The VISIONS Image Understanding System - 1986

Allen R. Hanson
Edward M. Riseman

COINS Technical Report 86-62

December 1986

This research was supported by the Air Force Office of Scientific Research under grant F49620-83-C-0099, by the Defense Advanced Research Agency under contracts N00014-82-K-0464 and DACA76-85-C-0088, and by the National Science Foundation under grant DCR-8318776.

THE VISIONS IMAGE UNDERSTANDING SYSTEM - 1986

A. Hanson
E. Riseman

Computer & Information Sciences Department
University of Massachusetts
Amherst, MA

This work has been supported by the Air Force Office of Scientific Research under grant F49620-83-C-0099, by the Defense Advanced Research Projects Agency under contracts N00014-82-K-0464 and DACA76-85-C-0008, and by the National Science Foundation under grant DCR-8318776.

Abstract

In this paper we consider some of the problems confronting the development of general integrated computer vision systems, and the status of the VISIONS project which has become an experimental testbed for the construction of knowledge-based image interpretation systems. The goal is the construction of a symbolic representation of the three-dimensional world depicted in a two-dimensional image, including the labeling of objects, the determination of their location in space, and to the degree possible the construction of a surface representation of the environment.

Our system involves three levels of processing for static image interpretation. Low-level processes manipulate pixel data and produce intermediate symbolic events such as regions and lines with their attributes. High-level processes focus attention on aggregates of these events via rule-based object hypotheses in order to selectively invoke schemas, which contain more complex knowledge-based interpretation strategies. Intermediate-level processes carry out grouping and reorganization of the error-prone symbolic representation extracted from the sensory data, utilizing both "top-down" control of the processing by the schema interpretation strategies as well as "bottom-up" data-directed organization of interesting perceptual events. Our design is being extended to integrate the results of motion and stereo processing throughout the three levels of processing, with depth arrays at the lowest level, partially correct surfaces at the intermediate level, as well as 2D and 3D motion attributes where appropriate.

In addition, a highly parallel three-level associative architecture is being developed to achieve real-time dynamic vision capabilities by allowing a bi-directional flow of information and processing across the levels. The machine will employ MIMD processing at the high level and Multi-SIMD (i.e. parallel local SIMD) under MIMD control at both the intermediate and low-levels.

Contents

1 INTRODUCTION

This paper will discuss a range of problems in computer vision and describe the VISIONS system,[1] an approach to the construction of a general vision system that relies on knowledge-based techniques for image interpretation. The goal of this effort is the construction of a system capable of interpreting natural images of significant complexity. Over the past twelve years, the VISIONS group at the University of Massachusetts has been evolving the system while applying it to natural scenes, such as house and road scenes, aerial images, and biomedical images. Our philosophy is that it is reasonable to expect that each new domain will require a different knowledge base, but most of the system should remain the same across task domains. This paper documents the status of the system in mid-1986 as its development continues. Note that we do not attempt to carefully survey the literature on knowledge-based vision; partial reviews may be found in [16,17,29,31,54] and some representative individual research efforts are described in [13,15,18,19,32,42,48,63,70,82,88,89,92,93,99,100,104,107,110,119,122,123,128,133,134].

1.1 The Problem of Image Interpretation

Our research has concentrated to a large extent on the identification of objects in static color images of natural scenes by associating 2D image events with object descriptions [19,20,21,53,54,55,56,57,58,59,60,61,72,106,113,114,115,116,117,118,148,149,150]. However, at a more general level, the VISIONS design utilizes many stages of processing in the transformation from "signals" to "symbols", or to use more specific terminology, from 2D image events to object labels and 3D hypotheses. Unless the domain is extremely simple and heavily constrained so that object matching processes can be applied directly to the image (e.g. via template matching), there must be some form of sensory processing which extracts information from an image to produce an intermediate representation. This representation must then be refined and associated with semantic

[1] VISIONS: Visual Integration by Semantic Interpretation of Natural Scenes

events by making use of general knowledge and constraints provided by the physical world represented in the scene domains of interest. This implies that the intermediate events must be mapped to hypotheses about the content and structure of the scene. Once a partial mapping is achieved a variety of interpretation strategies can be employed for matching, verifying, and refining contextual structures, and for grouping incomplete and ambiguous image data into organized perceptual structures.

The construction of a 3D representation of the environment from a static image is an additional aspect of our work. We have made some progress towards this goal [55,98,149,160,161], but it remains a difficult problem. While the results presented in this paper will focus more on the problem of object labelling in a static monocular 2D image, the system presented here can be naturally extended to include stereo and/or motion analysis of multiple images. The presence of 3D data significantly improves the ability to develop a 3D representation of surfaces and would make the interpretation process easier and more robust, although the problem is still extremely challenging. Thus, our current system is being extended to use depth maps of the visual field produced by motion and stereo algorithms as part of the 3D interpretation process.

1.2 Ambiguity and Context

The complexity of the visual task can be made explicit by examining almost any image of a natural scene. Even though it is very difficult to be introspective of one's own visual processing, we believe the conjectures and qualitative discussion in this subsection are intuitive and reasonable.

Humans are rarely aware of any significant degree of ambiguity in local portions of the sensory data, nor are they aware of the degree to which they are employing more global context and stored expectations derived from experience [26,27,131,132]. However, if the visual field is restricted so that only local information about an object or object-part is available, interpretation is often difficult or impossible. Increasing the contextual information, so that spatial relations to other objects and object-parts are present, makes the perceptual task seem natural and simple. Consider the scenes

in Figure 1.1 and the closeup images in Figure 1.2. In each case subimages of objects have been selected which show:

- "primitive" image events — image events which convey limited information about the decomposition of an object into its parts (which of course is a function at least partly of resolution); note that this implies that the path to recognition of the object via subparts is not available to our perceptual system;
- absence of context — information about other objects which might relate to the given object in expected ways is limited; note that this implies that the path to recognition of the object, via the scenes or objects of which it is a part, is not available to our perceptual system.

In Figure 1.2, as some of the surrounding context of the shoes and the head are supplied, the perceptual ambiguity disappears and the related set of visual elements is easily recognized. In each of the above cases the purely local hypothesis is inherently unreliable and uncertain, and there may be little surface information to be derived in a bottom-up manner. It appears that human vision is fundamentally organized to exploit the use of contextual knowledge and expectations in the organization of the visual primitives. However, it may be impossible to associate object labels with these ambiguous primitives until they are grouped into larger entities and collectively interpreted as a related set of object or scene parts [28]. Thus, the inclusion of knowledge-driven processes at some level in the image interpretation task, where there is still a great degree of ambiguity in the organization of the visual primitives, appears inevitable.

We conjecture that image interpretation initially proceeds by forming an abstract representation of important visual events in the image without knowledge of its contents. These primitive elements will be associated with tokens (forming a symbolic representation of the image data) which are then collected, grouped, and refined to bring their collective description into consistency with high-level semantic structures that represent the observer's knowledge about the world.

1.3 Issues in Image Interpretation

There are a variety of issues that must be addressed if robust vision systems are to be developed. We have identified some of the key problems that we believe are critical to the interpretation process

and which are addressed in the following sections. The reader should note, however, that we are not attempting to discuss all of the issues that are open in vision, since it would lead to a paper far longer than this one. Thus, in summary, we have abstracted the following set of overlapping problems that must be faced:

- Reliability of Segmentation Processes

  Segmentation, or low-level, processes vary in quite unreliable and unpredictable ways, due to a variety of confounding factors which include complex uncontrolled lighting, highlights and shadows, texture, occlusion, complex 3D shapes, and digitization effects.

- Uncertainty

  There is inherent uncertainty in every stage of processing; as the data is transformed it must be kept constrained in a way that preserves some degree of integrity and avoids large search spaces.

- Information Fusion

  Mechanisms are needed for integrating multiple sources of information, including multiple sensory sources, multiple representations output from low-level processes, and evidence from multiple knowledge sources.

- Representation

  It is difficult to represent natural objects and scenes which exhibit a tremendous variability in their 3D shape, color, texture, and size and hence in many of their measurable characteristics in the 2D image, yet such a representation is clearly needed in order to capture the structural, geometric, and spatial information required for visual perception.

- Local vs Global Processing in Matching

  There is a general need to provide information about the global context to the more localized processes attempting to reach some decision about a portion of the image; the process of hypothesis generation must work in the face of incomplete and inconsistent information.

- Inference

  Computational machinery is needed for assessing the indirect implication of direct, but uncertain, evidence; this inferencing capability must be able to deal with all of the issues mentioned above.

1.4 Difficulties with Segmentation

Probably the most fundamental problem blocking knowledge-based vision development has been the lack of stable low-level vision algorithms that can produce a reasonably useful intermediate representation. Vision systems must deal with the problem of dynamically transforming the massive

amounts of sensory data (in an image of reasonable resolution there are $512 \times 512 \cong 1/4$ million pixels) into a much smaller set of image events, such as regions of homogeneous color and texture, straight lines, and local surface patches, to which hypotheses will be attached.

There is little doubt that the segmentation problem[2] is a very difficult and ill-formed problem [56]. There is no ideal or "correct" segmentation because that is a function of the goals of the interpretation system. The 2D appearance of objects and their parts is affected by variations in lighting, perspective distortion, varying point of view, occlusion, highlights, and shadows. In addition objects and their parts may or may not be visually distinguishable from the background. Thus, one cannot predict what initial image information can be extracted that is relevant to the recognition of objects. The only thing that should be counted on is that the process will remain inherently unreliable; certainly some useful information can be extracted, but many errors will also occur, and no optimal solution exists in any non-trivial image domain. In general there is no set of parameter settings for any algorithm which will extract the desired image events without also generating additional non-optimal or undesirable events. For a given parameter setting, a region segmentation might be too fragmented in one area of an image (i.e. too many regions) while being overmerged in another area of the image (i.e. too few regions). As parameters are varied the partitioning will change, but never will a result be produced that is optimal or near-optimal throughout the scene. The same is true of line and surface extraction algorithms; they will produce fragmented and overmerged events in a varied and unpredictable manner.

1.5 Segmentation vs. the Recovery of 3D Surfaces

For the past decade there has been major controversy in the field of computer vision concerning the meaningfulness and validity of segmentation. Some researchers seem to restrict their criticisms to the actual partitioning of images via region segmentation, but feel more open to the process

[2]Note that we sometimes will use the term segmentation loosely to cover not only the usual definition of partitioning an image into connected, non-overlapping sets of pixels, but also other low-level processes such as line extraction.

of line extraction. It is our strong opinion that while both suffer from similar types of unreliability, they both result in descriptions which contain useful information, and therefore should not be distinguished with respect to the issues being discussed. Researchers who are more theoretically inclined, and whose work is sometimes referred to as computational vision [17,22,31,64,65,87,88,129], have concluded that the recovery of information about the 3D surfaces in the visible environment is the most (and some have implied only) appropriate goal. In the extreme this leads to systems which are restricted to using only those processes which directly recover 3D information (stereo [14,157], motion [2], shape from shading [65,68,158], shape from texture [71], and in general shape from x, where x is a monocular source of information).

It is our position that even when very good depth information is available, the complexity of the natural world will leave us facing many of the difficult issues that we have been discussing. Consider the problem of interpreting a natural environment, such as a typical crowded city street scene, even if a perfect depth map for all the visible surfaces was available; i.e., in addition to the original color information at each pixel, the exact distance to the corresponding visible surface element at each pixel is also recorded. How should the information be partitioned into meaningful intermediate entities such as surfaces, parts of objects, and objects? And then how could this be interfaced to the knowledge base so that control of the interpretation process is feasible? Given that many initial local hypotheses will be inherently uncertain and unreliable, how is globally consistent and reliable integration of the information achieved?

This, in fact, is exactly the set of problems confronting the reliable segmentation of 2D data into regions and lines. It does not appear that the problem of segmentation can be avoided; in some manner the partitioning of the depth map into surface patches of various types and the extraction of 3D lines via depth and orientation discontinuties must be accomplished. We believe that the principles and approaches presented here for analysis of 2D color data of static monocular images will also be applicable to the processing of 3D depth data derived from stereo and laser ranging

devices, as well as 2D and 3D motion data derived from a sequence of images.

1.6 Overview of the VISIONS System Approach

In response to the issues just outlined, a general, partially working, image understanding system has evolved over the last twelve years. The system design embodies certain assumptions about the process of transforming visual information. The first assumption is that the initial computation proceeds in a bottom-up fashion by extracting information from the image without knowledge of its contents. A second basic assumption is that every stage of processing is inherently unreliable. A third assumption is that local ambiguity and uncertainty in object hypotheses to a great extent can only be removed by satisfying expected relations between scene and object parts that are stored in a knowledge base about the domain.

Figure 1.3 is an abstraction of the multiple levels of representation and processing in the VISIONS system. The successful functioning of the system involves extracting image events which are then used to hypothesize scene and object parts for quick access to knowledge structures (called schemas) which capture the object descriptions and contextual constraints from prototypical scene situations. The hierarchically organized schemas contain interpretation strategies for top-down control of intermediate grouping strategies and allow feedback from high-level hypotheses to low-level processing.

The general strategy by which the VISIONS system operates is to build an intermediate symbolic representation (ISR) of the image data using segmentation processes which initially do not make use of any knowledge of specific objects in the domain. From the intermediate level data, a partial interpretation is constructed by associating an object label with selected groups of the intermediate tokens. The object labels are used to activate portions of the knowledge network related to the hypothesized object. Once activated, the procedural components of the knowledge network direct further grouping, splitting and labelling processes at the intermediate level to construct aggregated and refined intermediate events which are in closer agreement with the stored symbolic object

descriptions. Although the following discussion is motivated primarily via experience with 2D abstractions of the image data (such as regions and lines), the system is being extended to 3D abstractions such as depth arrays and surfaces, as well as to motion attributes of 2D and 3D tokens.

Briefly, the three levels of representation are:

1. Low-Level - Here, numerical arrays of direct sensory data are stored, including the results of algorithms which produce point/pixel data in register with the sensory data (e.g. a depth map produced from stereo point matching).

2. Intermediate-Level - This is referred to as the Intermediate Symbolic Representation (ISR) because symbolic tokens for regions, lines, and surfaces with attribute lists are constructed for the image events that have been extracted from the low-level data. Aggregrate structures produced by grouping, splitting, and/or modifying the primitive image events or other aggregates are also represented symbolically as tokens in the ISR.

3. High-Level - The knowledge base (called Long Term Memory or LTM) consists of a semantic network of schema nodes, each of which has a declarative and procedural component. The network is organized in terms of a compositional hierarchy of PART-OF relations and a subclass hierarchy of IS-A relations. The interpretation process constructs a network in Short-Term Memory or STM which is composed of image-specific instances of portions of the LTM network.

Now let us consider some of the stages of processing in a bit more detail (refer to Figure 1.3).

1. Segmentation

   - Segmentation processes are applied to the numerical arrays of low-level data to form a symbolic representation of regions and lines and their attributes such as color, texture, location, size, shape, orientation, length, etc. The region and line representations are integrated so that spatially related entities in either can be easily accessed [124]. If depth arrays are available, then 3D surfaces and their attributes can be extracted. Two-dimensional motion attributes can also be associated with regions and lines, while 3D motion attributes can be associated with surfaces and their boundaries.

2. Motion and Stereo

   - Multiple frames can be analyzed via motion and stereo algorithms to produce displacement fields, and from these displacement fields depth arrays can be extracted. It may be possible to process these depths arrays to provide initial surface descriptions of portions of the image.

3. Focus-of-Attention

   - Object hypothesis rules are applied to the region, line, and surface tokens in the ISR to rank-order candidate object hypotheses [20,21,118,148,149,154]. This initial mapping of spatially organized tokens to object labels provides an effective focus-of-attention mechanism to initiate semantic processing. The rules can also be viewed as sets of constraints on the range of attribute values on the ISR tokens; e.g. constraints on the color of region tokens, or length of line tokens, or orientation of surface tokens. Relational rules applied to tokens of a different type (e.g. line-to-region relations) allow fusion of information across multiple representations.

4. Object Hypothesis and Verification

   - More complex object-dependent interpretation strategies are represented in a procedural form within the schema knowledge structures [55,106,149,150]. These local control strategies provide top-down control over the interpretation process. Partial interpretations are extended from "islands of reliability" as in the HEARSAY paradigm [41,80]. General knowledge of objects and scenes is organized around a network of standard 2D views of 3D objects. In cases of simple 3D shapes, such as the planar surfaces forming a polyhedral "house" volume, 3D models and associated processing strategies are employed. Verification strategies exploit particular spatial relationships between the hypothesized object and other specific expected object labels or image features.

5. Perceptual Organization

   - Intermediate grouping algorithms, currently being developed [20,21,49,139,140], reorganize the region and line elements into larger aggregrate structures that are expected to more closely match expected object structures. These include line grouping and region merging capabilities and are intended to be applied in either a bottom-up or top-down manner. There are significant advantages in selective application of knowledge-directed perceptual organization mechanisms, and the schemas provide an effective means for specification of the relevant control knowledge. We hope to evolve similar intermediate grouping strategies for complex 3D shape representations in the future.

6. Goal-Oriented Resegmentation

   - Feedback to the lower-level processes for more detailed segmentation can be requested in cases when interpretation of an area fails, when an expected image feature is not found, or when conflicting interpretations occur. It may also be of utility in motion sequences for extraction of particular objects that have been found in previous frames. Both the region and line algorithms have parameters for varying the sensitivity and amount of detail in their segmentation output. The development and control of such strategies, as

well as the integration of their results, is currently being examined in an effort to build a low-level executive [72].

7. Inference

- Due to the inherent ambiguities in both the raw image data and the extracted intermediate representations, object hypotheses will have a high degree of uncertainty. In order to combine this information into a coherent view of the world, two capabilities are being developed: the capability to combine uncertain evidence from multiple sources of knowledge [116], and the ability to propagate confidences (beliefs or probabilities) through the network of schema nodes [85,142]. The latter capability will allow inferential capabilities about the presence of object/scene parts for control of the partially completed interpretation. Both heuristic approaches and theoretical formulations related to the Dempster-Shafer theory of evidential reasoning [38,121] are currently being explored.

2 The LLVS - An Interactive Algorithm Software Development Environment

2.1 Interpretive Control for Applying Image Operators

The Low-Level Visions System (LLVS) is a software environment specifically designed to facilitate all low-level image analysis research for the VISIONS image interpretation project. It is a direct descendent of the Image Operating System which has been in use since 1979 [74], and which was originally written in Fortran 77 and CLisp (a UMass dialect of Lisp which computationally is very efficient). The LLVS consists of a high-level interpretive control language (CommonLISP) with efficient "image operators" written in C. The image operators are viewed as local operators to be applied in parallel, at some level of resolution, to all pixels in an input image. The system is useful as both a research tool for algorithm development and as a production system for initial processing of images; it is currently being further developed and extended.

The user interface is one of the most important aspects of any system. LLVS provides a "friendly" environment for both experienced and naive users through the use of LISP as the high-level control language. LISP is particulary well suited for an experimental environment since it is an interpreted language and can therefore be modified interactively. Use of LISP allows interaction with the system to be conducted using LISP expressions. Images appear as LISP objects. On-line help, interactive verification of parameters, and simple installation of new operators are all provided. The environment supports the dynamic definition of experiments with various images and image operators. LLVS provides control mechanisms (vis LISP) by which the user can compose image operators and control the application of the image operators.

2.2 A Parallel Hierarchical Representation for Image Operators

The computational structure of images in LLVS is based on the concept of a *processing* cone proposed by Hanson and Riseman [51,57]; the processing cone is a model for hierarchical

parallel array processing and is related to other hierarchical structures [127]. At each level of resolution n, the *cone* contains 2^n by 2^n pixels, with a vector of values, V, stored at each pixel. The corresponding vector element V_i for all pixels at a given level of resolution can be considered as a single two-dimensional entity; this slice across the cone is referred to as a *plane*. Each pixel at level n has four unique descendents at level $n+1$ and exactly one ancestor pixel at each level $m < n\ , m \geq 0$.

A *plane* has attributes of *level*, *size*, *location*, *type*, *background-value*, and additional user-definable values. Having a size independent of the level and including a location allows part of a plane to be extracted for independent processing in a very efficient way and then related back to the original plane or other parts of the original plane. The type specifies the format in which the pixels of the plane are stored. The background-value provides a value to be used for neighborhood operations which access non-existent pixels.

Image operators are functions which are evaluated on a set of argument planes and which produce one or more output planes. Execution of an image operator is defined as the local parallel application of a prototype function to all pixels in a plane; the function is applied to a neighborhood around each pixel. This approach will allow the translation to hardware array technology to occur in a natural manner.

A pixel-centered coordinate system and correct system handling of boundary conditions greatly simplify the specification of the prototype image operator, since the operator generally does not need to compute subscripts for neighboring pixels or consider special cases for neighboring pixels which fall outside of the plane. The specification of the image operators is independent of the level of resolution of the data to which it will be applied and it is not necessary to specify the data type of the input plane. Images of different data types and ranges can be handled without any extra code in the image operator, since all access to the image data is through special access functions. Thus, image operators can be interactively applied during the debugging stage to a small window

of the image (by specifying *limits* for the processing), or a coarse representation of the data (by doing the operation at a different *level*), and then later applied to the full image at fine resolution. Image operators can be easily added to the system for any function desired by the user.

Image operators conceptually contain four sections. The two most important are the *process* section, which defines the computation to be performed at each pixel, and the *control* section written in LISP. The control section provides the logical association between the plane names utilized in the process section and the actual planes to which the image operator is being applied, and obtains any other parameters required. [3] The other two sections are the *initialization* and *termination* sections which may not be needed for particular image operators.

2.3 Summary of Distinguishing Features

The three most important characteristics which distinguish LLVS from other image analysis systems are:

1. The partitioning of the image analysis problem into interpretive control and non-interpretive image operators allows good dynamic control of the interactive experimental environment without sacrificing image operator efficiency.
2. The structure imposed on the image operators simplifies the construction of these operators which minimizes the time and cost of implementing them.
3. The human engineering of the user-interface and automatic documentation make the system easy to use for a novice, yet flexible and convenient for an expert.

[3] Because image operators are functions, the input planes and parameters are generally supplied as arguments to the function.

