffman, R.L. 1988. Evidence-based recognition of 3-D objects. *IEEE Trans. Pattern Anal. and Machine Intell.*, 10(6):783–802.

Kandel, E. and Schwartz, J.H. 1982. Molecular biology of learning: Modulation of transmitter release. *Science*, 218:433–443.

Keehn, D.G. 1965. A note on learning for Gaussian properties. *IEEE Trans. Information Theory*, IT-11:126–132.

Kohonen, T. 1988. *Self-Organization and Associative Memory*. 2nd edition, Springer-Verlag: Berlin.

Kolers, P.A., Duchnicky, R.L., and Sundstroem, G. 1985. Size in visual processing of faces and words. *J. Exp. Psychol. Human Percept. Perform.*, 11:726–751.

Lamdan, Y. and Wolfson, H.J. 1988. Geometric hashing: A general and efficient model-based recognition scheme. In *Proc. 2nd International Conf. Computer Vision*, pp. 238–246.

Lévy-Schoen, A. 1981. Flexible and/or rigid control of oculomotor scanning behavior. In J.W. Senders (Ed.), *Eye Movements: Cognition and Visual Perception*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 299–314.

Lippmann, R.P. 1987. An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4(2):4–22.

Loftsgaarden, D.O. and Quesenberry, C.P. 1965. A nonparametric estimate of a multivariate density function. *Ann. Math. Stat.*, 36:1049–1051.

Lowe, D.G. 1985. *Perceptual Organization and Visual Recognition*. Kluwer Academic: Hingham, MA.

Martinez, J.L., Jr. and Kessner, R.P. (Eds.) 1991. *Learning and Memory: A Biological View*. 2nd edition, Academic Press: San Diego.

Michalski, R., Mozetic, I., Hong, J., and Lavrac, N. 1986. The multipurpose incremental learning system AQ15 and its testing application to three medical domains. In *Proc. Fifth Annual National Conf. Artificial Intelligence*, Philadelphia, PA, pp. 1041–1045.

Nazir, T.A. and O'Regan, J.K. 1990. Some results on translation invariance in the human visual system. *Spatial Vision*, 5(2):81–100.

Pavlidis, T. 1992. Why progress in machine vision is so slow. *Pattern Recognition Letters*, 13:221–225.

Poggio, T. and Edelman, S. 1990. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266.

Pomerleau, D.A. 1989. ALVINN: An autonomous Land Vehicle in a Neural Network. *Advances in Neural Information Processing*, in D. Touretzky (Ed.), Vol. 1, pp. 305–313, Morgran-Kaufmann Publishers: San Mateo, CA.

Quinlan, J. 1986. Introduction of Decision Trees. *Machine Learning*, 1:81–106.

Pavlidis, T. 1977. *Structural Pattern Recognition*. Springer-Verlag: New York.

Rakic, P. 1988. Specification of cerebral cortical areas. *Science*, 241:170–176, 1988.

Ramachandran, V.S. 1990. Perceiving shape from shading. *The Perceptual World*, in I. Rock (Ed.), Freeman: San Francisco, CA, pp. 127–138.

Rowley, H.A., Baluja, S., and Kanade, T. 1995. Human face detection in visual scenes. Report CMU-CS-95-158, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Royden, H.L. *Real Analysis*. Macmillan: New York.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J. 1986. Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*, in D.E. Rumelhart and J.L. McClelland (Eds.), MIT Press, MA.

Sacks, O. 1993. To see and not see. *The New Yorker*, pp. 59–73.

Sato H. and Binford, T.O. 1992. On finding the ends of straight homogeneous generalized cylinders. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Urbana, IL, pp. 695–698.

Shatz, C.J. 1992. The developing brain. *Scientific American*, pp. 61–67.

Stein, F. and Medioni, G. 1992. Structural indexing: Efficient 3-D object recognition. *IEEE Trans. Pattern Anal. and Machine Intell.*, 14(2):125–144.

Sung, K. and Poggio, T. 1994. Example-based learning for view-based human face detection. A.I. Memo 1521, CBCL paper 112, MIT.

Swets, D., Punch, B., and Weng, J. 1995. Genetic algorithm for object recognition in a complex scene. In *Proc. Int'l Conf. on Image Processing*, Washington, D.C., pp. 22–25.

Thompson, P. 1980. Margaret Thatcher: a new illusion. *Perception*, 9:483–484.

Treisman, A.M. 1983. The role of attention in object perception. *Physical and Biological Processing of Images*, in O.J. Braddick and A.C. Sleigh (Eds.), Springer-Verlag: Berlin.

Turk, M. and Pentland, A. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86.

Weiss, I. 1993. Geometric invariants and object recognition. *Int'l Journal of Computer Vision*, 10(3):207–231.

Weng, J. 1993. On the structure of retinotopic hierarchical networks. In *Proc. World Congress on Neural Networks*, Portland, Oregon, Vol. IV, pp. 149–153.

Weng, J. 1996. Cresceptron and SHOSLIF: Toward comprehensive visual learning. In S.K. Nayar and T. Poggio (Eds.), *Early Visual Learning*, Oxford University Press: New York.

Weng, J., Ahuja, N., and Huang, T.S. 1992. Cresceptron: A self-organizing neural network which grows adaptively. In *Proc. International Joint Conference on Neural Networks*, Baltimore, Maryland, Vol. I, pp. 576–581.

Weng, J., Ahuja, N., and Huang, T.S. 1993. Learning recognition and segmentation of 3-D objects from 2-D images. In *Proc. 4th International Conf. Computer Vision*, Berlin, Germany, pp. 121–128.

Wilson, H.R. and Giese, S.C. 1977. Threshold visibility of frequency gradient patterns. *Vision Research*, 17:1177–1190.

Wilson, H.R. and Bergen, J.R. 1979. A four mechanism model for spatial vision. *Vision Research*, 19:19–32.

Yang, G. and Huang, T. S. 1994. Human face detection in a complex background. *Pattern Recognition*, 27(1):53–63.