3 SEGMENTATION ALGORITHMS

In an image of a scene there are a variety of sources of low-level information available which can be used to form an initial description of the structure of the image. The extraction of this description, in terms of significant image tokens, is an important precursor to the construction of a more abstract description at the semantic level. It is unlikely that any single descriptive process will produce a description which is adequate for an unambiguous interpretation of the image. Rather, multiple processes are required, each of which views the image data in a different way and each of which produces a partial description which may be incomplete or errorful. As will be seen in later sections, we view one of the functions of the perceptual grouping processes to be the resolution of the multiple descriptions into consistent higher-level symbolic descriptions of the image. Consequently, for the entire history of the VISIONS project, the development of multiple low-level algorithms[4] which are reasonably robust across a variety of task domains has been an important concern, even while recognizing that no single algorithm is sufficient.

It should also be noted that all of these algorithms operate initially in a bottom-up mode without any use of knowledge of the task domain. However, rough "sensitivity" settings on parameters of the algorithm, such as "low", "medium", and "high", can be set a priori in order to extract a larger or smaller number of intermediate tokens; these parameters may also be set or modified by other components of the system and the application of the process may be localized to a specified subimage.

While a variety of low-level algorithms have been developed, there are two primary low-level algorithms that are currently in use in our environment a) a region segmentation algorithm that is based upon analysis of histogram peaks and valleys in local subimages; and b) a straight line

[4]We will sometimes refer to the spectrum of low-level processes for extracting image events as segmentation algorithms, including region and line extraction algorithms, even though some of them do not actually partition in the image into disjoint subsets; thus, sometimes we loosely use "low-level processes" and "segmentation processes" interchangeably.

extraction algorithm that segments the intensity surface into connected subsets of pixels with similar gradient orientation.

In addition to these algorithms there are several others that are used as components of the two main algorithms, or of the perceptual grouping and interpretation mechanisms described later. These will be discussed very briefly in this section; they include an edge-preserving smoothing algorithm [92,105], a histogram-based multi-valued thresholding algorithm [73], a region-merging segmentation algorithm [49], and a low-level executive for directing goal-oriented low-level procesing via feedback from interpretation processes [72].

3.1 Histogram-Based Region Segmentation

The region segmentation technique that we employ was first developed by Nagin [94,95] and later extended by Kohler [75] and Beveridge [24]. The approach is in the spirit of the Ohlander-Price algorithms [103,110], since both their algorithms should be viewed as instances of histogram-based region segmentation. We conjecture that generally our approach will be more robust and computationally efficient because of the problems of recursive decomposition of global histograms in the Ohlander-Price approach [59,112]. The Nagin-Kohler algorithm allows a fixed number of windows to be processed on one pass and significantly reduces the problem of hidden clusters in global histograms that are often encountered in a recursive decomposition approach. However, there has not been a thorough objective comparison of the two approaches.

The histogram-based region segmentation algorithm involves detecting clusters in a feature histogram, associating labels with the clusters, mapping the labels onto the image pixels, and then forming regions of connected pixels with the same label. The process of global histogram labeling causes many errors to occur because global information will not accurately reflect local image events that do not involve large numbers of pixels, but which nevertheless are quite clear. Much of the focus of the algorithms will be to organize the segmentation process around local histograms from local windows and then have a postprocesing stage merge regions that have been arbitrarily split

along the artificial window boundaries.

The Nagin algorithm overcomes the problem of finding small "hidden" clusters by partitioning the image into NxN subimages (usually N = 16 or 32) called sectors; the histogram segmentation algorithm is applied independently to each sector. Thus, each sector receives the full focus of the cluster detection process and many of the problems of cluster overlap and "hidden" clusters are significantly reduced. The cluster extraction algorithm determines local maxima and local minima in the histogram, and then sequentially chooses the next 'best' maxima, which is that maxima for which the ratio of the local maximum to its adjacent valley is largest and the distance to any previously selected local maximum is greater than some minimum distance. Figure 3.1(a) shows the histograms from four adjacent sectors in the roof/chimney area of Figure 1.1(c); note the significant ambiguity in cluster definition that is (usually) present in the histogram. This suggests a natural "sensitivity" parameter in the segmentation process, achieved by controlling the number of clusters detected. The net result is a set of local maxima and their relationships to the valley separating them, each of which can be considered as a distinct cluster and given a unique label. Theoretically, any N-dimensional clustering algorithm could be used (the rest of the algorithm remains the same), but one must keep in mind that computational simplicity is a primary goal since this clustering algorithm will be executed many times across the image and is only one small step in the entire interpretation process. Each of these labels are then mapped back to the subimage and a connected components algorithm is used to generate a set of labeled regions.

The partitioning of the image into sectors has an obvious weakness. If an adjacent sector has a visually distinct region which does not overlap the central sector sufficiently, it is quite possible that the cluster will be undetected in the central sector. The small region representing the intrusion into the central sector will then be lost. Nagin attempted to limit this problem, while still preserving the locality of the histogram, by expanding the histogram domain of each sector to overlap adjacent sectors by 25%, while still maintaining the mapping of labels to the central non-overlapping core.

Nevertheless, if the sensitivity of the peak-extracting process is not sufficiently high, clusters will not be selected and boundaries will disappear. Many of these boundaries will be at corners of a sector window and show results that are inconsistent with an adjacent sector. Figure 3.1(a,b) shows the histograms and resulting individual sector segmentations; the diagonal house wall/boundary disappears.

The Kohler algorithm improves the clustering step by determining correspondences between peaks in adjacent sectors and by adding candidate peaks from surrounding sectors to the set of peaks selected for the central sector. Thus, small variations between peak labels will be accounted for, and if the 25% overlap is not sufficient, small intrusions of regions from one sector to the next will more likely result in a peak to be added. This can be iterated until there are no peak additions between adjacent windows. The augmented set of peaks forms the input to the labeling process. Figures 3.1(b) and 3.1(d) show the set of peaks for four sectors before and after peak addition; Figures 3.1(c) and 3.1(e) show the related individual sector segmentations after mapping cluster labels to pixels and performing a connected components region labeling. Figure 3.1(f) shows the subimage and three steps of the peak addition process in extracting the house boundary. Note that in this example the cluster extraction process was still not set at a sensitivity sufficient to extract the left diagonal boundary between house wall and sky.

The artificial sector boundaries are removed by a region merging algorithm which considers adjacent regions across sector boundaries. The merge/no merge decision mechanism compares the global mean and variance of the two regions, as well as the local mean and variance of the regions near the sector boundary. More recently this remerging algorithm has been generalized [49] to use a set of modular feature statistics so that many different remerging strategies can be used; it can also be applied separately from the region segmentation algorithm under discussion here.

The algorithm can be summarized in five steps:

1. Subdivide the image into sectors, compute the histogram of the expanded sectors, and select the cluster labels in each expanded sector.

2. Analyze the cluster labels from adjacent sectors and augment the label set where appropriate.
3. Segment each sector using the expanded set of labels by mapping each pixel value to a label and applying a connected components algorithm.
4. Remove sector boundaries by merging similar regions.
5. Perform small region suppression, which often reduces the number of regions by a factor of 4.

Figure 3.2 shows additional segmentations using this algorithm on several of the images in Figure 1.1. It should be noted that when multispectral data is available (e.g. RGB) the algorithm can be applied to each band. Figure 3.2(a,b) shows the result of applying our algorithm to the intensity and red band, respectively. Each of the spectral bands might extract additional boundaries, and the intersection of the region segmentations (or equivalently the union of the boundaries) will usually retain more information than the intensity segmentation at the risk of greatly overfragmenting an image, as shown in Figure 3.2(c-e). The reader should refer to Section 3.3.3 and Figure 3.5 for a technique that will deal with the overfragmentation problem.

3.2 Extracting Straight Lines and Line-Based Texture Features

Despite the large amount of research appearing in the literature, effective extraction of linear boundaries has remained a difficult problem in many image domains. The technique presented here [33,34] was motivated by a need for a straight line extraction method which can find straight lines, possibly long and possibly of very low contrast, in reasonably complex images.

A key characteristic of the approach that distinguishes it from most previous work is the global organization of the intensity surface into a "supporting edge context" prior to making any decisions about the relevance of local intensity changes. Pixels are grouped into edge support regions of similar gradient orientation, and then the associated intensity surface is used to determine a gradient-magnitude-weighted planar fit from which a representative line is extracted. A set of line attributes is extracted from the line-support region, the weighted planar fit, and the representative line. These attributes can then be filtered for a variety of purposes.

Figure 3.3(a) is a surface plot of a 32x32 intensity image extracted from Figure 1.1(c); it is the upper roof corner on the right hand side of the image. The vector field of Figure 3.3(b) shows the corresponding gradient image (where the length of the vector encodes gradient magnitude) with a superimposed region segmentation from the gradient orientation algorithm described below. Small 2x2 edge masks are used to compute the gradient so that closely spaced parallel lines will not be missed.

An extremely simple and computationally efficient process is employed to group the local gradients into regions on the basis of the orientation estimates. The 360 degree range of gradient orientation is arbitrarily partitioned into a small set of regular intervals, say eight 45 degree partitions, or sixteen 22.5 degree partitions. Pixels participating in the edge-support context of a straight line will tend to be in the same edge-orientation partitions and adjacent pixels that are not part of a straight line will tend to have different orientations. A simple connected components algorithm can be used to form distinct region labels for groups of adjacent pixels with the same orientation label (Figure 3.3(b)). The great degree of fragmentation of very low gradient magnitude areas into many small regions can be considered evidence for homogeneity and adjacent elements could be grouped into homogeneous regions.

A line whose local edges have an edge orientation in the vicinity of a partition boundary can be expected to be fragmented. To make the fixed partition technique more sensitive to edges of any orientation, the algorithm actually uses two overlapping sets of partitions, with one set rotated a half-partition interval. Thus, if a 45 degree partition starting at 0 degrees is used, then a second set of 45 degree partitions starting at 22.5 is also used. The critical problem of this approach is merging the two representations in such a way that a single representation of each line is extracted from the two alternate line representations. The following scheme is used to select such regions for each line: first the lengths of the lines are determined for each line support region; then, since each pixel is a member of exactly two regions (one in each gradient orientation segmentation), each pixel

votes for the longest interpretation; finally the percentage of voting pixels within each line-support region is the "support" of that region. Typically the regions selected are those that have support greater than 50%.

The underlying intensity surface of each line-support region is assumed to have a meaningful straight line associated with it. In order to extract a representative straight line, a plane is fit to the intensity surface of the pixels in each edge-support region, using a least-squares method similar to that used by Haralick in his slope-facet model. The pixels associated with the longest line support region are shown as dots on the surface plot of Figure 3.3(c). The pixels are weighted by local gradient magnitude so that pixels in rapidly changing portions of the intensity surface dominate the fit.

An obvious constraint on the orientation of the line is that it be perpendicular to the gradient of the fitted plane. A simple approach for locating the line along the projection of the gradient is to intersect the fitted plane with a horizontal plane representing the average intensity of the region weighted by local gradient magnitude. The straight line resulting from the intersection of the two planes is shown in Figure 3.3(d).

The line-support region and the planar fit of the associated intensity surface provides the basic information necessary to quantify a variety of attributes beyond the basic orientation and position parameters. Length, width, contrast, and straightness can be easily extracted. Based upon these line attributes, the large set of lines can be filtered to extract a set with specific charateristics such as short texture-edges, or to select a "working set" of long lines at different levels of sensitivity.

The algorithm described in the preceding sections was applied to two of the house images and two of the road scenes of Figure 1.1. The algorithm is reasonably robust and accurately extracts many low-contrast long lines when overlapping partitions of 45 degrees, staggered by 22.5 degrees, are employed. The results are shown in Figure 3.4, where Figure 3.4(a) shows the unfiltered output of the algorithm applied to the first house image. Note that all of the short and low contrast edges

are still present; thus, it is necessary to filter the very low magnitude lines and group their support regions into "homogeneous" regions that are interpreted as absence of lines, as shown in Figure 3.4(b). We also show the result of filtering on the basis of weighted gradient magnitude (change in gray-levels per pixel) followed by a filtering on length that separates the edges into two disjoint sets, one corresponding to short texture edges as shown in Figure 3.4(c), and the other to longer lines related to the surface structure of objects in the image as shown in Figure 3.4(d). Figure 3.4(e) shows the result of filtering on length alone (length ≥ 5) for one of the other house scenes. The two road scenes are shown in Figure 3.4(f,g).

3.3 Additional Low-Level Algorithms

3.3.1 Smoothing

Many segmentation algorithms or low-level processes use some form of smoothing operation as a pre-processing stage. Image data are complicated by errors of approximation due to the discrete nature of the representation, noise intrinsic to the sensors, and variations in the scene itself. Most simple smoothing operators do not remove noise without destroying some of the fine structure in the underlying image.

A technique by Nagao that has been used to counteract this problem is commonly referred to as "edge-preserving" smoothing [92]. The goal is to average the value associated with a pixel only with those pixels in a local neighborhood that do not cross a high contrast boundary, thereby avoiding blurring of information at the boundary. At each pixel a set of masks, each of which contains the pixel but is oriented from the pixel in a different direction, is used to compute the variance in that window. The pixels in the lowest variance mask are used to compute the new pixel average. The major disadvantage is the expensive computation at each pixel.

We have employed both the Nagao algorithm as well as an alternative developed by Overton and Weymouth [105]. The latter is a direct weighted average of pixels within a fixed shape mask.

An iterative updating process incorporates pixel values in the neighborhood into the calculation of the new average value by means of weights which are decreasing functions of both spatial separation and intensity difference. Values that are more similar (i.e. less likely to cross an edge) are weighted more, and the degree of weighting of the neighborhood versus the current value is increased as a function of the variance in the neighborhood.

This algorithm performs approximately as well as the averaging by the minimum variance mask. Its advantage is that it can be directly computed. Its disadvantage is that the averaging of each pixel in the window is an exponential function of distance, intensity difference, and variance. Thus, the computational cost is also approximately the same. However, it appears that linear approximations to this function might work as well at only a fraction of the cost. The effectiveness of the linear averaging process is currently being examined [35].

3.3.2 Thresholding

Simple thresholding algorithms typically partition an image by assigning one label to pixels with feature values which are equal or greater than some threshold T, and another label to pixels with feature values less than T[141]. For some images, such as chromosome images or hand printed characters, where a clear figure-ground relationship exists, a single threshold often will be able to detect all or most of the object boundaries at the object-background discontinuity. However, those boundaries which correspond to object-object discontinuities or internal structure of the object may not be detected by that threshold. Furthermore, in more complex images which do not exhibit a clear foreground-background distinction (such as images of natural outdoor scenes), a single threshold cannot be expected to detect all or even most of the object boundaries in the scene.

Kohler [73] has developed a segmentation algorithm based upon a complex form of thresholding which uses n thresholds rather than a single threshold. These n thresholds are used to partition the pixel values into $n + 1$ possible classes and a connected components algorithm is used to produce unique region identifiers. The thresholds are selected sequentially and each next threshold is selected

to maximize the global average contrast of edges detected by the threshold across the image. To obtain the final segmentation, an intersection process combines the set of binary segmentations by simply overlaying them and defining a new region label for each distinct combination of the n labels in the set of binary images.

3.3.3 Rule-Based Region Merging

It became apparent during experimentation with the Nagin/Kohler histogram-based region segmentation algorithm that the best results were obtained by making the cluster selection very liberal (i.e. selecting many clusters in order to overfragment the image), and then merging most of the regions in the postprocessing stage. Consequently, the merging process was extracted to form an independent system. The result of that development effort led to a rule-based merging algorithm, which is used to merge regions along the artificial sector boundaries the Nagin/Kohler algorithm, but which can also be used with any algorithm which produces an overly fragmented region segmentation [49].

The rule-based region merging algorithm starts with an initial segmentation as input and selects pairs of adjacent regions which are candidates for merging into a single region. For example, the initial segmentation can be produced by a region growing algorithm with an extremely conservative threshold which produces a highly fragmented segmentation. Because the system only merges (as opposed to splitting) regions, the initial segmentation must contain all the region boundaries desired in the final result.

The merging phase of the algorithm is a locally iterative, but globally parallel, selection and testing of region pairs for merging. For each pair of regions the boundary between them is given a similarity score based on a similarity evaluation function. Then a search for all the local minimal boundaries is made. A boundary is considered locally minimal if and only if it is the lowest scoring boundary for both of its associated regions. Those minimal boundaries below the global threshold for region similarity are removed and the statistics for the newly created regions updated; removal

of each boundary creates one new region from two old regions. The merging processes is applied iteratively and terminates when none of the remaining boundaries score below the global region similarity threshold. Note that many local pairs can be simultaneously processed.

The similarity function, which of course is the key to the effectiveness of the algorithm, takes into account both global and local information. The global features are extracted from the total populations of the pixels in the two regions, while the local features are extracted from the pixels in the neighborhood of the boundary between the two regions. The similarity function is defined as a set of rules, each of which measures the similarity of two regions on the basis of a single feature, such as means of color or intensity, variance of color or intensity, common boundary length, boundary contrast or a combination of features. The results of many such rules are combined through a function which computes the overall similarity measure.

The combination function is the product of the score of each of the similarity rules, each of which returns a value centered on one (implying no information about merging); values less than one indicate a vote for a merge (bounded by zero which guarantees a merge), and values greater than one indicate a vote against a merge. In practice the rules are never absolutely certain, and often return values close to one.

Expectations of the characteristics of the images and goals regarding the desired segmentation can be encoded in the rules, and the nature of the segmentation produced can be controlled by the choice of rules and relative weights on the chosen rules. The philosophy and methodology of rule-based expert systems can be adapted here to allow modular contributions of features in a situation that is theoretically intractable (see discussion in introductory sections). By defining contributions from a feature set where the features can easily be varied for empirical evaluation or can be varied in situations where the goals of segmentation change, many of the advantages of expert system construction are realized. Currently, the system includes seven rules that can be applied to a pair of regions to determine the desirability of a proposed merge:

1. Difference of the global means normalized by the sum of standard deviations of the candidate

regions.

2. Region size - small regions are encouraged to merge with larger regions.
3. Degree of adjacency - regions connected by a relatively long boundary are encouraged to merge.
4. Similarity of the standard deviations.
5. Difference of local means.
6. Maximum allowable standard deviation of the merged region.
7. The degree to which the region could have been caused by mixed pixels during the digitization process (i.e. a narrow region between regions of locally lower and higher means).

Figure 3.5 shows the results of applying the merging algorithm to the output of the histogram-based region segmentation algorithm (described in Section 3.1.) set at very high sensitivity in order to intentionally overfragment the image. In Figure 3.5(a) the algorithm is applied to the overfragmented three-color segmentation of Figure 3.2(e). This result should be compared to both Figure 3.2(c) and 3.2(e) since it produces a qualitatively better segmentation with a smaller number of regions than either. A second road scene example is shown in Figure 3.5(b-d), with the intensity segmentation (derived using a high sensitivity setting) shown in Figure 3.5(b), the additionally fragmented segmentation produced by intersecting regions from the individual R,G, and B segmentations shown in Figure 3.5(c), and the result after rule-based remerging of the RGB segmentation shown in Figure 3.5(d). In the final result one can see important detail that is extracted which was not in the intensity segmentation; the remerging algorithm was successful in reducing the number of regions significantly while preserving important detail.

It should be noted that this approach differs significantly from the approach taken by Nazif [99,100], although both systems are rule-based. Nazif's system is more general in that it allows both splitting and merging, and utilizes both lines and regions. However, Nazif's rules are of the form "If condition, Then action"; it is our position that it is not computationally reasonable to use a production rule paradigm as a full segmentation processs. Our rules are all applied simultaneously

to any decision to merge a pair of regions, but require an external control structure to decide globally which merge they execute.

3.3.4 Segmentation of Surfaces

Recently there has been some work on the process of segmenting depth data into surfaces [2,14,23,45,50,78,91,130,153,157,158]. The problems in segmenting depth arrays are very similar to those encountered when segmenting intensity arrays if the scenes are unconstrained natural environments; almost all the issues in processing of intensity and color data will be present. We believe that some of the algorithms that have proven to be robust on intensity and/or color data may be adapted to the problem of segmenting depth data. This is a topic that we intend to explore in the future.

3.4 Low-Level Executive

The varying goals, features, and algorithms and their parameters makes the problem of delivering the "best" intermediate representation to the interpretation processes an intractable problem. The "best" choices will almost certainly vary across images and vary across different locations in a single image. Consequently, it is unreasonable to expect that an optimal, good, or complete set of image events will be extracted. Rather, the goal is to deliver a usable representation.

We are in the process of developing the feedback loops from interpretation processes to a low-level executive which has knowledge about the set of low-level algorithms previously described. The low-level executive will also be able to accept goal-oriented requests for re-analyzing the sensory data in particular locations in order to extract tokens with particular characteristics (see Section 6.4). If successful, this capability will change the analysis of the sensory data in a fundamental way.

4 INTERMEDIATE SYMBOLIC REPRESENTATION

4.1 Size of Intermediate Symoblic Representation

It is generally accepted that a computer vision system must perform a variety of transformations of the data during the interpretation process. One of the key abstractions is the transformation of pixels, or more generally arrays of sensory data, into named image events which are more abstract than pixels, but less abstract than concepts of objects such as 'house' or 'road'. The representational level at which the named image events are accessible as tokens is referred to as "the intermediate symbolic representation" (ISR). Examples of symbolic events represented in the ISR are regions, lines, and surfaces and their properties [54,118]; corners and textured areas are others that might prove useful, and in general any event extractable from the sensory data may be represented.

In many cases, region and line extraction processes produce large numbers of image tokens. Many of these elements have to be repeatedly accessed, as demonstrated in the line grouping algorithm discussed in the next section and further examples in subsequent sections. The size of the intermediate data base can become enormous if the problem domain encompasses multiple sensor modalities, multiple feature representations, and multiple segmentation algorithms with varying parameter settings, motion sequences, temporally or spatially varying image sets, etc. This, coupled with the multiple grouping hypotheses that each primitive element could take part in, implies that the amount of data which must be maintained at the intermediate level often will be rather significant.

As we shall see in Section 6, the high level interpretation processes in the VISIONS system are implemented as schemas using a blackboard system as the communication protocol. Rather than entering all of the intermediate symbolic elements on the blackboard VISIONS stores this information in an image feature database [124] with a uniform set of access functions forming the interface between the intermediate and high-level processes.

4.2 Organizing the ISR into a Database

Image events stored in the ISR generally have the common property that they are related to some subset of pixels in 2D arrays of sensory data, either directly sensed data such as light intensity or depth from a range sensor, or derived data like normalized color or depth from motion and/or stereo. Thus, the ISR is first organized around "tokens" defined first as sets of pixels from 2D arrays, and then as relatinal groupings of other tokesn (described in Section 5). A set of tokens of a single type that is selected from an image by some process is called a tokenset.

Each token has a type and a descriptor for the associated image event of that given type corresponding image event. For example, a line token will have a decriptor that includes the end points, angle, length, contrast, and straightness of the line. The specification of the descriptor for a type of token in the ISR is called a lexicon. There is exactly one lexicon associated with each token type in the ISR, and of course there will usually be many tokensets associated with each lexicon of a given type.

The representation of a token consists of a name, a referent to the data it describes, and a list of attributes. Each image is given a unique name, as is each tokenset, and each token in a tokenset is associated with a pointer to the actual token. A token name is made up of its image name, its tokenset name and its token index. Associated with each token is a specification of which pixels the token is associated with, represented as a bitplane in the coordinate system of the image. There is also a slot for each of the attributes defined in its lexicon. Tokens are similar to frames in the sense that their slots have procedural attachments for computing the value of the slot. Thus, the definition of an attribute in a lexicon specifies not only the name and data type of the attribute, but a method of computing the value of the attribute. Consequently token attributes need not be computed until they are required, and this permits flexibility in developing computational strategies.

Some of the grouping processes to be described shortly do not examine the image data directly,

but form tokens from groups of other tokens. For example, two perpendicular pairs of parallel lines might be grouped to form a quadrilateral. This more abstract token is still given a name and a list of attributes, and its bitplane can be formed by either the union of the bitplanes of the component lines, or the area bounded by the component lines. Tokens can also be formed to represent events which have no referent in the image, such as occluded boundaries or incomplete geometric shapes. Thus, as with the other levels of representation, there are multiple levels of abstraction included within the general intermediate level representation.

4.3 Accessing Tokens

Tokens in the ISR can be accessed either by name or by attribute. That is, one can access the value of an attribute of a particular token (e.g. what is the average intensity of REGION-27?), or access all the tokens with a particular set of attributes (e.g. which lines are longer than 20 pixels?). Some of the accessing modes embody the capabilities of the associative processing that is a key feature of our image understanding architecture (see Section 9). There is special support for the efficient access of spatially related tokens, implemented by coarsely quantitizing $x - y$ space into sectors to allow selective search for tokens "near" some location. This is isomorphic to the spatial organization of the middle level of our image understanding architecture.

Our approach differs from a straight dataflow approach in which data is restricted to moving up the hierarchy of abstraction. Grouping processes are allowed to incrementally organize tokens under the direction of schema control programs according to the current state and overall goals of the system. This capability is vital if the combinatoric explosion caused by imprudent application of expensive relational measures and grouping operators is to be avoided. For example, a rectilinear line grouping activity which would be appropriate for forming a detailed description of a window frame would be likely to involve an enormous amount of wasted work if applied to an area of coarse tree texture. In our approach, initial results from the application of inexpensive, global processes are examined by specific object interpretation schemas which will selectively apply processes to

particular areas of an image. Thus, a schema with knowledge of houses directs productive line grouping in the area of the window, whereas a schema with knowledge of foliage would invoke different processes for grouping inside the random tree texture that are more likely to be productive than rectilinear structures. This is discussed in more detail in Section 5 and 6.

The ISR provides uniform access functions to different kinds of tokens, and to the degree it makes sense, uniform measurement of the relations between different types of tokens. There are system functions in the ISR for efficiently performing set operations on the bitmaps of tokens. These can be used to compute relations like percent of a line contained in a region and the overlap between a region and the projection of a surface (see Section 5). These relations can be stored directly via pointers, or could be maintained as attributes of tokens (e.g. a region has a list of related lines as an attribute). However, caution must be exercised, because the number of computable relations grows exponentially with the number of tokens. Again, advantage is taken of the fact that relatively few of these relations are "interesting"; Thus, relations often will only be selectively computed under top-down control.

5 FOCUS-OF-ATTENTION, PERCEPTUAL GROUPING, AND INFORMATION FUSION

The mechanisms by which the VISIONS system deals with focus-of-attention, perceptual grouping, and information fusion are related. The attributes of individual tokens, and the relations betwen pairs or sets of tokens, in the ISR are the basis for focussing the system's attention. However, this same information is at the heart of both the grouping mechanisms and the methods for fusing information across multiple representations. The relation between a pair of lines can be used to determine whether they can be viewed as an example of a geometric entity, while the presence of a pair of vertical lines properly bounding a dark brown narrow region would increase confidence in a particular object hypothesis beyond the information in a single token.

Some of the experimental results that will be presented in Section 5 and 6 will be based upon the three region segmentations shown in Figure 5.1. They were derived from the histogram-based region segmentation algorithm of Section 3.1, followed by the rule-based region merging algorithm of Section 3.3.3.

5.1 The Need for Focus of Attention Mechanisms

An important problem in computer vision is the appropriate application of world constraints and expectations; knowledge must be represented in the right form and applied at the right time. This paper has repeatedly stressed that although the processing of a new image must start bottom-up, more restrictive contexts may be used to selectively focus relevant knowledge as information is accumulated and assessed. In this section we describe a simple rule-based focus of attention mechanism for generating an initial set of object hypotheses about the conceptual content of the image [60,61,117,118,148,149,154]. The most reliable of these hypotheses are used to form a kernel interpretation, which can be used to activate the relevant schemas in the long-term knowledge base. Once this has been accomplished, object-dependent strategies associated with the schemas

are responsible for verifying and extending the kernel.

The object hypothesis system provides the first link between the image data and the knowledge structures. In many cases, it is possible to construct rules, defined over the tokens in the ISR, which provide evidence for and against the semantic concepts representing the domain knowledge. While no single rule is totally reliable, the combined evidence from many such rules should be more effective in implying the correct local interpretation. The 'islands of reliability' thus generated can then be used to extend the hypotheses to surrounding image structures which are spatially and semantically related.

The goal is to focus attention upon particular image events that are likely candidates for particular object labels, rather than the selection of the best object label for each region, line, and/or surface. For example, given a set of regions in an outdoor scene we might wish to select a few strong possibilities for the object 'sky'. This approach avoids some of the problems inherent in the traditional approaches to pattern classification where one attempts to classify every region [61]. The application of the hypothesis generation rules can also be utilized top-down as part of a hypothesis verification or extension strategy associated with some active schema.

5.2 Rules Applied to a Single Token: Token Attribute Rules

A constraint on a token feature (or attribute) provides a very simple representation of a unit of knowledge that can be used to generate object hypotheses. For many objects (e.g. sky or tree), there is a natural semantics and a descriptive simplicity that can be associated with a set of token attributes; for example, values may be defined as 'large' (size), 'lightly textured' (texture), 'blue' (color), 'bright' (intensity), etc. A range on each token attribute, even if it is loosely constrained, captures some aspect of the object semantics. Thus, we define a simple rule as a constraint on the range of a token attribute and combine several simple rules into a complex rule for a specific object class. Complex rules are defined as hierarchical combinations of simple rules and may be viewed as defining a volume in a multi-dimensional feature space which represents the set of joint constraints

on the feature set.

5.2.1 Simple Token-Attribute Rules

The degree to which the attribute value of a token satisfies the constraints of a simple rule can be translated into a confidence that the token is associated with an instance of the object. The output of a simple rule can then be viewed as a vote for the assertion that the token represents the object (or more generally a concept). This approach was first applied to regions [61,118,149,154] where the region attributes include color, texture, shape, size, image location, and relative location to other objects. More recently [20,21] the approach has been extended to line tokens, where the features include length, orientation, contrast, width, etc.

A simple rule is defined as a piecewise-linear mapping function on a feature. It consists of a central positive voting range (where the response of the rule is 1), surrounded by zero voting ranges, surrounded in turn by veto ranges. The rule response is linearly ramped from values of one to zero on both sides of the central positive range. The veto ranges effectively eliminate the consideration of those tokens with unacceptable attribute values as an instance of the object class; for example, a region can be removed as a "sky" candidate if it is green or at the bottom of the image.

Simple rules are parameterized by six thresholds corresponding to the endpoints of the ranges and typically are formed by examining the relationship of the histogram of an attribute across all tokens in multiple images to the histogram of that attribute extracted from tokens associated with a single object class (the 'class-conditional' histogram; see Figure 5.2). For example, the color or texture of grass regions would be compared to the color and texture of all regions. Using the six values, non-symmetric rules may be formed; the simple rule example shown in Figure 5.3(a) is based on an 'excess-green' opponent color feature (2G-R-B) of regions and is part of a complex rule (see below) for hypothesizing 'grass'. Figure 5.3(b) shows the 'grass' object label hypotheses from this simple rule for one of the standard images; in this figure, brightness encodes the rule response (with a response of 1 as brightest). Note that black regions either evaluated to 0 or were vetoed.

5.2.2 Complex Token-Attribute Rules

A complex rule is a hierarchical collection of other rules, which may be either simple or complex. A typical structure is illustrated in Figure 5.3(c), which shows the complex rule structure for 'grass'. The top-level rule is organized as a set of five other complex rules (one for color, texture, size, shape, and location), each of which is composed of one or more simple rules. The response of a rule at one level in the hierarchy is determined by mapping the responses from lower level rules through a combination function, in this case a simple weighted average (the weights are shown on the arcs in the figure). Our intention is that rules be weighted with a small integer value (3 to 5 levels) representing strong, medium, and weak contributions. The rules should be robust enough so that a fairly rough specification will suffice without the necessity for "parameter twiddling". Many other rule structures and combination functions are possible. Figure 5.4 shows typical results from the application of the complex rules for 'grass', 'sky', and 'foliage'.

It is also worth mentioning here that the token attribute rules (and the relational rules introduced in the next section) can also be used for filtering and grouping. A rule might be applied to a token set to retain only those tokens that have certain properties. These rules can be applied sequentially, so that the subset of tokens remaining after application of the rule can become the focus of another simple or complex rule. Rules used in this manner provide a form of control where subsets of tokens are narrowed (or filtered) via a sequence of rule applications. In later sections, we will show how these techniques can also be used to form composite tokens which are aggregates of various token types, thus achieving grouping and fusion capabilties at the same time.

5.2.3 Interactive Environment and Alternative Rule Forms

Knowledge engineering of rules can be greatly facilitated by an interactive environment for rule construction. A user can get an immediate sense of the effectiveness of proposed rules by displaying the rating of each symbolic candidate in intensity or color. Thus, rule development becomes a

dynamic process with a natural display medium for user feedback.

An interactive environment for generating and testing rules has been developed. A user may create a rule, display its results on a set of test images, edit the rule, and construct one or more complex rules which incorporate it. When look-up tables on an image display system are available, the user can interactively change the piecewise-linear function that is loaded into the look-up table. By encoding all of the pixels of each image token (e.g. pixels of a region or line) with its value of a particular attribute (e.g. intensity, length, etc), the look-up table can immediately display the rule vote. Thus, the user can change the rule and see the results in real time (i.e. at video rate).

In addition, a simple language interface has been constructed so that rules can be specified on any feature in terms of five intervals of the dynamic range of a feature - "very low", "low", "medium", "high", "very high" [54,61]. These labels induce a partition on the range of the feature, and for each interval the user specifies whether the rule response is "ON", "OFF", or "VETO". Figure 5.4(c) was obtained using a rule of this type.

There have been several different rule forms employed in our research environment. Rather than the six-point central positive range for the mapping function, any piecewise linear function with multiple veto and positive voting ranges might be used; for example, extracting an orientation image of a line around the origin of the orientation scale will require positive voting ranges at each end of the scale (since the scale is circular).

An alternative rule form has developed from an initial version of an automatic rule development subsystem that has been motivated via a Bayesian viewpoint. The rule structure can be viewed as a discrete approximation to a continuous distribution. It is based upon the ratio of histograms of a token attribute for a particular object to that of all objects, sampled from a labelled set of images. For a single feature and meaningful statistics this is an optimum maximum likelihood decision rule [61]. Thus, if a sufficient range of object examples are available to meaningfully represent an object, automatic development of rules should be possible.

Finally, some users have explored different variations of combination functions with simple rules. The version we have presented involves a weighted sum of the output of simple rules whose values range from 0 to 1 (with a veto range). Some experiments in our group have utilized a multiplicative form ranging from 0 (a veto) to some number greater than 1, with 1 as a neutral value.

5.3 The Need for Perceptual Grouping Mechanisms and Information Fusion

We have argued in preceding sections that low-level algorithms, which initially do not use any knowledge of the scene or its characteristics, should never be expected to produce symbolic descriptions of image events that are a close match to the object descriptions in the knowledge base. Consequently, there must be additional mechanisms for dealing with the problems of an incomplete and partially incorrect intermediate symbolic representation. In the following subsections we describe both data-driven (bottom-up) and knowledge-based (top-down) processing techniques which can build an intermediate representation with tokens that better match the primitive elements of the object descriptions in the knowledge base. Thus, the emphasis here is on the reorganization of the intermediate representation into new, more abstract collections of tokens.

5.3.1 Perceptual Grouping and the Reorganization of Intermediate Level Data

There has been surge of interest in data-directed grouping phenomena in computer vision. One general strategy is to form collections of tokens by detecting relations between tokens which are likely to be non-accidental (e.g. see [151,156]). The relation represents a grouping criterion for organizing the set of tokens to produce a new token. These ideas are much less vague if particular grouping strategies are considered, such as linking fragmented co-linear lines with nearby endpoints, grouping sets of parallel and perpendicular lines that have some spatial coherence, merging similar short lines that form a textured area, and grouping regions (possibly non-adjacent) which have similar characterisitcs of color, texture, size, shape, and/or location.

Grouping algorithms must, of course, contend with the combinatorics of the large number of

image tokens (such as the 2,000 to 10,000 lines and the 200-500 regions for some of the algorithms described in the previous sections), the many different attributes of each token and the many possible relationships between tokens. Thus, there must be some technique to focus attention on areas where the grouping processes will most likely be useful. One technique for data-directed grouping is to locally histogram (i.e. via subimages) various attributes of a given token, and look for interesting events such as many parallel lines, or orthgonal lines, or many small textured regions in certain areas of the image. Certain properties, such as orthogonal line groupings, might be of sufficient universal interest that they are always extracted. In each of the cases cited, the strategies can be applied without the application of knowledge, but the computational cost might be rather severe.

Knowledge-driven application of perceptual grouping mechanisms is the obvious alternative to the data-driven control just described. The grouping mechanisms themselves can remain substantially the same; only their invocation need be modified. Thus, specific grouping strategies would be applied to specific areas of the image, often with specific goals. We envision that many grouping strategies will be invoked via the top-down schema processes that will be described in Section 6. It should be obvious that if the system focusses upon a particular region, regions that are similar in color, texture, etc., and that are spatially close (i.e. within a certain distance of each other) can be selected. Thus, if tree foliage can be expected to be highly fragmented and have dark homogeneous shadow regions adjacent to lighter textured green regions, these characterisitcs can be used to group such regions and form a new region token with very different properties than the individual tokens in the set. If short texture lines were extracted, and a connected components algorithm was applied to their support regions (section 3.2), then textured regions would be extracted rather simply.

In Section 5.5 we will outline new algorithms, as well as specify how some of the algorithms and techniques already described can be used to fill the needs of perceptual organization strategies. These discussions sometimes are fairly brief, because the work is recent or has been described

elsewhere in this paper. Note also that the goals and techniques that will be described heavily overlap those of information fusion that are outlined in Section 5.4.

5.3.2 Information Fusion

A major problem confronting vision systems which use multiple sensors or which generate multiple low-level descriptions from image data is the coherent and consistent integration of information contained in the multiple representations. It has also become evident that each low-level process extracts only partial descriptions, and that there is a great deal of redundancy among these descriptions which can be profitably exploited. Thus, maximum reliability can only be achieved through processes that integrate information represented in widely varying forms. For the reasons discussed earlier in this paper, we take the view that information fusion can most effectively be accomplished during the interpretation process, rather than at the time that the tokens are first extracted (e.g. by directly integrating region and line algorithms). This approach bears similarity to the philosophy of the CMU approach to sensor fusing in their development of an autonomous vehicle [120]. One approach to fusion will be illustrated here by extending the rule-based hypothesis system to operate over multiple token types, in this case the regions and lines extracted from the image data [20,21]. The techniques could be extended to include fusion of information from lines, regions, corners, surfaces, volumes, and generally any other token abstracted from the sensory data.

5.4 Relational Rules Applied to Multiple Tokens

Up to this point, only rules based on properties associated with a single token (e.g. a region) have been considered; these may be viewed as unary rules, defining a unary relationship between the token and the hypothesis. By defining binary relationships between token pairs, an approach to the problems of both grouping and information fusion is obtained.

It is important to understand that the token attribute rules are not directly classifying or labeling regions, but rather allowing the system to focus attention. They provide a rank ordering

of tokens as candidates for object hypotheses. This rank ordering can be used in various ways to associate object class hypotheses with the token. Note that some strategies might involve the application of a sequence of different types of rules to a tokenset, each time filtering the tokenset so that less likely candidates are eliminated. For example, a single exemplar region for tree crown might be selected via a complex rule for foliage. Then a distance relational rule could be used to select regions whose centroids or boundaries are nearby (i.e. within some maximum distance). Finally, for those remaining regions a similarity relational rule on brightness and color might further filter the possible tree regions into a reasonably sized set of candidates.

The rule system that has been developed can be extended in a very simple way to compare and aggregate tokens of the same type. A single attribute of any pair of scalar-valued tokens can be compared by simply taking the difference in their values. (Note that for some non-scalar attributes such as orientation, which has a circular scale, a somewhat more complex orientation difference must be defined, so that values of 0 degrees and 359 degrees will be evaluated as a difference of 1 degree). The difference of attribute values provides a relational measure of similarity, or attribute difference, between tokens. Given some token, there is now a basis for selecting among all other tokens those that satisfy a set of similarity relations applied to a set of attributes. For example, regions that are of "similar" intensity and "close" in distance can be extracted. Lines which are "parallel" or of "similar" orientation to a given line token can be selected, or those of "orthogonal" orientation with endpoints which are less than some specified distance apart can be extracted.

Of course the basis for the selection of tokens is an appropriate specification of the similarity or comparison relations. Terms such as "similar", "different", "near", "far", "parallel", and "orthogonal" must all have a computational definition. The rule system can be used to implement comparisons of token attributes by defining a simple rule on the difference between the attribute values of all tokens to the attribute value of a given token (or some other value somehow specified by the user or the system). In effect the voting function is a computational measure of the degree

to which a relation has been satisfied, and therefore, in some sense, defines the value of a "fuzzy" relation.

5.4.1 Information Fusion via Relations Across Multiple Token Types

Here, the goal is to extend the rule system to include composite tokens, which are aggregations of tokens across the multiple representation which satisfy a specified set of relations. The key integrating mechanism is the use of relations between tokens of different types. From those aggregate objects satisfying the relation, new features are extracted and used to generate object class hypotheses. To be somewhat more specific, consider the interpretration of a road scene with a view down the road. The formation of a 'road' hypothesis should not be based on any single token type (e.g. regions), but rather on an aggregation of lines, regions, and surfaces that have specific relations to each other and which share road attributes. Ideally, one would like to find a homogeneous region, of the correct relative brightness and color, bounded by 2 converging straight lines, and approximately covered by a horizontal planar surface.

The simultaneous use of both region and line information, for example, permits two types of perceptual grouping operations to take place across representations. On the one hand, a region or set of regions can guide the grouping of lines (an example of this is developed in Section 6) while on the other hand a line or set of lines can guide the grouping of regions. In the following discussion the terms region and line are used to refer either to the tokens obtained from the corresponding segmentation process or to sets of regions and lines already grouped by other processes.

There are many ways to use relations between tokens to form aggregations of tokens. An initial implementation in the VISIONS system utilizes a primary token type (region tokens) through which tokens of other types are selected and filtered via a sequence of simple, complex, and relational rules. Since the region and line tokens stored in the ISR are both pixel-based, intersection turns out to be a convenient mechanism for associating line tokens with a region. Intersection of the support pixels of a line (see Section 3.2) with the pixels in a region allows the definition of various relations

in a straightforward manner. Examples include relations such as INTERSECTING (whose value is either TRUE or FALSE depending upon whether a region and line have a non-empty intersection); and BOUNDING, and INTERIOR, with a numeric value ranging between 0 and 1 representing the degree to which these relations are satisfied [20,21].

5.5 Some Example Grouping Mechanisms

5.5.1 Grouping via Rule-Based Token Attributes and Relations

Many grouping strategies naturally fit into, and are definable with, the tools that have already been presented. Grouping via relations between different token types, such as line and region groupings, can be achieved in a fairly simple yet general manner. The idea is very similar to that described in the previous section for the object-hypothesis rules (which accepted only a single token type as the argument). Features can be measured from the aggregate structure and used to generate object hypotheses. Perhaps more importantly, a set of relational rules can be applied sequentially to filter the initial set of aggregate tokens in order to select sets with particular properties.

The intersection type of relational rule can be used in some very diverse ways. One example is to use INTERSECTION and INTERIOR relational rules to select only interior lines, lines which are sufficently or completely interior to any region, and then to use a complex token attribute rule to rank short, high contrast horizontal lines (Figure 5.5(a)). The line score could then be averaged to form a texture feature on regions. Alternatively, a line density measure can be formed as a region texture measure by counting the occurences of lines which receive a high score from the line attribute rule and then normalizing by the size of the region. Figure 5.5(b) shows this measure mapped onto the regions. The texture measure can be used to hypothesize or verify the roof region which has horizontal shingle texture. Note that grass is the only other region that has any significant horizontal texture rating.

A simple shape measure can be computed by determining if a region is bounded by a pair

of long vertical lines. The filtering rule for this feature selects only those lines which lie on a region boundary and the ranking rule assigns high values to overlapping pairs of long vertical lines. Thus, given a region token, those line tokens which INTERSECT the region are selected; then the BOUNDING relation can be used to retain only those lines that are sufficiently on the boundary of the region; next vertical lines are extracted via a simple rule applied to line orientation; and finally the PARALLEL relation on the resulting lines would yield an aggregate structure of a region and those parallel vertical lines which bound that region. As shown in Figure 5.4(c,d), this rule is useful for extracting telephone poles. As a last example, horizontal and vertical lines that are on region boundaries can be extracted in an effort to find rectangular structures such as the regions and lines forming the shutter-window structures of the house; see Figure 5.5(e,f). Note that windows usually are very difficult to extract due to reflections and transparency. Additional details about the structure of the relational rule system and further experimental results may be found in [20,21].

5.5.2 Grouping Line Tokens Based on Geometric Relationships

Abstract collections of lines can prove to be very useful in the development of object hypotheses. Certainly once an object has been hypothesized, confirmation of an expected relationship between a set of line tokens can be used as further evidence for the correctness of the hypothesis. There are certain 'primary' relations between subsets of straight lines that are sufficiently important that one might be interested in extracting all occurrences prior to attaching semantic significance. A set of parallel lines whose endpoints form straight lines usually will prove to be important (e.g. for geometric structures such as the shutters in a house, telephone poles along a road, etc.). Similarly, perpendicular lines that have a pair of nearby endpoints can form "corner" structures which could be grouped into larger geometric objects such as rectangles. If the orthogonality criterion is dropped and corners of any orientation are extracted, then they could be grouped into parallograms and trapezoids by using a parallel line criterion for the further grouping of corner-line-pairs. If an orthogonal intersecting line grouping was attempted, weak structures, such as the window panes in

windows might be extracted and used as the basis of hypothesizing or verifying windows between shutter rectangles.

When large numbers of tokens are present there are some interesting computational, storage, and retrieval issues that must be considered in the search for meaningful structures. A decision must be made regarding those relations which are explicitly represented between pairs of tokens, since there can be a significant cost associated with finding and storing all token pairs (or sets) that satisfy a given relation such as parallel or "near". Unfocussed grouping of tokens also leads to the problem of generating a large number of uninteresting token sets, which would also defeat the purpose of searching for interesting aggregations.

Within our research group, there is an active research effort by Reynolds and Beveridge [114] on the development of line grouping processes based upon geometric and spatial relations; their initial effort produces rectilinear line groupings using the spatial proximity of lines. Measures for relations between lines have been quantified for spatially proximate orthogonal (SPO), spatially proximate parallel (SPP), and spatially proximate collinear (SPC) lines. We have provided an example of the SPO relation in Figure 5.6(a) that is dependent upon the length, parameterized distance of each line from their intersection, and deviation from orthogonality.

Example line groupings are shown in Figure 5.6(b) as a set of straight lines extracted from the aerial image of Figure 1.1. The three relations can each be applied to four pairs which satisfy each of the three relations. Groupings of various sorts can now be applied. In Figure 5.6(c) a connected components algorithm was used to form line aggregations; a line is considered "connected" to another if any of the three relations are satisfied. The result is a grouping of lines where local proximity is always satisfied, and rectilinear structures of parallel and orthogonal lines are extracted. In order to do this efficiently the lines in the ISR are stored in 10 degree orthogonal line buckets; i.e. the bucket for the 11-20 degree range will also contain lines in the 101-110 degree range. Some of these could be the basis for further processing, such as the formation of rectangles.

Top-down strategies allow the geometric/spatial relationships to be used in a much more directed manner. In aerial images at appropriate resolution one can expect roads to appear as sets of parallel lines as in Figure 5.6(c), and intersections as perpendicular line groupings of roads. One can expect buildings to appear as rectilinear structures and thus one might use weak evidence for the fourth side of a rectangle to complete it as in subgraphs of Figure 5.6(c). In another case the geometry of houses and roofs might imply that a parallelogram should be present, and that shingle texture should appear within the parallelogram. If only 3 sides of a parallelogram were found, a top-down line linking procedure might be invoked, and a texture extraction procedure for short horizontal edges might be applied to the area within the parallelogram (see Figure 5.5(a)). An example of such a knowledge-directed grouping strategy is given in Section 6.3.2 and Figure 6.4.

5.5.3 Hierarchical Grouping of Co-Linear Line Segments

One very important property of images is the presence of long straight lines. Unfortunately, most algorithms detect line fragments, which then must be joined into longer lines. In this section we restrict our attention to the grouping of lines into longer, straight lines based on the general notions of collinearity and proximity [139,140]. The algorithm by Boldt and Weiss presented here could be viewed as a straight-line extraction process, beginning with the output of a standard edge-operator, and terminating with a set of straight lines. Consequently, it can be viewed as an alternative to the Burns line extraction algorithm of section 3.2. However, the relevant point here is the geometric grouping mechanism that is the basis for combining shorter straight lines (in the limit edge elements) into longer straight lines. A straight line can be viewed as a sequence of line segments in which consecutive pairs are roughly collinear and similar in contrast, where each segment is close to its successor, and arranged such that the entire sequence passes a straightness test. Figure 5.7 illustrates the relations that are relevant to grouping lines. These criteria depend on scale; long line fragments, for example, can be separated by a larger gap than small ones and still be close. A sequence of lines which is not straight at one scale can be part of a longer sequence

that passes the same straightness test at a larger scale.

The idea of grouping line segments is not new. Nevatia and Babu [101] developed criteria for grouping edges; however, they only applied them locally and they did not take into account more global context. McKeown et al [89] have also devised rules which are similar to those used by Boldt and Weiss and have applied them to the problem of linking elongated regions. They use the criteria that a) the regions must be close; b) they must not overlap too much; c) they must have similar orientations; and d) the lateral distance must be small. Thus, they implement the rules of collinearity and proximity. A major difference between their approach and the Boldt and Weiss approach however, is that they do not use a hierarchy.

The algorithm, when viewed as a segmentation process, begins with an edge-detection operator (in this case the zero-crossing of a Laplacian), although any operator which produces measurements of the contrast, direction, and location (including start and end points) of the edge could be substituted. The remainder of the algorithm can then be viewed as a line-grouping process which starts with a set of straight lines derived from any process. In fact the grouping process can be applied to the set of lines produced by the Burns algorithm of Section 3.2.

Each grouping cycle consists of two steps: linking and merging. In the linking step pairs of line segments are tentatively connected based on binary relations. In the merging step sequences of linked line segments are examined and possibly replaced by a single line segment. The linking step improves the efficiency of the merging step by significantly reducing the number of line sequences that would be examined by blind search. The computational complexity is reduced by repeated hierarchical grouping of lines into longer lines. Thus, there is an implicit scale-space hierarchy [155] and the search always remains local with respect to the current scale.

The linking step consists of a search for pairs of lines that satisfy the geometric and non-geometric criteria which make them candidates for grouping. The geometric criteria are collinearity and proximity (the end points must be close, but the lines must not overlap too much), and the

non-geometric criterion is similarity of contrast; these criteria (and others) are shown in Figure 5.7(a-e). The same linking process is performed separately for each endpoint of each line, and can obviously be done in parallel. The result is a directed graph with the line segments as its vertices and the links as its arcs; a portion of the link graph is shown in Figure 5.7(f). In general, a line will be linked with several other lines.

The merging process consists of search and replacement. The merging process will examine paths in the link graph and test them for straightness. The amount of geometric context used is determined by a *search radius*, shown superimosed over a portion of the link graph in Figure 5.7(f), which bounds the length of a sequence of lines which is tested for straightness. Each sequence of lines is approximated by a straight line illustrated in Figure 5.7(g), and if the straightest path for a given line passes a straightness threshold, that sequence is replaced by a single straight line. The algorithm examines each line in the link graph at one scale, and the linking and merging process is repeated at each scale, with the linking and replacement radii increasing by a constant factor from one scale to the next.

The algorithm has been applied effectively to natural scenes and aerial images. Figure 5.8(a) shows the initial edge segments which are input to the grouping algorithm; these are obtained by associating unit length edges with selected locations along the Laplacian zero crossing contours. Figure 5.8(b-d) shows the output of the geometric grouping algorithm at each stage of the iterative grouping. Figure 5.8(e) shows the final result on the entire image; unit length lines remaining after completion of the grouping cycles have been removed. Figure 5.8(f) demonstrates the application of a filter on length and contrast to the full set of lines shown in (e). The full line set can be filtered and manipulated in various ways to extract different views of the data, much the way that the output of the Burns' algorithm was. Figure 5.9 shows the output of the line grouping algorithm on several full images.

5.5.4 Region Grouping

The segmentation algorithms presented earlier sometimes produce dramatically unexpected results, and at other times produce expected but fragmented representations. The low-level executive system [72] under construction (see Section 3.4) can coordinate knowledge-based feedback to the segmentation and feature extraction level.

As an example, we briefly note here that the rule-based region merging algorithm of section 3.3.3 can also be used to perform certain types of grouping. Again, it will be crucial to have some amount of knowledge beyond what was available during the initial application of segmentation algorithms. If object-dependent strategies can be developed which anticipate the manner in which the segmentation algorithms fail, then knowlege-based processing can re-invoke the region merging processes with appropriately different parameter settings. We are only beginning to explore strategies to utilize it in a general manner.

6 SCHEMAS AND IMAGE INTERPRETATION

A central problem in image understanding is the representation and appropriate use of all available sources of knowledge during the interpretation process. Each of the many different kinds of knowledge that may be relevant at various points during interpretation imposes different kinds of constraints on the underlying representation. In general, the representation must be sensitive enough to capture subtle differences and variations in object classes, yet be robust enough to capture broadly applicable 'sketches' of objects and expected scenarios.

In some cases, the knowledge is declarative in form (e.g. 'Sky is blue.') and consequently vision systems must have access to knowledge of the attributes of objects and their parts. In other cases the knowledge is relational in form: the decomposition of objects into parts, the relationships between the parts, and the contextual properties of larger collections of objects. Other kinds of knowledge are more procedural in form, such as knowledge of the perspective transformations mapping from 3D to 2D, or the knowledge encoded in a top-down 2-D grouping procedure for extracting tree foliage and shadows. In some form or another, knowledge about individual objects (instances, classes, and descriptions), of events (action, situations, cause and effect), of performance (how-to, skills), and metaknowledge (knowledge about what we know: extent, origins, reliability, etc.) must be encoded in an accessible representation.

In this section we define a general methodology for knowledge-based parallel processing and a development environment to support that work.

6.1 Schemas as a Representation of Knowledge in VISION

In the VISIONS system, scene-independent knowledge is represented in a hierarchical schema structure organized as a semantic network [30] of schema nodes [55,58,106,148,149,150]. Each schema defines a highly structured collection of elements in a scene or object, and encodes the knowledge of how to recognize that object/scene. It should be noted that a related concept of

perceptual and motor schemas has been long advocated by Arbib [9,10]. Our goal is to develop a network of concurrent processes that will cooperatively build an interpretation of the scene. Each object in a scene schema, or part in an object schema, can have an associated schema which further describes it. One can view the schemas as a vertical structuring of knowledge in that the knowledge is organized across representational levels by the object description. Each schema node has both a declarative description appropriate to the level of detail describing the relations between the parts of the schema, and a procedural component describing object recognition techniques expressed as a set of hypothesis and verification processes called interpretation strategies.

Although such a hierarchy provides rich descriptive capabilities, it is also necessary to distinguish between the representation of a specific instance of a visual object (Ed's house) and the general class of objects (a 2-story Victorian) to which it belongs. Thus, a further division of knowledge into long term (LTM) and short term memory (STM) across the levels of the hierarchy is necessary for differentiating the system's permanent a priori knowledge base from the representation derived from the sensory data of a specific image during the interpretation process. Figure 6.1 illustrates the idea that both STM and LTM are a multilevel representation of 2D symbolic tokens of points (or vertices), lines, and regions, and 3D tokens of surfaces and volumes, and then abstract semantic tokens of objects and scenes. A node in LTM represents the general class of an object and may be related to a node in STM by an instance-of arc, which specifies that the node is an instance or member of the general class.

Under the model that views active schemas as generating a set of concurrent communicating processes, another view of the schema system and the manner in which it is intended to relate to the image understanding architecture discussed in Section 8 is illustrated in Figure 6.2. Since schemas are intended to run in parallel wherever possible, a multiprocessor is shown interposed between LTM and STM, with a schema running on each processor (in the limit). STM then becomes a blackboard, with different access functions tuned for efficient retrieval of data from the large

volumes of intermediate tokens, and for the smaller amounts of object and scene hypotheses. LTM stores the schema classes, and the active schemas are responsible for instantiation of additional schemas.

The set of KSs are processes which can be invoked and applied to particular portions of the image and STM under the control of interpretation strategies of the schemas. In particular, the difference between schemas and KSs must be understood. Schemas currently are object-and scene-based, with fairly complex control strategies for recognition. In our system the KSs are processes that are to be applied by interpretation strategies and consequently have relatively simple control strategies. The schemas invoke the KSs and pass parameter information sufficient for the KS to return a result. The KSs vary greatly in form and complexity while the schemas have a uniform representation.

Many of the strategies relating to geometric structures are based on standard or constrained two-dimensional views of an object or scene. The view are not image-specific views but rather projections of expected structure to the image plane for a loosely constrained viewpoint. Only very simple three-dimensional models have been experimented with (however; see [160,161]); Weymouth [149] employed a simple wire-frame model of a house and simple relationships between the ground plane and house wall. Parma [106] also utilized simple 3D reasoning schemes. However, the major results have been achieved using a network of two-dimensional views [149] where each view represents a canonical viewpoint on the Gaussian sphere of all possible viewponts. This canonical viewpoint is representative of a particular sector of the Guassian sphere for which the features of the object (scene) are qualitatively invariant [36]. Thus, in some sense we are compiling three-dimensional information into this network in a form which leads to efficient recognition via associated strategies for matching the expected two-dimensional projections of three-dimensional objects with the image.

6.2 Example Interpretation Strategies

Schemas may be viewed as (possibly nonoverlapping) partitions of the knowledge base. The procedural knowledge embedded within object-specific (or to be more precise, schema-node-specific) interpretation strategies provides local control information. Once activated, the interpretation strategies are responsible for verifying and rejecting hypotheses, assessing their validity, extending hypotheses by selecting and grouping related data and hypotheses, instantiating the schema node, and activating related schemas. The interpretation strategies may be domain and object dependent, or uniform across domains.

In their initial implementation, the interpretation strategies have been encoded primarily in procedural form. However, it is unlikely that any intermediate level process would be useful only for a specific object (whether it be matching of expected object attributes with token attributes, geometric grouping, verifications of spatial relations, etc). As more strategies are implemented and we gain experience with their utility and generality, our goal is to extract some of the components and to represent them as parameterized 'knowledge sources'. These will then become the building blocks from which interpretation strategies are constructed. Ultimately, each schema interpretation strategy could become nothing more than a structured series of calls to the knowledge sources.

The contextual verification and extension of hypotheses via consistency with stored knowledge and expectations leads to a variety of interpretation strategies that may be data-directed (bottom-up strategies) or knowledge-directed (top-down strategies) or both. In the next few sections we will present examples of several different classes of interpretation strategies. The first example is a hypothesis generation and extension strategy which uses the object-hypothesis rule system (Section 5) to select 'exemplar' regions hypothesized to represent an object, and then extends them to similar regions. The second class of strategies uses geometric information to direct the grouping of intermediate events to better match the expected model (in this case a house roof). The last example strategy involves the detection and correction of errors in the interpretation strategy (this

strategy is discussed in the context of the overall interpretation results in Section 6.3).

6.2.1 Exemplar Selection and Extension

Object hypotheses can be obtained by applying the object hypothesis rules (refer to section 5) to the intermediate symbolic representation in order to select object "exemplars" which capture image-specific characteristics of the object. The rank-ordered set of image tokens for each object can be used to select one or more exemplar candidates. The set of exemplars can be viewed as a largely incomplete kernel interpretation.

Exemplar regions can be used in various ways to select similar regions [60,61,118,148]. In many situations another instance of an object or object part can be expected to have a similar color, size, or shape. For those objects for which the spectral characteristics can be expected to be reasonably uniform over the image, the similarity of region color and texture can be used to extend an object label to other regions, perhaps using the expected spatial location and relative spatial information in various ways to restrict the set of candidate regions examined. This strategy assumes that the image-specific variation of a feature of an object is expected to be much less than the inter-image variation of that feature for the same object.

The similarity criteria might also vary as a function of the object, so that regions would be compared to the exemplar in terms of a particular set of features associated with that object. Thus, a sky exemplar region would be restricted to comparisons with regions above the horizon and that look similar (in terms of color and texture) to the largest, brightest region located near the top of the picture. A house wall showing through foliage can be matched to the unoccluded visible portion based upon color similarity and spatial constraints derived from inferences from the house wall geometry.

One possibility is to utilize the object-specific set of simple token features that were associated with the object hypothesis rules. As shown in Section 5, similarity relational rules can be functions of feature differences of similar tokens. It is also easy to use vetos for large token differences or for

spatial constraints to restrict the spatial area over which candidates for exemplar extension will be considered.

The shape and/or size of a region can also be used to detect other instances of multiple objects, as in the case when one shutter of a house has been found [106], or when one tire of a car has been found, or when one car on a road has been found. In many situations another instance can be expected to have a similar size and shape in the physical environment, and geometric constraints can be used to determine expected region sizes and locations. This permits reliable hypotheses to be formed even with high degrees of partial occlusion. The presence of a single shutter of a house provides strong spatial constraints on the location of other shutters. If two shutters are found then perspective distortion can be taken into account when looking for the other shutters, even without a camera model, under an assumption that the tops and bottoms of the set of shutters lie on a straight line on the face of the house.

In Figure 6.3(a) we show the regions forming the highest ranked object hypotheses as the object exemplars. The region features were used via similarity relational rules to measure the difference between the exemplar region and each candidate region. The rule response was converted into a distance metric and bright regions correspond to small differences. Each of the rules contained location and size components which enter into the final distance measurement, but in general, there are more intelligent ways of using these features in the interpretation strategy responsible for grouping regions. Figure 6.3(b) shows the similarity rating of regions obtained using the features of the grass rule.

6.2.2 Knowledge-Based Control of Grouping Processes

In this section we briefly motivate the types of additional top-down strategies that will be necessary for properly reorganizing the intermediate tokens to match the object better. The basic idea will be sketched using an example from Weymouth's thesis [149]: the problem of grouping and interpreting a house roof from a fragmented intermediate representation. Figure 6.4 shows a

number of intermediate stages in the application of a house roof interpretation strategy associated with the house roof schema. Figures 6.4(a,b) portray a pair of region and line segmentations that exhibit the difficulties expected in the output of low-level algorithms; Figure 6.4(a) also shows the initial roof hypothesis. In this example the region segmentation algorithm was set to extract more detail from the image by producing a larger number of regions. The result, which is typical of a class of segmentation problems, is the fragmentation of the roof; the left portion, which was shadowed by trees, was broken into several regions separate from the main roof region. The set of lines intersecting the roof region are shown in Figure 6.4(c). When examined carefully, the line extraction results show line segments that are fragmented, multiple parallel lines, and gaps in lines.

The goal is to use these typical segmentation results to produce the trapezoidal region (which is almost a parallelogram) representing the perspective projection of a rectangular roof surface and to determine the orientation in 3D space of that surface. The top-down grouping strategy employed here is organized around evidence of the almost parallel lines forming the two sets of sides of the trapezoid. There are alternate strategies for other typical situations where some of this information is missing. Thus, this roof grouping strategy expects some evidence for each of the four sides, and in particular uses the long lines bounding the hypothesized roof region (Figure 6.4(d)). By merging similar regions which are partially bounded by the long lines (Figure 6.4(e)), removing shorter, parallel, almost-adjacent lines, joining co-linear line segments, and then fitting straight lines to the boundaries a partial approximate parallelogram can be formed as shown in Figure 6.4(f). Finally, Figure 6.4(g) shows the complete hypothesized roof trapezoid. The three-dimensional geometry of the roof can then be computed (up to some possibly non-trivial degree of error) based upon either the location of the pair of vanishing points of the two sets of image lines that are parallel in the physical world, or one pair of parallel lines and an assumption of perpendicular angles to a third line [96,97,98].

The point of this discussion is that the interpretation process required a flexible strategy for

grouping and reorganizing the lines and regions obtained from imperfect segmentation processes. At this point we have developed each such strategy independently, but we are beginning to define some standard intermediate grouping primitives that will form the basis of a variety of general top-down strategies (see Section 5); this area is being actively explored by our research group.

6.3 Interpretation

Interpretation experiments are being conducted on a large set of "house scene" images. Thus far, we have been able to extract sky, grass, and foliage (trees and bushes) from many of these images with reasonable effectiveness, and have been successful in identifying shutters (or windows), house walls, roofs, roads, and telephone/power lines in some of them. Object hypothesis and exemplar extension rules as described in previous sections were employed. Additional object verification rules requiring consistent spatial relationships with other object labels are being developed. The features and knowledge utilized vary across color and texture attributes, shape, size, location in the image, relative location to identified objects, and similarity in color and texture to identified objects. In the following figures, we show isolated intermediate and final results from the overall system.

Figure 6.5 shows selected results from the object hypothesis rules after exemplar extension and region merging. Some of the interpretation results shown were obtained using a different (somewhat coarser grained) set of initial segmentations than those presented earlier, and a different set of object hypothesis and exemplar extension strategies than those presented earlier.

Figure 6.6 illustrates typical interpretations obtained from a house-scene schema interpretation strategy that utilizes a set of object hypothesis rules for exemplar selection, extends the partial model from the most reliable of these hypotheses, and employs relational information for verifying hypotheses and predicting image location of object parts and related objects. The image areas shown in white in Figure 6.6 are uninterpreted either because the object did not exist in the knowledge network (and hence no label could be assigned), or because the object varied in some way from the rather constrained set of alternate descriptions of the object stored in the knowledge

base.

The interpretation shown in Figure 6.6 illustrates a problem which may be expected to occur quite frequently. The original interpretation was produced using a fairly coarse segmentation, which might be desirable from the standpoint of computational efficiency since fewer regions are involved. However, the sky and house wall are merged into one region as shown more clearly in Figure 6.7(a). An example of the effectiveness of semantically-directed feedback to the segmentation processes is depicted here. The missing boundary between the house wall and sky led to competing object hypotheses (sky and house wall) based upon local interpretation strategies. The region is hypothesized to be sky by the sky strategy; application of the house wall strategy using the roof and shutters as spatial constraints to select a region exemplar for house wall and then extend it via similarity of regions, color, and texture, leads to a wall hypothesis. There is evidence available that some form of error has occurred in this example: 1) conflicting labels are produced for the same region by local interpretation strategies; 2) the house wall label is associated with regions above the roof (note that while there are houses with a wall above a lower roof, the geometric consistency of the object shape is not satisfied in this example); and 3) the sky extends down the image close to the approximate horizon line in only a portion of the image (which is possible, but worthy of closer inspection).

In this case resegmentation of the sky-housewall region, with segmentation parameters set to extract finer detail, produces a reasonable segmentation of this region (Figure 6.7(b). It should be pointed out that in this image there is a barely discernable boundary between the sky and house wall. However, once the merged region is resegmented with an intent of overfragmentation, this boundary can be detected. Now, the same interpretation strategy used earlier produces the quite acceptable results shown in Figure 6.7(c). We note that this capability (of detecting labelling conflicts and resegmentation) was not automatic at the point these results were produced, but is now being implemented as part of Kohl's forthcoming doctoral dissertation on knowledge-directed

feedback under the control of a low-level executive [72].

6.4 Experience with Knowledge Engineering in the Schema System

Weymouth's [149] design of the schema network was a first approximation to a fairly complex system; from this experience we were able to distill some observations about design techniques [150]. We found that several generations of investigation and redesign of the schemas and interpretation strategies were required before semi-robust results could be achieved. A new schema system, based on this experience, is being designed and implemented; it is described in the next section. The integration of interpretation strategies into a schema network provides a mechanism for controlling the interpretation of a scene. Dependencies between and among schemas are expressed in both the relational arcs between nodes and in the interpretation strategies themselves. These dependencies, together with the results of partial interpretations, provide the basis for control of interpretation. This section outlines some heuristics about creating and using networks of this kind, and about how control of the interpretation process can be achieved.

A large range of knowledge has been integrated into a fine-grained representation of interpretation strategies that can successfully cooperate to build a global interpretation. Image interpretation in less constrained domains requires a large amount of knowledge about the particular domain, and the development of this knowledge bears similarity to the construction of "expert systems": the information must be collected, checked for consistency, experimentally verified, modified, and compiled. The challenge lies in the fact that the domain of vision is not one in which there is a clear understanding of the expert knowledge; it is extremely difficult for anyone to give a description (especially at the level of detail required by an expert system) of the information that we use for visual perception.

Deciding where to place information about object parts was one type of design decision. Decisions had to be rendered on whether to represent object parts as separate schemas, or solely within the schema of the whole object. The guiding principle in the early effort came from the number and

type of dependencies among the parts. In objects where there were complex relationships among the parts, the interpretation strategy for the object whole created the hypotheses for the parts. Thus, the creation of shutter and shutter-pair hypotheses was the dominion of the shutters schema, and the creation of the hypothesis for each individual wall of the house is the responsibility of the walls schema. On the other hand, for a schema in which there were methods for recognizing the parts which were less dependent on the relations between the parts (i.e., roof as part of house), the parts were constructed as separate schemas.

Our concentration on a modularized procedural representation aided in the development of interpretation strategies. We were often able to test a strategy in isolation before combining it with others. Our methodology was to attempt to construct interpretation strategies with as little interaction as possible with other interpretation strategies, relying on the verification stages to incorporate relational information. The limits of this approach were felt most acutely in the lack of sophistication in the type of communication and "coordination" available to the schema. When schema instances can only be linked to satisfy a goal, and then when they can only interact at a very coarse granularity in their respective strategies, they are necessarily limited in the types of cooperation that are possible.

Another type of design decision concerned control and communiation of information between schemas. When should the activation of schemas be made explicit and when should activation occur indirectly through STM [37]? When should explicitly activated schemas be activated in parallel and when should they be activated sequentially? How should interpretation strategies be designed to maximize the amount of overlap in processing? When objects are closely related - such as the house, walls, and roof - the amount of influence they might have on one another suggests that a direct communication link will be helpful (that is, through requesting a goal and communicating through the goal contract). For example, the interpretation of the house depends on the interpretation of its parts (roof and wall) and these were invoked directly. Furthermore,

the house strategy suspends processing (by waiting for the parts hypothesis) after making the goal requests. On the other hand, where the dependencies are less critical and more indirect (the objects on the ground-plane, for example), especially when the related objects are not essential for further processing, then the interpretation strategy was structured to do what processing it could and use indirect communication via the blackboard to obtain information about hypotheses. Periodic polling of STM for requested hypotheses can continue while a schema is engaged in other useful processing and when no other processing remains it can suspend activity until the requested hypotheses are posted in STM.

Parallel distributed control is being developed and explored in the use of schemas. Allowing schema activity to proceed, whenever feasible, increases the opportunities for parallel execution and encourages (to some extent) the early acquisition of evidence to support primary hypotheses. A simple principle to enhance parallelism is to request all parts and sub-classes early in the interpretation, thereby spawning multiple schema processes which can proceed in parallel.

6.5 Implementing Schemas

The development of a prototype schema system confronts many of the same issues that have arisen in other interpretation and control domains, such as speech understanding [80,81,159]. Among them are questions of the knowledge representation, the communication of information, error recovery and the selection of knowledge sources. Building upon the experiences of Weymouth [149] with the previous schema system, a new programming shell for research implementation of schemas as a set of concurrent cooperating processes has been developed by Draper [39]. The purpose of the shell is to encourage development and testing of schema strategies by optimizing the programming and testing time. A prototype shell is in place and various object schema types are at different stages of development and testing under the current shell. Experience from a set of early experiments will lead to improvements in the shell structure itself. The implementation is in Common LISP on TI Explorers, with image data and low-level processes handled on a VAX

11/780.

6.6 Inferencing Under Uncertainty

As we have shown, the construction of a semantic representation of a scene involves sources of information and processes whose local view of the data may introduce inherent ambiguities into the interpretation of that data. Sometimes it is not possible to say what the data represents but rather what it does not represent, i.e. some interpretations can be ruled out. In many cases, the data obtained from an image, and the partial interpretations obtained from them, can be considered to be partial *evidence* for or against the occurrence of semantically meaningful events. The schema strategies resonsible for constructing an interpretation of an image cannot be certain that the available information is complete, accurate, or independent. Thus, major problems facing systems which must perform inferences from this kind of data include the representation of uncertainty within the system, the mechanisms by which uncertain information is combined, and the manner in which uncertain evidence may be extended through inference.

For the past several years, our research group has been examining the use of the Shafer-Dempster theory of evidence [121,38] as the basis for evidential reasoning in the VISIONS system. Lowrance's thesis [85] showed how the Shafer-Dempster approach could be implemented in *dependency graphs*, which are the formal representations of dependency relations (arcs) between propositions (nodes). In this formalism, the likelihood of a set of propositions is represented by a distribution of a unit mass (similar to a probability distribution) over subintervals of the unit interval. The lower bound of the subinterval represents the degree of support provided to a proposition by a body of evidence while the upper bound of the subinterval represents the extent to which the proposition remains plausible. The width of the interval represents the uncertainty (e.g. ignorance) in the belief of the proposition.

Lowrance showed how mass distributions could be extended through the dependency graph from the set of propositions the evidence directly supports to those porpositions which it indirectly

supports. This work has since been extended by Lowrance and Garvey [86,44] and by Wesley, Lowrance, and Garvey [144]. Wesley et al [142] viewed image interpretation as the process of inferring belief in the presence or absence (in the image) of a subset of visual entities represented in the knowledge base and implemented LTM as a hierarchical dependency graph suitable for inferencing using the Lowrance model. The results of a limited set of experiments indicated that the inferencing model performed as expected and that the evidential model was an attractive alternative to traditional Bayesian-based inferencing mechanisms [40,67,76].

Reynolds et al [116,113] and Lehrer et al [79] address the problems of the automatic generation of mass functions, the computational complexity of Dempster's rule, and the mapping of continuous feature values into initial object hypotheses. They propose a unified computational framework for generating and combining evidence. This approach, using the concept of a plausibility distribution, is an alternative to the rule-based object hypothesis system introduced in Section 5; a large scale experiment comparing the two methods is currently underway [79].

Integrating information from multiple sources and propagating the combined information over the knowledge base is only one aspect of image interpretation. Complex domains require incremental extension of partial models, so that the constraints represented in the knowledge network can be exploited at the right time and at the right place. Consequently, a second important problem in image interpretation is the issue of control of the interpretation process: where should attention be focussed, what measurements should be made, how competing hypotheses should be resolved, and how the goals of the system can be accomplished. Wesley's thesis [146,147] extends the idea of evidential reasoning to control of knowledge based systems, in particular to control in the VISIONS image understanding system [145]. In Wesley's system, a second *control* layer is added to the VISIONS system. Control related information is obtained by control knowledge sources, which measure the state of the interpretation process. This information is evidential in nature and is used to reason about the best course of action to pursue in order to extend the current partial

interpretation.

6.6.1 The Schema Shell

The Schema Shell defines an object called the *schema*. A schema is a software construct which contains all the system's knowledge about some high level task. Schema instances are active entities that apply this knowledge to accomplish the task. Schema instances are restricted to communicate through a blackboard to coordinate their actions. Of course this methodology will allow simulation of message passing as a more direct form of communication. When the presence of an object is suspected, a schema instance of the corresponding object class is created; this schema instance runs concurrently with any other active instances. If aspects of its chore can be done independently, it can spawn multiple concurrent *strategies* which share a common memory, adding another level of parallelism to the system. The instantiated schema determines what further low- and intermediate-level processes to run, and incorporates the results in its object instance model. In this way it provides a vertical structuring of both knowledge and control. This form of schema design is both a simplification and a generalization of the schemas used by Weymouth [149]. Major differences include the adoption of a single, general purpose communication mechanism, the separation of local memory from communication, and a simplified method of schema creation. In particular Weymouth chose a strategy of structuring hypothesis and verification strategies which needed to be successfully executed prior to a schema writing a hypothesis on the blackboard; thus, weak hypotheses were not made visible and kept internal to the schema instance. Draper [39] is now exploring a strategy of allowing weak hypotheses to be posted with associated weak confidences, leaving the responsibility of whether such information should be used to the schemas reading this information, rather than those writing it.

In order to arrive at a coherent global interpretation, schema instances must communicate with each other. This is achieved without violating modularity by means of a central blackboard (BB). All messages are written to, and read off, the blackboard, as opposed to standard object-

oriented systems, in which messages are passed directly between the objects [102]. This severs the link between the sender and receiver of a message. A schema that produces information is not required to know of all the schemas that use it, nor do any two instances have to be active at the same time to communicate. Similarly, a schema instance does not need to know of the possible sources of a message. Another effect of the blackboard is that schemas choose which messages they will receive. Instead of the function/caller semantics associated with most message passing systems, communication in the schema shell is a cooperative venture between two (or more) equal participants.

Posting a message to the blackboard simply requires a message and a blackboard section name to which it is written. The mechanisms for reading, erasing and modifying messages, on the other hand, are of necessity more complex. To read a message, a schema instance must specify the section to be read from, and some method for selecting among the messages present on that section. The selection criterion is provided in the form of a predicate; all messages evaluating to true under the predicate are returned (in a list) by the read function. A problem occurs when no messages are found. There is then a choice between returning the empty list or suspending the schema instance until a suitable message has been written. Since both are useful in different circumstances, the schema shell provides two basic reading functions, *read-or-nil* and *read-or-wait*. If no messages are found, the former returns nil, while the latter suspends the calling schema instance. A suspended instance is said to be *sleeping*, and we expect it to incur extremely little overhead until a suitable message is written and the instance is awakened. The Shell also provides routines for removing messages from the blackboard as they are read (*erase-or-nil* and *erase-or-wait*) and for altering existing messages (*modify-blackboard*). Messages may also be selected according to when they were written (or most recently modified).

6.6.2 The Schema Template

Weymouth [149] developed a set of VISIONS schemas which used five basic types of object recognition (i.e. interpretation) strategies. He provided a limited demonstration of feasibility in a natural domain, carrying out experiments on four house scenes in six images. However, this research only begins to explore the possibilities for object recognition using schemas. Therefore, in order to encourage the development of new and different object and scene recognition schemas that can productively cooperate, a set of conventions in the form of a Schema Template has been developed to guide the knowledge engineer. However, we wish to emphasize that this is a part of ongoing research and has yet to be adequately tested. As our experience with schemas grows and we introduce new domains, these conventions will undoubtedly be altered. Our goal here is only to avoid the development of chaotic systems of schema networks.

The Schema Template divides the action of a schema into seven concurrent *interpretation strategies*, as in Figure 6.8. The OHM for (*O*bject *H*ypothesis *M*aintenance) strategy is activated when the schema is instantiated. Its task is to monitor the blackboard for information relevant to the object which the schema describes and to activate the other strategies as they become appropriate. Whenever data appears that is relevant, the OHM forwards it to the corresponding strategy. It is also responsible for the external appearance of the schema. The OHM maintains the object hypothesis on the blackboard, including the confidence measures, etc., and any goals that are posted by that schema.

The other six interpretation strategies involve specific recognition subtasks. The categories of interpretation strategies are meant not as a strict taxonomy of visual knowledge, but rather as a useful organization for the types of knowledge we have chosen to exploit thus far. The Initial Hypothesis strategy provides a focus-of-attention mechanism. Its job is to provide an indication of where in the image the other strategies should expend further effort, usually by generating inexpensive but plausible hypotheses. The other strategies are Extension, which extends the current

hypothesis based on partial results; Support, which exploits redundancy by seeking additional evidence for the current hypothesis; Consistency, which must reason about any missing or inconsistent tokens and relations; Conflict, which handles multiple interpretations of the image or 3D world; and Subpart, which activates schemas for expected visual subparts of the object. Obviously, not all strategies are needed or even meaningful for all objects; some objects, for instance, do not have subparts represented in the knowledge base. Furthermore, the boundaries between the strategies are drawn on a pragmatic basis. Subpart reasoning is a special case of Support, but was made a separate strategy to give the OHM more explicit control in expanding the compositinal hierarchy. Note that since schemas are independent of their parents, creating a subpart's schema could continue to an unknown depth.

Each interpretation strategy has access to a library of low and intermediate level routines which it uses as knowledge sources (KS's). The job of an interpretation strategy is to select those KS's which will aid it in its task and to run them. It must then integrate these results into the schema instance's current object model. As a result, a strategy is composed of code for object specific control and object model maintenance. The exceptions to this rule are the OHM strategies, which are the schema's interface to the global context. As we come to understand schema strategies better, we will provide improved tools for their construction.

As part of its task of monitoring the blackboard, the OHM must support error recovery and data synchronization. Errors emerge when later evidence refutes previously believed hypotheses. One solution to this problem is to maintain a history as hypothesis dependency graphs [55], depicting for each conclusion the hypotheses upon which it depends. Unfortunately, recalculating the conclusion from scratch in such systems can be expensive. The schema system therefore uses a procedural form of this technique that preserves the computational state. The template requires that an OHM never terminate, unless it has removed its object hypothesis from the blackboard. Instead, it waits for messages that might affect its hypothesis. In this way, the OHMs provide a procedural dependency

network. The invalidation of any hypotheses will cause all schema instances dependent on it to wake up and compensate for the loss. This is efficient only to the extent that the Schema Shell can ensure that any process waiting on a (currently) non-existent message consumes virtually no resources. While this is true in our current implementation, we are presently running on only a single processor. As we move during the next year onto a truly parallel, distributed machine in the near future, this method of error recovery will be tested experimentally for its feasibility and effectiveness.

OHM's are also responsible for data synchronization. Generally, a strategy will read messages of a certain type, processing each in turn. If, however, an OHM reads a message that alleviates the need for a currently executing KS, it can abort or reset the strategy before that KS has completed. Although currently unused, this ability could allow the system to synchronize data if necessary. It must be kept in mind that the ideas just presented here are currently being developed and evaluated in an empirical manner. New experiments are being developed in new domains, and the schema system is being modified and tuned as these domains are explored.

7 EXTENSIONS TO MOTION AND STEREO DATA

There are two obvious reasons why static image interpretation must be extended to analysis of multiple images. First, many robotics applications involve a sensor moving through an environment and information must be temporally integrated so that a dynamic model of the world is constructed. The development of dynamically changing egocentric representations of a world in motion is a challenging problem for computer vision. Second, the scene depth of objects and their surface properties are important features for the interpretation of a scene and subsequent interaction with the objects and the environment. Both motion and stereo may be able to provide this kind of information from multiple images, although perhaps only over restricted ranges. Consequently, a significant secondary focus of our research activity involves effcient recovery of this information and integration with the static image analysis components of the VISIONS system.

Let us briefly consider some of the extensions that are necessary as we bring static vision interpretation and motion analysis together. First, the current ISR must be augmented to include surface patches derived from depth arrays (sparse or dense) obtained from motion analysis. Second, tokens in the ISR should be enriched with depth and motion attributes: line and region tokens can have 2D motion attributes, while surface tokens can have 3D depth and motion attributes. The enriched representation provides additional features for interpretation; for example, a car moving on a road might be segmented in an image sequence more reliably, and hypothesized to be a vehicle more easily and reliably. Third, motion analysis of the past N frames allows predictive capability for the next sequence of frames [25,108,125]. Thus, much of the computation can involve monitoring the following frames for expected changes. Attention can then be focussed on areas which are different than expectations or where a more accurate analysis is required.

Our research has both empirical and a theoretical components. We will outline the key areas of our motion and stereo research, providing references of our own work, but we will not attempt to survey the extensive literature in these areas. For recent surveys in these areas refer to [4,7,22,69,71].

The goals of our work include the following:

1. Correspondence of 2D image events occuring in multiple images [5,6,7,47].

   This is a key capability that in some manner is at the foundation of the computation for any motion, stereo, or registration algorithm. Matching of local intensity windows via correlation, or points and lines via symbolic token matching are carried out for either a binocular pair or a motion sequence. The result is a disparity array in stereo analysis or a displacement field in motion analysis. Hierarchical processing, both for correlation matching and gradient-based methods, is an important and effective technique in motion analysis that is actively being explored in our laboratory [7,46,152].

2. Temporal Grouping of 2D Image Events.

   The coherence of an image event across a sequence of images should provide a robustness that is impossible to achieve in a single static image. Thus, perceptual organization of space-time edges into lines can use both dimensions where neither alone would suffice.

3. Recovery of relative sensor motion when it is unknown [1,2,3,77,78].

   This requires decomposition of general motion into its translational component and rotational component. In unconstrained general motion 5 parameters must be recovered, while in cases of constrained sensor motion fewer parameters specifying the sensor motion must be recovered (e.g. pure translation involves 2 parameters).

4. Computation of relative environmental depth from known or recovered sensor motion [1,2,3,25,77,78,125,126,153,154].

   Once the axis of translation is available (and the rotational component is removed), then displacements of image features are constrained to move along linear paths radiating from the focus of expansion (FOE), which is the point of intersection between the infinite image plane and the axis of translation (see Figure 7.1(a,b)). The rate at which the projection of an environmental point moves between frames is a function of the depth of the environmental point, the distance of its image plane projection from the FOE, and the relative rate of motion of the sensor to the environmental point. Given known camera parameters and a displacement field, the depth of environmental surface elements can be computed up to the accuracy of the image measurements.

5. Detection of independently moving objects [1,2,3,8].

   The recovery of motion parameters and environmental depth is complicated when the scene contains independently moving objects. The moving objects must be segmented from the static environment, so that the motion of the object can be recovered. The relative translation between the moving object and the moving sensor can be used to compute the depth of the moving object. This is a very difficult problem and may not be solvable in many situations due to inherent ambiguities in the measurements.

6. Computation of depth from stereo [7,152].

   This area is well-studied, but still needs to be integrated with motion processing. Processing of stereo and known translational motion have the common problem of obtaining depth from feature point correspondences along constrained match paths, although the problem of ambiguity in the match process is still an issue.

7. Fusion of stereo and motion processing.

   Constraints from binocular motion should allow more effective depth recovery than either by itself. This area is only beginning to receive attention in the field.

8. Surface recovery from depth arrays.

   Once depth is available, then the problem of recovering surface information must be considered. There is growing interest in this area, but there has been limited research in the field up to this point in time (see also Section 3.3.4.), and one can expect many difficult problems similar to those for intensity and/or color segmentation.

Our current research projects have made significant headway in a number of these areas. The research has a theoretical foundation, but a significant consideration has been the development of robust algorithms tested on synthetic data with ground truth, as well as motion sequences derived from a sensor moving through natural scenes. Developments within the laboratory include a robust algorithm (developed by Lawton) for recovery of sensor translation [77,78,108], possibly the only algorithm at this time for detection of moving objects under general unknown sensor motion that has been successfully applied to real data (developed by Adiv) [1,2], a 2D matching algorithm that integrates a hierarchical orientation-dependent confidence measure with a smoothness assumption in the extraction of dense displacement fields (developed by Anandan) [5,6,8], and a hierarchical gradient-based relaxation method that achieves local smoothness in a dense displacement field (developed by Glazer) [46,47]. This set of algorithms demonstrates some degree of effectiveness for the recovery of motion parameters, displacement fields, and depth arrays in a variety of situations, although the techniques have not been integrated into motion systems for analyzing natural images.

As a first step in the process of integrating motion analysis with our interpretation system, we have chosen to focus on some well defined subproblems of the general problem of a dynamic sensor translating through an unknown environment. We are seeking to determine a sparse depth map of environmental points under translational motion that is known up to some limit of experimental accuracy in the motion parameters [25,125,126]. The local ambiguity in the matching process (often due to the resolution of the discrete matching process) leads to ambiguity in recovering environmental depth. Bharwani has developed an algorithm that uses correlation matching at

a coarse image resolution to obtain a coarse estimate of depth from an initial pair of frames. The coarse depth result is used to predict displacements and a search window in a future frame. Additional frames are then used to refine the depth map via a finer resolution search for a match in the localized area (see Figure 7.1(c)). The technique allows refinement in depth estimates over the sequence of images while maintaining constant computational limits on processing between frames: as the accuracy of the environmental depth increases, the search window gets smaller and the match resolution can be correspondingly increased. Thus, both "start-up" and "updating" strategies follow naturally as part of a temporal and spatially hierarchical processing paradigm.

The inaccuracies actually are not one-dimensional as implied in Figure 7.1(b) due to digitization error (among other sources of error). Snyder [125] provides a thorough quantitative analysis of the effect of uncertainty in the locations of the FOE and image feature points in a pair of digital frames. A reasonable default assumption is that there is some degree of uncertainty due to the digitization process, typically $\pm$ 1/2 pixel when subpixel interpolation has not been used. Thus, the search window is actually two-dimensional as in Figure 7.1(d). The analysis includes the estimated accuracy in depth from previous frames to form an error range in computed depth, and with the error range in locations of FOE and feature points, a two-dimensional search space can be computed for future frames. Snyder's main conclusions are that uncertainty in the feature points (U_p) dominates uncertainty in the FOE (U_{foe}) unless U_{foe} is much larger than U_p. This theoretical analysis has lead to a modification of the windows to a two-dimensional area as shown in Figure 7.1(e). Note that the best match in successive frames shows the object moving and more accurately estimates the image displacement and hence the depth range.

Finally, there is a similar analysis underway by Snyder related to the relative accuracy of stereo and depth computation for environmental points at different distances and which appear in different locations in the image relative to the FOE. This analysis will also explore the theoretical limit of stereo and motion analyses to discriminating a moving object from the adjacent static background

at a similar distance [126].

As these algorithms provide information on the depth of environmental points, they will be used to enrich the ISR so that more effective interpretation can take place. In addition, there is a research project underway to use both efforts to navigate a mobile vehicle in an outdoor environment on the UMass campus [11,12].

8 PARALLEL ARCHITECTURES

8.1 Architectural Requirements for Vision

Real-time machine vision has turned out to be one of the most computationally intractable domains of artificial intelligence research. It requires that an interpretation of a changing scene be updated with every new video frame, once every thirtieth of a second, or no worse than once every N frames, for some small N (e.g. 3-15). Each color video frame of reasonably high resolution (512x512) contains approximately three-quarters of a million values. Thus, just performing a single operation on each data value would require executing about 23 million instructions per second to keep up at full video rate. However, many researchers believe that vision requires 2 to 4 orders of magnitude more computation.

Some of the architectural issues to be addressed for vision stem from the specific requirements of the problem:

- The ability to process both pixel and symbol data.
- The ability to transform an image into a set of meaningful symbols that describe it.
- The ability to select particular subsets of data for varying types of processing.
- The ability to quickly access both low and intermediate data without having to move large amounts of information in and out of the processor.

A key design goal for an effective architecture is the ability to maintain the low and intermediate representations simultaneously in the same machine: sensory data in pixel format and symbolic region, line and surface representations. The necessity of transferring an image to an external processor for evaluation by a sequential program in order to continue further processing must be avoided. It is time consuming to transfer the volume of information contained in an image and its derived descriptions. Even if it took no time to accomplish the transfer, the time required for serial evaluation would still be too great. Instead, the low-level vision processor must be able to provide enough feedback to the controlling processor to allow all of the operations to take place within the low-level vision machine itself.

There are many types of parallel architectures that have been proposed for use at specific levels of vision processing. At this point in the development of the field of computer vision there is no consensus among leading researchers on approaches to real-time vision. However, as we outlined earlier, there seems to be a growing consensus towards the need for at least two, and probably three forms, of processing. Consequently, there almost certainly needs to be at least two types of processing elements, and possibly three, with different levels of computational granularity.

Thus, we believe that the most significant architectural conclusions derived from an understanding of the full computer vision problem are:

1. the multiple levels of representation and stages of processing require very different types of processing elements, in terms of both granularity of data and the form of algorithms that need to be applied; and
2. the communication pathways between, and within, the processing levels and their bandwidth requirements must be considered an integral part of the architectural structure [83].

8.2 Processing Characteristics at Each Level of Vision

We have mapped the three levels of abstractions discussed throughout this paper into a three-level tightly coupled architecture in which each level has the appropriate structure for the set of tasks at that level. The most well understood stage is the low-level where there is uniform computation at local points, usually at each pixel across the image. At this level we are concerned with pixels, which are numerical image data organized as a spatial array. Operations such as smoothing, edge detection, region segmentation, feature extraction, and feature matching between frames in motion and stereo processing are performed in SIMD fashion. Some of these operations may require processing over larger image distances, and therefore this type of processing may overlap some forms of intermediate-level processing discussed next.

At the intermediate level our concern is with symbolic tokens for lines, regions, and surfaces which specify extracted image events. The types of operations needed at this level are partitioning and merging operations. These tokens are transformed into more useful structures on the basis

of information from both the low-level operations and high-level hypotheses about the possible objects in the image. The intermediate level also requires specialized processors. Consider the large numbers of line and region fragments that can be generated by even the most robust low-level algorithms. To perform grouping operations we need a large amount of less local communication between intermediate processors. We need to match fragments of lines and merge them across potentially large fractions of the image. Similarly, regions need to be merged, and sometimes compared with others from possibly non-contiguous areas. For it to be done quickly requires architectual support for both inter-level and intra-level communication, as well as a flexible data manipulation repertoire of instructions. The Intermediate Symbolic Representation is a database which must operate as a server for the queries made by both the high-level and intermediate-level processing elements.

At the high level we are concerned with semantic processing involving mechanisms for focus of attention, the formation and verification of object hypotheses, and knowledge-based inference using complex control strategies for fusion of information from multiple knowledge sources. This type of processing involves extensive non-local symbolic computation and is most effectively performed on LISP type machines. The computational and communication needs of these processes is provided by a network of high-level processors which provide the third tier of processing power needed to solve the image understanding problem. Knowledge-based processing involves structures such as frames or schemas, with associated rules to be applied to tokens, in order to carry out interpretation using the context of expected objects and their relationships.

8.3 Communication Between Processing Levels

Central to vision processing is the flow of communication and control up and down through all representation levels. In the upward direction, the communication consists of image abstractions and segmentation results from multiple algorithms (and possibly from multiple sensory sources). It also involves the computation of a set of attributes of each extracted image token to be stored in

a symbolic representation. Summary information and statistics allow processes at the higher levels to evaluate the status of low-and intermediate-level operations.

In the downward direction the communication consists of knowledge-directed control of processing and grouping operations, the selection of subsets of the image, the specification of further processing in particular portions of the image, the modification of parameters of lower level processes, and requests for additional information in terms of the intermediate representation.

Thus, a key problem in selecting the appropriate two or three types of processing elements for the required computation is how the communication between machines can be structured. One cannot simply plug two interesting machines together if they cannot achieve the required communication bandwidth between them. Levitan's recent doctoral dissertation [83] documents research in the UMass architecture group over the last several years to identify and quantify the important aspects of communication in parallel algorithms and architectures.

8.4 Motivation of Associative Processing

Because associative computation may not be very familiar to many computer scientists and computer vision researchers, let us briefly motivate their concepts in a bit of detail [43,137]. There are four processing capabilities that we will consider basic to associative computation:

1. global broadcast/local compare,
2. some/none response,
3. count responders, and
4. select first responder

Associative processing is a technique whereby the processors of the array have the ability to compare sets of data broadcast from a central controller to their own local (pixel) data. They can then conditionally process both local data, as well as broadcast data based on the results of those tests. As a simple example, processors can conditionally store different broadcast tags based on the values of local data and thus *associate* their values with tags to support symbolic processing.

Associative processing can best be understood by an example of a single controller (a teacher) interacting with an associative array (a class of students) [43,136]. A teacher who needs to know if any student in a class has a copy of a particular book can ask each student, in turn, if they have the book. This would correspond to a computer program running sequentially though all the pixels in an array (or all the region tokens in a symbolic representation) looking for a particular intensity value. On the other hand, the teacher can simply state, "If you have the book, raise your hand." The students each make a check, in parallel, and respond appropriately. This corresponds to a broadcast operation from the contoller, and a local comparison operation at each pixel in an array to check for a particular value. Both operations assume that the local processors have some "intelligence" to perform the comparison. Clearly, the cost of the extra local intelligence is offset by the savings gained in time, which for images is on the order of the number of pixels (i.e. 1/4 of a million).

Query and response is just the first part of associative processing. Thus far, only a scheme for content addressable comparison ("If your hand is up, I'm talking to you.") has been described. To perform associative processing, we must be able to conditionally generate tags based on the value of data and use those tags for further processing. An example of this kind of association would be, "If you have an excuse from your mother or a doctor, then you can consider yourself 'EXCUSED'." A condition of the data has been associated with a label or tag. For region tokens simple operations could be performed such as: if the intensity, color, and texture attributes are each in a certain range (i.e. rules on token attributes), label yourself "POSSIBLE-SKY". The interesting processing comes when the controller starts performing multiple logical operations on tags. Since each pixel or token could have multiple tags based on properties of the data, as well as things like spatial coordinates, operations could be performed like: if you are NOT-TREE and NOT-ROOF and (POSSIBLE-SKY or POSSIBLE-CLOUD) and IN-TOP-OF-IMAGE then label yourself LIKELY-SKY. As processing continues, only sub-sets of the pixels are involved in any particular operation. Pieces of the image

are selectively processed based on their properties, but all pixels with a given set of properties are operated on in parallel.

The ability to associate tags with values is half the battle for high-speed control. Responses must be received back from the array quickly. Forcing the teacher to ask each student if they have their hand up defeats the process. The teacher can see immediately if any of the students have their hands up, and can quickly count how many do. Similarly, a Some-response/No-response (Some/None) wire running though the pixel array allows the controller to immediately determine properties about the data in the array, and therefore the state of processing in the array *without looking sequentially at the data values themselves.*

Additionally, fast hardware to perform a count of the responders allows the controller to see summary information about the state of the data in the array. Programs can then be written that can *conditionally* perform operations based on the state of the computation. The teacher can see if most students have their books, or not, and plan the lecture accordingly. By using the properties of the radix representation of numeric values in the array we can use the counting hardware to sum the values in the array. The ability to sum values gives us the power to compute spatially weighted averages such as the center-of-mass of a region.

These examples of students and pixels illustrate the power of associative processing. Associative processing is a paradigm for communication in the upward direction and control in the downward direction between each pair of levels in the hierarchy. Criteria can be broadcast for selecting pixels, or regions, or symbolic tokens for selective processing. In this way higher levels control lower levels. The response that comes after processing data can be tested and/or counted to control conditional branching for the next step of processing in a given algorithm. Thus, the lower levels provide feedback to higher ones.

The associative select operation "select-first-responder" provides one more mechanism for transferring non-summary information from a lower to a higher level in the vision processing hierarchy.

The higher level may select a single value in a lower level and read it out. This transfer of information is one-to-one: a single value at a lower level is copied to a higher level. The select-first operation is also useful for spreading-activation algorithms, where an initial seed point must be chosen.

8.5 Overview of the UMass Associative Image Understanding Architecture (IUA)

The University of Massachusetts (UMass) is developing a three-level tightly-coupled associative architecture to achieve real-time performance at all levels of vision processing [84,90,138]. The UMass Associative Image Understanding Architecture (the Umass Associative IUA) combines an integrated approach to the three types of computation outlined, including the critical problems of communication between the three levels. The proposed architecture is a synthesis of an associative (or content addressable) processor and a cellular array processor at the bottom level (developed in Weems's doctoral dissertation [137]), interfaced to a network of intermediate communications processors, and a network of high-level symbolic processors at the top level.

The IUA is depicted in Figure 8.1 with the following levels:

- CAAPP (Content Addressable Array Parallel Processor [138].

  A 512x512 array of one-bit-serial ALUs and associative processing capability;

- ICAP (Intermediate Communication Associative Processor)

  A 64x64 array of 16-bit-parallel ALUs and associative processing capability;

- SPA (Symbolic Processing Array)
  An 8x8 array of microprocessors capable of LISP symbolic processing, each of which can serve as a global associative controller to the sub-arrays below.

The associative nature of the UMass IUA (at all levels), combined with the tight coupling between levels, supports the high-bandwidth inter-level communications and control needed to

perform both sensory-to-symbolic transformations and high-level symbolic processing. Of particular importance is the ability to store both sensory data and the symbolic representation of lines, regions, and surfaces in the same physical cellular memory at the bottom two levels so that the global associative processing mechanisms can be applied to both without any movement of data.

The key to associative parallel processing is that data can be queried and retrieved in parallel extremely quickly. The retrieval mechanism is based upon accessing data by its content. The machine allows global broadcast from the controller to all cells; an activity bit to be set by each cell for its response; and the global response from the array of cells to the controller in terms of a count, some/none and select-first functions. In particular, a comparand may be broadcast from central control, and cells whose contents fail to match the broadcast comparand will be turned off so that operation of exact match to comparand, greater than (less than) comparand, maximum, and minimum, may each be performed in parallel on all cells of the memory. Thus, in the intermediate symbolic representation, one can set the activity bits for region tokens which are blue and high in the image by properly broadcasting comparison instructions to all cells, and then the number of responders, if any, can be ascertained immediately. If desirable, the find-first capability will allow sequential access by the global controller to all responding cells at approximately the instruction rate of the machine.

The CAAPP array is a 512x512 SIMD array of 262,144 simple processing elements, with 64 cells on a chip and 8x8 chips on a board. Thus, the design includes a processing element for each pixel at the sensory processing level with 256 bits of RAM, five register bits, and a one-bit ALU for bit-serial arithmetic and logic functions, and simple data routing circuitry. Information in each cell can be moved North, South, East, or West on the array so that neighboring cells can communicate with each other. It excels at very tightly coupled, very fine-grained parallelism; its architecture is especially oriented towards associative processing with global summary feedback mechanisms.

The ICAP is also a square grid array processor ((N,S,E,W) communication network), intended to

perform intermediate-level processing on image events and tokens that are in physical registration with the pixel data. It provides data reduction and a high bandwidth communication channel between the lower level processors and the SPA, (see below) and allows partial results from widely separated image events to be combined quickly. There will be an array of 64x64 ICAP elements, one for each chip in the CAAPP array at the lowest level. Each of the 4096 ICAP processors consists of a 16-bit parallel ALU, 64K bytes of RAM, and data routing hardware.

The Symbolic Procesing Array (SPA) is configured as a network of 64 powerful, general purpose microprocessors intended for performing MIMD high-level symbolic operations, and for controlling sub-array processing in the ICAP and CAAPP arrays. Each SPA processor is associated with an 8x8 array of ICAP processors and 64x64 array of CAAPP cells. The communication between processes will primarily be in terms of a "blackboard" architecture with communication managed by the processes themselves. As these processes run and make requests to the ICAP (and sometimes directly to the CAAPP), they extract information about the image and post the results of their analysis on the blackboard for other processes to use. At the beginning of processing, the lower arrays will be controlled by a single global controller and run entirely as a full-scale SIMD processor; at later stages when hypotheses are posted on the blackboard the machine will be multi-SIMD with each of the 64 symbolic processors controlling an ICAP and CAAPP sub-array to run independent processing strategies.

During the last four years the architecture and vision research groups at UMass have gone through three designs of the CAAPP, two VLSI layouts of the CAAPP, built a set of simulators, implemented 35 algorithms, analyzed and revised instruction sets, and produced two doctoral dissertations on associative processing and parallel algorithms that form the basis of this work [83,137]. The ICAP and SPA are more recent design stages. A test chip of the 3μ NMOS version of the CAAPP processing element has been received from the MOSIS facility and is undergoing testing. The ICAP is being designed now and in its first implementation may be built with a

commercially available chip. The goal of our research effort is the construction of a 1/64 processor slice of the proposed hardware architecture with an associated software support environment; the implementation and final design effort is being carried out in cooperation with Hughes Research Laboratories, which will be building the initial board(s). The initial prototype will be a board with 4096 CAAPP cells, 64 ICAP cells, and controlled by one Lisp processor.

8.6 The VISIONS System on the IUA

The VISIONS system on the IUA can be depicted by modifying Figure 6.2 as shown in Figure 8.2. The SPA implements the schema system and communicates object hypotheses through the STM blackboard. The three major differences in the diagrams reflect:

1. the mapping of low-level sensory processing onto the CAAPP array;
2. the embodiment of the ISR (i.e. the intermediate tokens) in the CAAPP and ICAP memory, instead of having the lower half of the STM blackboard as a separate data base; and
3. the parallel and associative nature of communication and retrieval of data between the SPA and the ICAP and CAAPP arrays.

The last point that should be noted is that the ISR can be shared in the CAAPP and ICAP memory. The information in the 64 CAAPP cells (64 x 256 bits = 16k bits) can be copied into the associated ICAP cell (64k bytes x 16 bits = 1024k bits) in a fraction of a frame time, and high-level queries can access information from either at approximately instruction rate. The symbolic data in the CAAPP can be structured via a linked list so that any token of sufficient size (i.e. greater than some size threshold of pixels) can use the memory of the CAAPP cells as a linked list, storing an ordered set of token attribute-value pairs across the memory of the cells/pixels in each region.

Of course we expect there to be many questions in the reader's mind concerning the implementation of the algorithms that we have described in this paper. Any reasonable level of detail is beyond the scope of the broad presentation of this paper. Thus, we can only briefly motivate the techniques that underlie the algorithms; in that regard we provide a single algorithm in a bit of detail in the next section.

8.7 Implementing the Straight Line Algorithm on the Associative IUA

The straight line extraction algorithm described in Section 3.2 was modified to take into account the massive parallelism available on the IUA and will be used to motivate how the structure of the machine might be used. The general idea is to run the line detection part of the algorithm in a hierarchical fashion using CAAPP cells, one ICAP processor, and one SPA processor. The resulting line support (or gradient orientation) regions extending over the artificial boundaries are merged in a later step. Thus, each pyramid is concerned only with the area of the image underneath it. The timing information included here is an estimate of the execution times of the various portions of the algorithm; it reflects all data movement times as well as algorithm execution times.

The line computation process begins by loading the image into the CAAPP, computing the X and Y directions of the gradient at each cell, and then uses a table lookup technique to generate the gradient direction to a resolution of 1 degree. Each cell now has the gradient direction and magnitude. The quantized orientation bucket for each cell is then determined by broadcasting the bucket boundaries to each cell (in parallel). Since there are two sets of buckets (rotated from each other by 22.5 degrees), this is done twice. The total execution time up to this point is approximately 1/2 ms, assuming an 8-bit image. Next, a connected components algorithm is executed independently in the CAAPP area under each ICAP cell for each of the two partitions generated by the overlapping orientations, and one of these is selected by a voting process using a select-first and response count scheme. The connected components and voting process takes approximately one ms.

A connected components algorithm is now executed at the ICAP level over the 8x8 ICAP cells associated with one SPA processor. This involves associatively accessing the line support region attributes from the CAAPP level and requires approximately 3 ms. Since some regions extend outside the boundaries of one SPA, the whole process is repeated at the SPA layer and requires approximately an additional 5ms. At this point, all of the attributes of a line support region are

either in the memory of an ICAP or a SPA processor. For each region, the line corresponding to the moment of maximum inertia is found, rather than by intersecting the two planes in the original algorithm. Here, both ICAPs and SPAs are running concurrently. Assuming that each ICAP or SPA is computing for no more than 3 or 4 lines, the maximum time for this step is roughly 3-4ms.

Finally, the line attributes are broadcast back to the CAAPP level to the endpoints and other attributes (if necessary). This is accomplished at the controller level and involves broadcasting the line label and characteristics. Each pixel in each line support region then calculates whether it lies on the line. The broadcasting step for the line information is done sequentially for each region and the pixel calculations at the CAAPP level is done in parallel for all regions. The first step requires approximately 3.5 us for each region; assuming 1000 regions the time required is on the order of 3.5ms. The second step requires 1ms.

At this point, each line is represented by the pixels which belong to it and a line label; total execution time of the algorithm is roughly 17 ms.

9 Conclusions

The VISIONS system has undergone a continuous evolution over the years, driven primarily by a deeper appreciation of the nature of image understanding as both our research and that of the field have progressed. The current version of our system has produced reasonable interpretations of fairly complex outdoor scenes (e.g. house scenes and road scenes), and variations of the system have been applied to aerial images with some success. We view the system as an experimental testbed within which further evolution may be accomplished; we expect that there will be long term but slow improvement in the power and capabilities of the system as particular areas/modules/system components are improved and generalized.

Several areas of research are being vigorously pursued. One of these is the integration of motion into the interpretation process and the concurrent investigation into dynamically changing egocentric world representations. A second component of the system which needs significant development (and there is some effort underway) is the three-dimensional representation of shape and methods for using three-dimensional information during interpretation. Although the current system has some three-dimensional knowledge of objects and can infer some three-dimensional structure, the capabilities are not adequate for a system expected to exhibit a reasonable degree of generality. There is an effort underway that involves compilation of 3D information into a network of 2D characteristic views associated with objects and scenes [36]. A third area of research involves the inferential/evidential reasoning capabilities of the system, particularly methods for inferencing over the knowledge base using partial information from many sources. We have explored Dempster-Shafer mechanisms for evidential reasoning over the past 3-4 years [85,113,116,142,143,146], and we now have a version of an object hypothesis system based on feature evidence that operates in a manner similar to the rule-based object hypotheses presented in this paper; comparisons of effectiveness are planned. However, the use of Shafer-Dempster evidential approaches to carry out propagation of partial beliefs/evidence in a semantic network has not yet developed to a point

where it is clear how and whether to integrate it into our system. Mechanisms for propagating confidences/probabilities/beliefs will contine to be a primary research issue. Fourth, the knowledge base of the system needs to be expanded to more objects and more intepretation strategies, and the system needs to be exercised on a wider variety of images. We have a series of experiments planned for the near future which will exercise both our entire system and test our research group's ability to build a knowledge base for new task domains relatively quickly. Finally, in the UMass Image Understanding Architecture, we are actively exploring computational architectures suited to the types of processsing inherent in image understanding, explicitly taking into account the set of algorithms presented in this paper and the bandwidth requirements of the communication channels between the different levels. When a physical slice of the architecture is constructed it will be used to run real-time vision interpretation experiments.

Acknowledgements

The authors wish to thank the members of the VISIONS research group, both past and present, for their contributions to a long term dream. Since the beginning there have been moments where particular individuals have provided significant leadership in the maturation of the ideas

We wish to specifically acknowledge several individuals who have made major contributions to our twelve year effort. Tom Williams was there at the inception of the VISIONS project and contributed greatly to the design of the system through the first 6 years. Ralf Kohler designed the VISION software development environment that has supported empirical development in a flexible manner for many years. Joey Griffith has had his fingers in almost every aspect of the project for the last 6 years. George Reynolds has taken on a significant leadership role in the development of many aspects of intermediate processing. We also wish to acknowledge the leadership provided by Daryl Lawton and P. Anandan in developing our motion group.

In addition, a number of other people have made their presence felt including Terry Weymouth, John Lowrance, Frank Glazer, Paul Nagin, Bryant York, Gilad Adiv, Charlie Kohl, Len Wesley, Cesare Parma, and current members Bruce Draper, Rob Belknap, Bob Collins, Michael Boldt, Brian Burns, Ross Beveridge, Nancy Lehrer, Seraj Bharwani, Val Cohen, Jim Burrill, Bob Heller, Mark Snyder, Igor Pavlin, Rich Weiss, Les Kitchen, Chip Weems, and Steve Levitan.

We also wish to thank Janet Turnbull for her continuing support over the years, and Laurie Waskiewicz for her perserverance and good humor during the production of this paper.

10 References

REFERENCES

[1] G. Adiv, "Determining 3-D Motion and Structure from Optical Flow Generated by Several Moving Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Volume PAMI-7, July 1985, pp. 384-401.

[2] G. Adiv, "Interpreting Optical Flow," Ph.D. Dissertation, Computer and Information Science Department, University of Massachusetts at Amherst, September 1985.

[3] G. Adiv, "Inherent Ambiguities in Recovering 3-D Motion and Structure from a Noisy Flow Field, " *Proc. of the Computer Vision and Pattern Recognition Conference,* San Francisco, CA, June 1985, pp. 7077.

[4] J. Aliomonos and A. Basu, "Shape and 3D Motion from Contour without Point-to-Point Correspondence: General Principles", *Proc. of the Computer Vision and Pattern Recognition Conference,* pp. 518-527, 1986.

[5] P. Anandan, "Computing Dense Displacement Fields with Confidence Measures in Scenes Containing Occlusion," *SPIE Intelligent Robots and Compter Vision Conference,* Volume 521, 1984, pp. 184-194; also *DARPA IU Workshop Proceedings,* 1984; and COINS Technical Report 84-32, University of Massachusetts at Amherst, December 1984.

[6] P. Anandan and R. Weiss, "Introducing a Smoothness Constraint in a Matching Approach for the Computation of Optical Flow Fields," *Proc. of the Third Workshop on Computer Vision: Representation and Control,* October 1985, pp. 186-196, also in *DARPA IU Workshop Proceedings,* 1985.

[7] P. Anandan, "Motion and Stereopsis," COINS Technical Report 85-52, University of Massachusetts at Amherst, December 1985; also to appear (in Spanish) in *Vision por Computador,* (Carme Torras, ed.), to be published by Alianza Editorial, Spain.

[8] P. Anandan, "Measuring Visual Motion From Image Sequences", Ph.D. Dissertation, University of Massachusetts at Amherst, January 1987.

[9] M.A. Arbib, "Segmentation, Schemas, and Cooperative Computation", in *Studies in Mathematical Biology, Part 1,* S. Leven, ed., MAA Studies in Mathematics, Vol. 15, 1978. pp. 118-155.

[10] M.A. Arbib, "Perceptual structures and distributed motor control", in *Handbook of Physiology: The nervous system, II Motor control,* V.B. Brooks (Ed.), Amer. Physiol. Soc., Bethesda, MD, pp 1449-1480.

[11] R. Arkin, "Path Planning and Execution for a Mobile Robot: A Review of Representation and Control Strategies," COINS Technical Report 86-47, University of Massachusetts at Amherst, October 1986.

[12] R. Arkin, "Path Planning for a Vision-Based Autonomous Robot," COINS Technical Report 86-48, University of Massachusetts at Amherst, October 1986.

[13] R. Bajcsy and M. Tavakoli, "Computer Recognition of Roads from Satellite Pictures," *IEEE Transactions on Systems, Man, and Cybernetics,* **SMC-6**, September 1976, pp. 623-637.

[14] H.H. Baker, "Depth from Edge and Intensity Based Stereo", *Report No. STAN-CS-82-930,* Department of Computer Science, Stanford University, California, September 1982.

[15] D. Ballard, C. Brown, and J. Feldman, "An Approach to Knowledge-Directed Image Analysis," in *Computer Vision Systems* (A. Hanson and E. Riseman, Eds.), Academic Press, 1978.

[16] D. Ballard and C. Brown, *Computer Vision*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982.

[17] H.G. Barrow and J.M. Tenenbaum, "Computational Vision", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1981, pp. 572-595.

[18] H. G. Barrow and R.J. Popplestone, "Relational Descriptions in Picture Processing", in *Machine Intelligence 6*, (B. Meltzer and D. Michie, Eds), Edinburgh University Press, Edinburgh, 1971.

[19] H. Barrow and J. Tenenbaum, "MSYS: A System for Reasoning About Scenes," Technical Note 121, April 1976, AI Center, Stanford Research Institute.

[20] R. Belknap, E. Riseman, and A. Hanson, "The Information Fusion Problem and Rule-Based Hypotheses Applied to Complex Aggregations of Image Events", COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1987.

[21] R. Belknap, E. Riseman, and A. Hanson, "The Information Fusion Problem and Rule-Based Hypotheses Applied To Complex Aggregations of Image Events," *Proc. DARPA IU Workshop,* Miami Beach, FL, December 1985.

[22] S.T. Barnard and M.A. Fischler, "Computational Stereo", *Computing Surveys*, Vol. 14, No. 4, December 1982, pp. 553-572.

[23] P.J. Besl, and R.C. Jain, "Range image undersanding," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, Calif., June 9-13, 1985, New York, pp. 430-451.

[24] R. Beveridge, A. Hanson, and E. Riseman, "Segmenting Images using Localized Histograms," COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1986.

[25] S. Bharwani, A. Hanson, and E. Riseman, "Refinement of Environmental Depth Maps over Mutliple Frames," *Proc. DARPA IU Workshop,* Miami Beach, FL, December 1985.

[26] I. Biederman, A. Glass, and E.W. Stacy, "Searching for Objects in Real-World Scenes", *Journal of Experimental Psychology* Vol. 97, No. 1, 1973, pp. 22-27.

[27] I. Biederman, "On the Semantics of a Glance at a Scene", *Perceptual Organization* (M. Kubovy and J.R. Pomerants, Eds), Erlbaum, 1981.

[28] I. Biederman, "Scene Perception: A Failure to Find Benefit from Prior Expectancy or Familiarity," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, Vol. 9, No. 3, 1983, pp. 411-429.

[29] T. Binford, "Survey of Model-Based Image Analysis Systems," *International Journal of Robotics Research*, **1**, 1982, pp. 18-64.

[30] R. Brachman, "What's in a Concept: Structural Foundations for Semantic Networks," BBN Report 3433, 1976.

[31] M. Brady, "Computational Approaches to Image Understanding," *Computing Surveys*, **14**, March 1982, pp. 3-71.

[32] R. Brooks, "Symbolic Reasoning Among 3-D Models and 2-D Images," *STAN- CS-81-861*, and **AIM-343**, June 1981, Department of Computer Science, Stanford University.

[33] J.B. Burns, A. Hanson, and È. Riseman, "Extracting Linear Features," *Proc. 7th ICPR*, Montreal, 1984. Also COINS Technical Report 84-29, University of Massachusetts at Amherst, August 1984.

[34] J.B. Burns, A.R. Hanson, and E.M. Riseman, "Extracting Straight Lines," *IEEE Transactions on Pattern Analysis and Machine Intelligence 8*, No. 4, July 1986, 425-455.

[35] J. Burrill, "Speeding Up the Weymouth-Overton Smoothing Operator," COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1986.

[36] J. Callahan and R. Weiss, "A Model for Describing Surface Shape", *Proc. Computer Vision and Pattern Recognition*, San Francisco, June 1985, pp. 240-245.

[37] D.D. Corkill and V.R. Lesser, "A Goal-Directed Hearsay-II Architecture: Unifying Data- and Goal-Directed Control", COINS Technical Report 81-15, University of Masschusetts at Amherst, June 1981.

[38] A.P. Dempster, "A Generalization of Bayesian Inference," *Journal of the Royal Statistical Society*, Series B, Vol. 30, 1968, pp. 205-247.

[39] B. Draper, A. Hanson and E. Riseman, "A Software Environment for High Level Vision", COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1986.

[40] R. Duda, P. Hart, and N. Nilsson, "Subjective Bayesian Methods for Rule-Based Inference Systems", National Computer Conference, *AFIPS Conference Proc.* Vol. 45, 1976, pp. 1075-1082.

[41] L. Erman, et al., "The Hearsay–II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty," *Computing Surveys*, **12(2)**, June 1980, pp. 213-253.

[42] O. Faugeras and K. Price, "Semantic Descriptions of Aerial Images Using Stochastic Labeling." *IEEE Transactions on Pattern Analysis and Machine Intelligence 3*, November 1981, pp. 638-642.

[43] Caxton C. Foster, "Content Addressable Parallel Processors", Van Nostrand Reinhold, New York, 1976.

[44] T. Garvey, J. Lowrance, and M. Fischler, "An Inference Technique for Integrating Knowledge From Disparate Sources", *Proc. Seventh IJCAI*, 1981, pp. 319-325.

[45] D.B. Gennery, "Modelling the Environment of an Exploring Vehicle by Stereo Vision", Ph.D. thesis, Stanford AI Laboratory, June 1980.

[46] F. Glazer, G. Reynolds, and P. Anandan, "Scene Matching by Hierarchical Correlation," *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 1983, pp. 432-440.

[47] F. Glazer, "Hierarchical Motion Detection", forthcoming Ph.D. Dissertation, Computer and Information Science Department, University of Massachusetts, Amherst, 1986.

[48] J. Glicksman, "A Cooperative Scheme for Image Understanding Using Multiple Sources of Information", Ph.D. Dissertation, University of British Columbia, November 1982.

[49] J. Griffith, A. Hanson, E. Riseman and R. Kohler, "A Rule-Based Image Segmentation System," COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1987.

[50] W.E.L. Grimson, "On the Reconstruction of Visible Surfaces", in *Image Understanding 1984*, S. Ullman and W. Richards (Eds.), Ablex Publishing Co., New Jersey, 1984, pp. 195-223.

[51] A.R. Hanson, and E.M. Riseman, "Preprocessing Cones: A Computation Structure for Scene Analysis," COINS Technical Report 74C-7, University of Massachusetts at Amherst, September 1974.

[52] A.R. Hanson, and E.M. Riseman, "The Design of a Semantically Directed Vision Processor (Revised and Updated)," COINS Technical Report 75-C1, University of Massachusetts at Amherst, September 1975.

[53] A.R. Hanson and E.M. Riseman, "Design of VISIONS: Segmentation and Interpretation of Images," *Conference Record of 1976 Joint Workshop on PR and AI,* Hyannis, Massachusetts, June 1976, pp. 135-144.

[54] A.R. Hanson and E.M. Riseman (Eds.), *Computer Vision Systems*, New York, Academic Press, 1978.

[55] A. Hanson, and E. Riseman, "VISIONS: A Computer System for Interpreting Scenes, in *Computer Vision Systems* (A. Hanson and E. Riseman, eds.) pp. 303-333, Academic Press, 1978.

[56] A. Hanson and E. Riseman, "Segmentation of Natural Scenes," in *Computer Vision Systems,* (A. Hanson and E. Riseman, Eds.), Academic Press 1978, pp. 129-163.

[57] A. Hanson and E. Riseman, "Processing Cones: A Computational Structure for Image Analysis," in *Structured Computer Vision*, (S. Tanimoto and A. Klinger, Eds.), Academic Press, New York, 1980.

[58] A.R. Hanson and E.M. Riseman, "A Summary of Image Understanding Research at the University of Massachusetts," COINS Technical Report 83-35, University of Massachusetts at Amherst, October 1983.

[59] A.R. Hanson, E.M. Riseman, and P.A. Nagin, Authors Reply to "Image Segmentation: A Comment on 'Studies in Global and Local Histogram-Guided Relaxation Algorithms'", *PAMI-6, No.2*, March 1984.

[60] A.R. Hanson, E.M. Riseman, J.S. Griffith and T.E. Weymouth, "A Methodology for the Development of General Knowledge-Based Vision Systems", *Proc. of IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, CO, December 1984, pp. 159-170.

[61] A.R. Hanson and E.M. Riseman, "A Methodology for the Development of General Knowledge-Based Vision Systems," to appear in *Vision, Brain, and Cooperative Computation,* (M. Arbib and A. Hanson, Eds.) 1986, MIT Press, Cambridge, MA.

[62] R.M. Haralick, "Ridges and Valleys on Digital Images," *Computer Vision, Graphics and Image Processing,* 22(1), 1983, pp. 28-39.

[63] M. Herman and T. Kanade, "The 3D MOSAIC Scene Understanding System: Incremental Reconstruction of 3D Scenes from Complex Images", *Proceedings of the DARPA IU Workshop,* October 1984, pp. 137-148.

[64] B.K.P. Horn, "Obtaining Shape from Shading Information," in *The Psychology of Computer Vision,* P.H. Winston (Ed.), McGraw-Hill, New York, 1975.

[65] B.K.P. Horn, "Understanding Image Intensitites," *Artificial Intelligence,* Vol. 8, 1977, pp. 201-231.

[66] B.K.P. Horn and B.A. Schunck, "Determining Optical Flow," *Artificial Intelligence,* Volume 17, 1981, pp. 185-203.

[67] R.A. Hummel, and M. Landy, "A Statistical Viewpoint on the Theory of Evidence", New York University, Courant Institute of Mathematical Sciences, Technical Report No. 194, December 1985.

[68] K. Ikeuchi, "Numerical Shape for Shading and Occluding Contours in a Single View", AI Memo 566, AI Lab, MIT, February 1980.

[69] M. Jenkin, "The Stereopsis of Time Varying Imagery", RBVC Technical Report, RBVC-TR-84-3, Department of Computer Science, University of Toronto, 1984.

[70] T. Kanade, "Model Representation and Control Structures in Image Understanding," *Proc. IJCAI-5,* August 1977.

[71] J.R. Kender, "Shape from Texture", Ph.D. Dissertation, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA 1980.

[72] C. Kohl, Ph.D. Dissertation, Deparment of Computer and Information Science, University of Massachusetts at Amherst, in preparation, 1987.

[73] R.R. Kohler, "A Segmentation System Based on Thresholding," *Computer Graphics and Image Processing,* **15,** 1981, pp. 319-338.

[74] R.R. Kohler and A.R. Hanson, "The VISIONS Image Operating System," *Proc. of 6th International Conference on Pattern Recognition,* Munich, Germany, October 1982.

[75] R.R. Kohler, "Integrating Non-Semantic Knowledge into Image Segmentation Processes," Ph.D. Thesis, University of Massachusetts at Amherst, September 1983. Also COINS Technical Report 84-04, University of Massachusetts at Amherst, March 1984.

[76] H.E. Kyberg, "Bayesian and Non-Bayesian Evidential Updating", University of Rochester, Department of Computer Science Technical Report 139, July 1984.

[77] D.T. Lawton, "Processing Translational Motion Sequences," *Computer Graphics and Image Processing,* Vol. 22, pp. 116-144, 1983.

[78] D.T. Lawton, "Processing Dynamic Image Sequences from a Moving Sensor," Ph.D. Dissertation (COINS Technical Report 84-05), Computer and Information Science Department, University of Massachusetts, 1984.

[79] N. Lehrer, G. Reynolds, and J. Griffith, "A Method for Initial Hypothesis Formation in Image Understanding", COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1987.

[80] V.R. Lesser, R.D. Fennell, L. D. Erman, and D.R. Reddy, "Organization of the Hearsay-II Speech Understanding System," *IEEE Trans. on ASSP 23*, 1975, pp. 11-23.

[81] V.R. Lesser and L.D. Erman, "A Retrospective View of the Hearsay-II Architecture," *Proc. IJCAI-5*, 1977, pp. 790-800, Cambridge, MA.

[82] M. Levine and S. Shaheen, "A Modular Computer Vision System for Picture Segmentation and Interpretation," *IEEE Transactions on Pattern Analysis and Machine Intelligence 3*, September 1981, pp. 540-556.

[83] S.P. Levitan, "Parallel Algorithms and Architectures: A Programmers Perspective," Ph.D. Dissertation, Computer and Information Science Department, also, COINS Technical Report 84-11, University of Massachusetts at Amherst, May 1984.

[84] S.P. Levitan, C. Weems, A.R. Hanson, and E.M. Riseman, "The UMass Image Understanding Architecture", to appear in *"Pyramid Multi-Computers"*, (Leonard Uhr, Ed.), Academic Press, New York, 1987.

[85] J. Lowrance, "Dependency Graph Models of Evidential Support," Ph.D. Thesis, Computer and Information Science Department, also, COINS Technical Report 82-26, University of Massachusetts at Amherst, 1982.

[86] J. Lowrance, and T. Garvey, "Evidential Reasoning: A Developing Concept", *Proc. International Conference on Cybernetics and Society*, October, 1982.

[87] D. Marr, and E. Hildreth, "Theory of Edge Detection," *Proc. of the Royal Society of London,* B., 207, 1980, pp. 187-217.

[88] D. Marr, *VISION*, W.H. Freeman and Company, San Francisco, 1982.

[89] D.M. McKeown, W.A. Harvey and J. McDermott, "Rule Based Interpretation of Aerial Imagery," Department of Computer Science, Carnegie-Mellon University, September 1984.

[90] D.I. Moldovan, J.G. Wu, S. Levitan, and C. Weems, "Parallel Processing of Iconic to Symbolic Transformation of Images", *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1985, pp. 257-264.

[91] H.P. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover", Ph.D. Thesis, Stanford University AI Laboratory, California, September 1980.

[92] M. Nagao and T. Matsuyama, "Edge Preserving Smoothing," *Proc. of the Fourth International Joint Conference on Pattern Recognition,* pp. 518-520, November 1978.

[93] M. Nagao and T. Matsuyama, "A Structural Analysis of Complex Aerial Photographs," Plenum Press, New York, 1980.

[94] P.A. Nagin, "Studies in Image Segmentation Algorithms Based on Histogram Clustering and Relaxation," COINS Technical Report 79-15, University of Massachusetts at Amherst, September 1979.

[95] P.A. Nagin, A.R. Hanson, and E.M. Riseman, "Studies in Global and Local Histogram-Guided Relaxation Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 3* May 1982, pp. 263-277.

[96] H. Nakatani, et. al. "Extraction of vanishing point and its application to scene analysis based on image sequence," *5th Int. Conf. on Pattern Recognition,* pp. 360-372, 1980.

[97] H. Nakatani, T. Kitahashi, "Inferring 3-d shape from line drawings using vanishing points," *1st Intn'l Conf. on Computers and Applications,* 1984.

[98] H. Nakatani, R. Weiss, and E. Riseman, "Application of Vanishing Points to 3D Measurement," *Proc. SPIE,* Vol. 507, 1984, pp. 164-169.

[99] A.M. Nazif and M.D. Levine, "Low Level Segmentation: An Expert System," Technical Report 83-4, April 1983, Electrical Engineering, McGill University.

[100] A.M. Nazif, "A Rule-based Expert System for Image Segmentation," Ph.D. Dissertation, E.E. Dept., McGill U., 1983.

[101] R. Nevatia and K.R. Babu, "Linear Feature Extraction and Description," *Computer Graphics and Image Processing*, Vol. 13, 1980, pp. 257-269.

[102] H. Penny Nii, "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures", in *AI Magazine*, pp. 38-53, Vol. 7, Number 2 (Summer 1986).

[103] R. Ohlander, K. Price, and D.R. Reddy, "Picture Segmentation Using a Recursive Region Splitting Method," *Computer Graphics and Image Processing 8*, Volume 3, 1979.

[104] Y. Ohta, "A Region-Oriented Image-Analysis System by Computer," Ph.D. Thesis, Computer Information Science Department, Kyoto University, Kyoto, Japan, 1980.

[105] K. Overton and T.E. Weymouth, "A Noise Reducing Preprocessing Algorithm," *Proc. IEEE Conference on Pattern Recognition and Image Processing,* Chicago, IL, 1979, pp. 498-507.

[106] C.C. Parma, A.R. Hanson and E.M. Riseman, "Experiments in Schema-Driven Interpretation of a Natural Scence," COINS Technical Report 80-10, University of Massachusetts at Amherst, April 1980. Also in *NATO Advanced Study Institute on Digital Image Processing* (R. Haralick and J.C. Simon, Eds.), Bonas, France, 1980.

[107] J.L. Paul, "An Image Interpretation System", Ph.D. Dissertation, University of Sussex, June 1977.

[108] I. Pavlin, A. Hanson, and E. Riseman, "Analysis of an Algorithm for Detection of Translational Motion," *Proc. DARPA IU Workshop,* Miami Beach, FL, December 1985.

[109] J.M. Prager, "Segmentation of Static and Dynamic Scenes", Ph.D. Thesis, Computer and Information Science Dept., University of Massachusetts at Amherst, 1979.

[110] K.E. Price, "Change Detection and Analysis in Multispectral Images," Ph.D. Thesis, Carnegie-Mellon University, Pittsburgh, PA, December 1976.

[111] K.E. Price and R. Reddy, "Matching Segments of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence 1,* June 1979.

[112] K.E. Price, "Image Segmentation: A Comment on 'Studies in Global and Local Histogram-Guided Relaxation Algorithms'", *IEEE Transactions on Pattern Analysis and Machine Intelligence 6,* Volume 2, March 1984.

[113] G. Reynolds, D. Strahman, N. Lehrer, L. Kitchen, "Plausible Reasoning and the Theory of Evidence", COINS Technical Report 86-11, University of Massachusetts at Amherst, April 1986.

[114] G. Reynolds, J. Ross Beveridge, "Geometric Line Organization Using Spatial Relations and a Connected Components Algorithm", COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1986.

[115] G. Reynolds, N.Irwin, A. Hanson and E. Riseman, "Hierarchical Knowledge- Directed Object Extraction Using a Combined Region and Line Representation," *Proc. of the Workshop on Computer Vision: Representation and Control,* Annapolis, Maryland, April 30 - May 2, 1984, pp. 238-247.

[116] G. Reynolds, D. Strahman, N. Lehrer, "Converting Feature Values to Evidence," *Proc. DARPA IU Workshop,* Miami Beach, FL, 1985.

[117] E. Riseman and A. Hanson, "The Design of a Semantically Directed Vision Processor," COINS Technical Report 74C-1, University of Massachusetts, February 1975, Revised version COINS Technical Report 75C-1.

[118] E.M. Riseman and A.R. Hanson, "A Methodology for the Development of General Knowledge-Based Vision Systems," *IEEE Proc. of the Workshop on Computer Vision: Representation and Control,* 1984, pp. 159-170.

[119] L.G. Roberts, "Machine Perception of Three-Dimensional Solids", *Symposium of Optical and Electro-optical Information Processing Technology,* Boston, 1964, pp. 159-197.

[120] Steven A. Shafer, Antohony Stentz, and Charles E. Thorpe, "An Architecture for Sensor Fusion in a Mobile Robot", *International Conference on Robotics and Automation*, San Francisco, California, 1986, pp. 2202-2011.

[121] G. Shafer, "A Mathematical Theory of Evidence," Princeton University Press, 1976.

[122] Y. Shirai, "Recognition of Man-Made Objects Using Edge Cues," in *Computer Vision Systems* (A. Hanson and E. Riseman, Eds), Academic Press, New York, 1976.

[123] K. Sloan, "World Model Driven Recognition of Natural Scenes," Ph.D. Dissertation, Moore School of Electrical Engineering, University of Pennsylvania, 1977.

[124] R. Southwick, "FEATSYS - An Intermediate-Level Representation of Image Feature Data," Master's Thesis, Computer and Information Science Department, University of Massachusetts at Amherst, February 1986.

[125] M. Snyder, "The Accuracy of 3D Parameters in Correspondence-Based Techniques," *Proc. of the Workshop on Motion: Representation and Analysis,* Charleston, S.C., May 1986; also COINS Technical Report 86-28, University of Masschusetts at Amherst, July 1986.

[126] M. Snyder, "A Comparison of the Accuracy of Depth Recovery from Motion and Stereo," in progress, 1986.

[127] S. Tanimoto and A. Klinger, *Structured Computer Vision*, Academic Press, New York, 1980.

[128] J.M. Tenenbaum and H. Barrow, "Experiments in Interpretation-Guided Segmentation" Technical Note 123, AI Center, Stanford Research Institute, 1976; also in *AI Journal*, Vol. 8, No. 3, 1977, pp. 241-274.

[129] J.M. Tenenbaum, H.G. Barrow, "Recovering intrinsic scene characteristics from images," in *Computer Vision Systems,* (A. Hanson and E. Riseman Eds.), New York: Academic Press, 1978, pp. 3-26.

[130] D. Terzopolous, "Multi-Level Reconstruction of Visual Surfaces", AI Memo, 6712, MIT AI Lab, Cambridge, MA; also in Rosenfeld [1983].

[131] A. Triesman, "Features and Objects in Visual Processing", Scientific American, November 1986, pp. 114-125.

[132] A. Triesman, "Properties, Parts, and Objects", in *Handbook of Perception and Performance*, (K. Boff, L. Kaufman, and J. Thompson, Eds.), John Wiley & Sons, 1986.

[133] J. Tsotsos, "Knowledge of the Visual Process: Content, Form and Use," *Proc of the 6th International Conference on Pattern Recognition*, October 1982, pp. 654-669.

[134] J.K. Tsotsos, "Representational Axes and Temporal Cooperative Processes," Technical Report RCVB-TR-84-2, University of Toronto, April 1984.

[135] S. Ullman, "The Interpretation of Visual Motion," The MIT Press, Cambridge, MA 1979.
S. Ullman, "Visual Routines" in *Visual Cognition*, (Steven Pinker, Ed.), MIT Press, Cambridge, MA, 1985.

[136] C. Weems, S. Levitan, D. Lawton, and C. Foster, "A Content Addressable Array Parallel Processor and Some Applications," *Proc. DARPA IU Workshop*, Arlington, VA, June 1983.

[137] C. Weems, "Image Processing on a Content Addressable Array Parallel Processsor", Ph.D. Dissertation and COINS Technical Report 84-14, University of Massaschusetts at Amherst, September 1984.

[138] C. Weems, S. Levitan, C. Foster, E. Riseman, D. Lawton, A. Hanson, "Development and Construction of a Content Addressable Array Parallel Processor (CAAPP) for Knowledge-Based Image Interpretation," *Proc. Workshop on Algorithm- Guided Parallel Architectures for Automatic Target Recognition,* Leesburg, VA July 16-18, 1984, pp. 329-359.

[139] R. Weiss and M. Boldt, "Geometric Grouping Applied to Straight Lines", *Proceedings on the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June, 1986, pp. 489-495.

[140] R. Weiss, A. Hanson, and E. Riseman, "Geometric Grouping of Straight Lines," *Proc. 1985, DARPA IU Workshop,* Miami Beach, FL, 1985.

[141] J.S. Weska, "A Survey of Threshold Selection Techniques", *Computer Graphics and Image Processing* 7, pp. 259-265, 1978.

[142] L. Wesley and A. Hanson, "The Use of Evidential-Based Model for Representing Knowledge and Reasoning about Images in the VISIONS System," *Proc. Workshop on Computer Vision,* Rindge, NH, August 23-25, 1982.

[143] L. Wesley, "Reasoning about Control: The Investigation of an Evidential Approach," *Proc. 8th IJCAI,* Karlsrube, West Germany, August 1983, pp. 203-210.

[144] L. Wesley, J. Lowrance, and T. Garvey, "Reasoning About Control: An Evidential Approach", SRI Technical Note #324, Artificial Intelligence Center, SRI International, Menlo Park, California 94025.

[145] L. Wesley, "Evidential knowledge-based computer vision", *Optical Engineering*, Vol. 25, No. 3, March 1986, pp. 363-379.

[146] L. Wesley, "The Application of an Evidential Based Technology to a High-Level Knowledge-Based Image Interpretation Systems", Ph.D. Dissertation, University of Massachusetts, in preparation, 1987.

[147] L. Wesley, and A. Hanson, "Evidential-Reasoning: Its Application to a High-Level Knowledge-Based Image Interpretation System", COINS Technical Report, University of Massachusetts at Amherst, in preparation, 1987.

[148] T.E. Weymouth, J.S. Griffith, A.R. Hanson and E.M. Riseman, "Rule Based Strategies for Image Interpretation," *Proc. of AAAI-83*, August 1983, pp. 429-432, Washington D.C. A longer version of this paper appears in *Proc. of the DARPA Image Understanding Workshop,* June 1983, pp. 193-202, Arlington, VA.

[149] T.E. Weymouth, "Using Object Descriptions in a Schema Network For Machine Vision," Ph.D. Dissertation, Computer and Information Science Department, University of Massachusetts at Amherst. Also COINS Technical Report 86-24, University of Massachusetts at Amherst, 1986.

[150] T.E. Weymouth, A. Hanson, and E. Riseman, "Schema-Based Image Understanding: Interpretation Strategies for Natural Scenes", forthcoming Department of Computer and Information Science Technical Report, 1987.

[151] A.P. Witkin and J.M. Tenenbaum, "On the role of structure in vision", in *Human and Machine Vision*, J. Beck, B. Hope, and A. Rosenfeld (Eds.), Academic Press, New York, 1982.

[152] Lance R. Williams and P. Anandan, "A Coarse-to-Fine Control Strategy for Stereo and Motion on a Mesh-Connected Computer", COINS Technical Report 86-19, University of Massachusetts at Amherst, May 1986.

[153] T.D. Williams, "Depth from Camera Motion in a Real World Scene," *IEEE Transactions on Pattern Analysis and Machine Intelligence, 2* Volume 6, November 1980, PP. 511-516.

[154] T.D. Williams, "Computer Interpretation of a Dynamic Image from a Moving Vehicle," Ph.D. Thesis and COINS Technical Report 81-22, University of Massachusetts at Amherst, May 1981.

[155] A. Witkin, "Scale-Sapce Filtering", *Proceedings of IJCAI*, Karlsruhe, 1983, pp. 1019-1021.

[156] A. P. Witkin and J.M. Tenenbaum, "What Is Perceptual Organiztion For?", *IJCAI*, 1983, pp. 1023-1026.

[157] R.J. Woodham, "Photometric Stereo: A Reflectance Map Technique for Determining Surface Orientation from Image Intensity", *Proc. 22nd Annual SPIE Conference*, San Diego, CA, August 1978, pp. 136-143.

[158] R.J. Woodham, "Photometric Method for Determining Shape from Shading", in *Image Understanding 1984*, S. Ullman and W. Richards (Eds.), Ablex Publishing Co., New Jersey, pp. 97-126.

[159] W.A. Woods, "Theory Formation and Control in a Speech Understanding System with Extrapolation Towards Vision," in *Computer Vision Systems*, (A. Hanson and E. Riseman, Eds.), Academic Press, 1978.

[160] B. York, A. Hanson, and E. Riseman, "3D Object Representation and Matching with B-Splines and Surface Patches," *Proc. IJCAI-7*, August 1980, pp. 648-651.

[161] B. York, "Shape Representation in Computer Vision", COINS Technical Report 81-13, University of Massachusetts at Amherst, May 1981.

(a)

(b)

(c)

Figure 1.1. Original images. These images are representative samples from a larger data base that is used in our experiments. All are 256 x 256 pixels in spatial resolution; the color resolution is 8 bits in each of the red, green, and blue components, except (h), which is monochromatic.

(d)

(e)

(f)

Figure 1.1, continued

(g)

(h)

Figure 1.1, continued

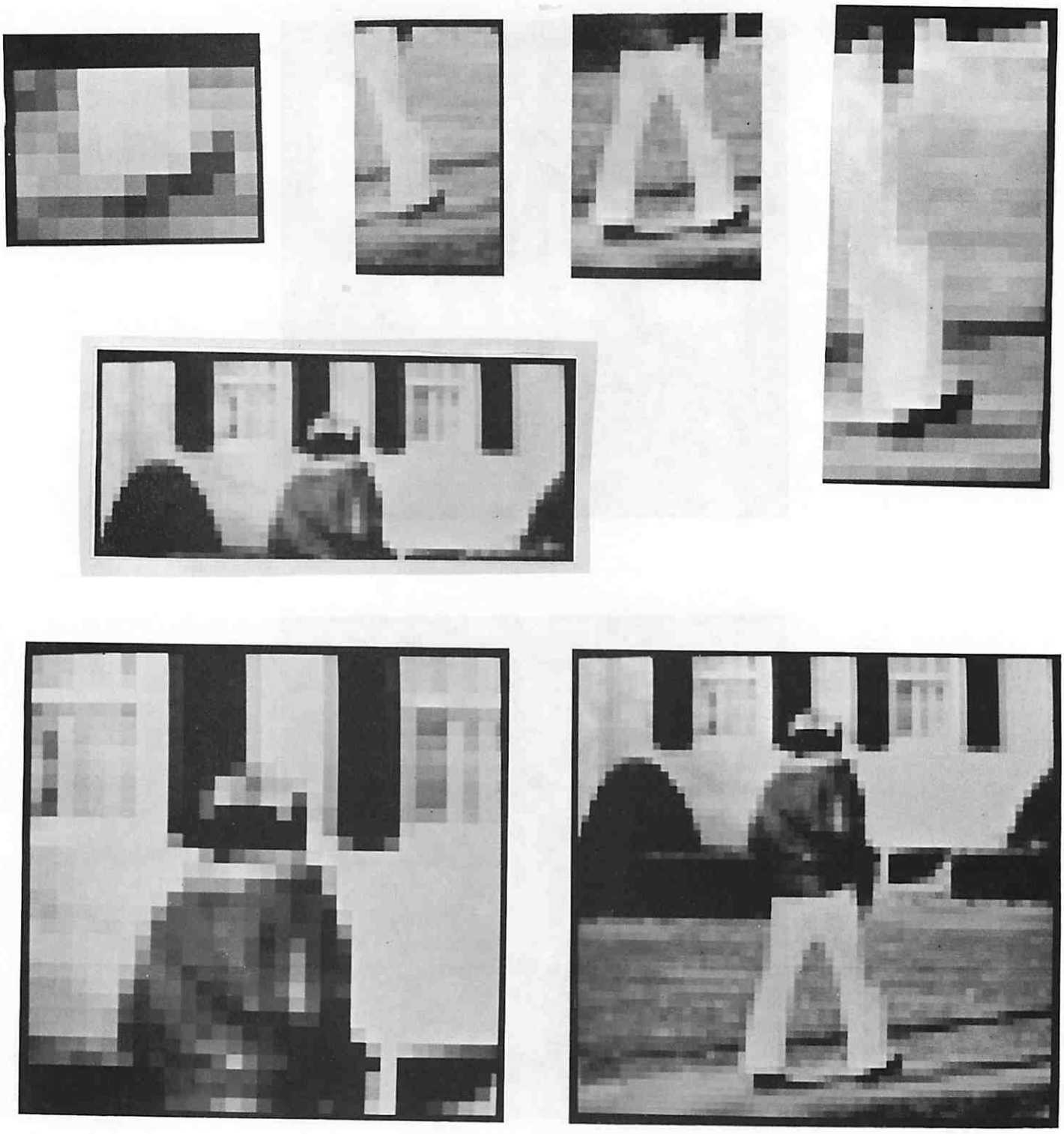


Figure 1.2. Closeup subimages from original images. In many cases, the identity or function of an object or object part cannot be determined from a small local view. Only when the surrounding context becomes available can the objects be recognized.

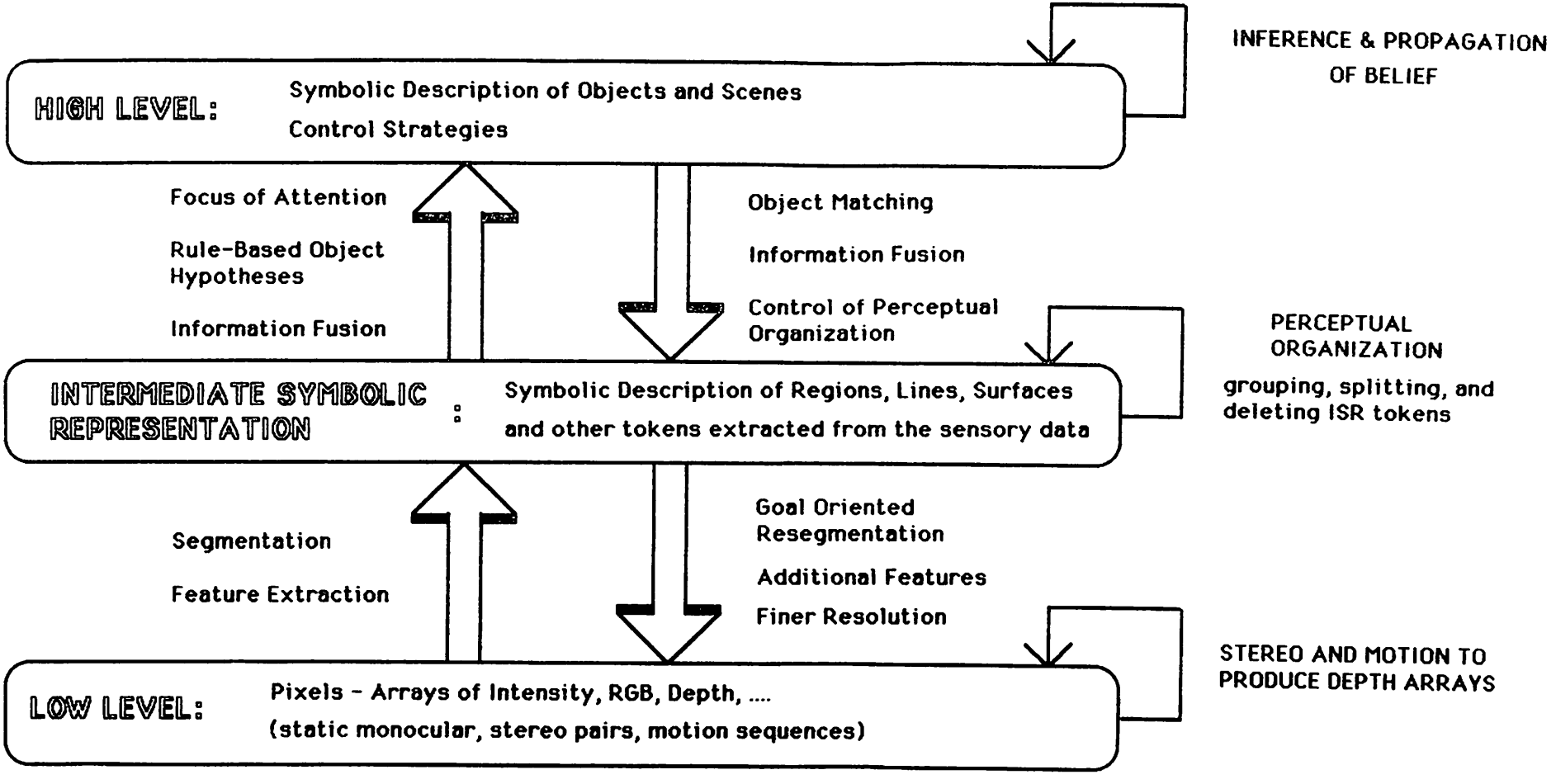
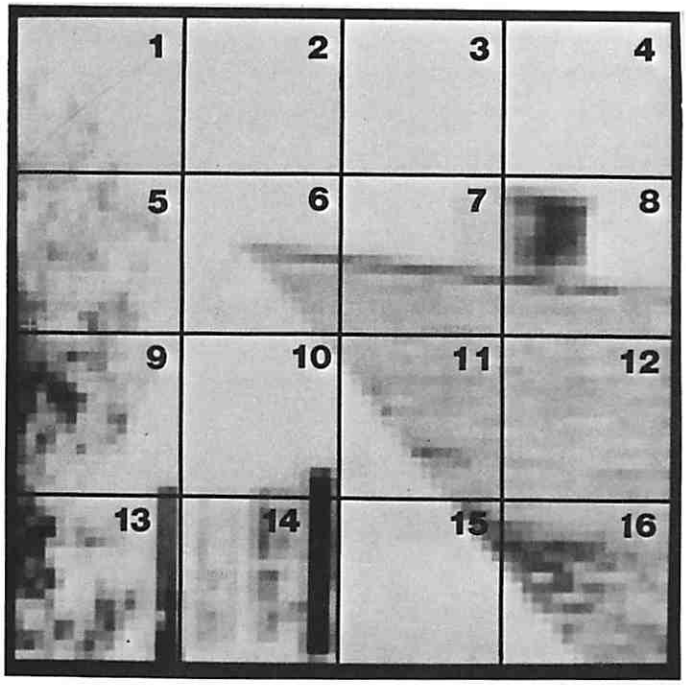
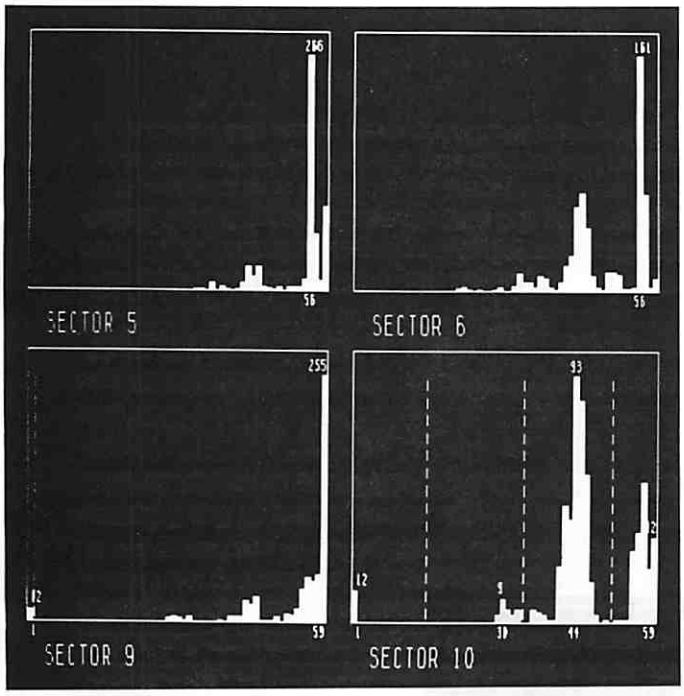


Figure 1.3. The VISIONS System: Processing and control across Multiple levels of representation are depicted in this system overview.

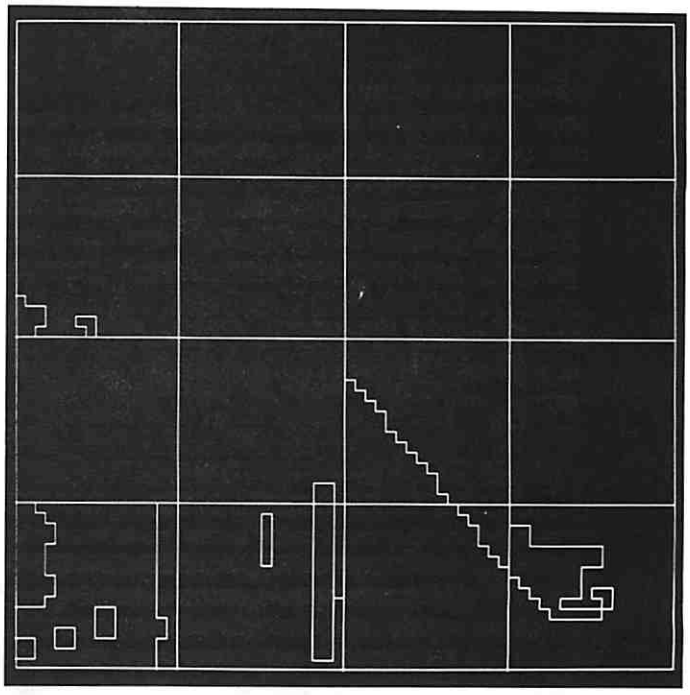


(a)

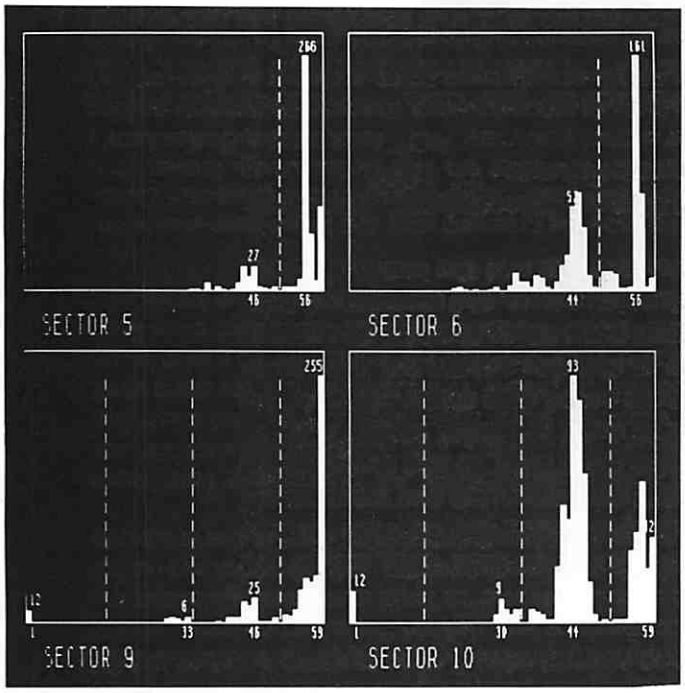
Figure 3.1 Histogram-Based Region Segmentation. (a) Original image from roof/chimney area of Figure 1.1(c) showing the sector boundaries. (b) Histograms of the four adjacent sectors 5,6,9, and 10. Peaks and valleys are extracted, and labels are associated with peaks; the dotted lines mark the valleys and the numbers denote the size of the peaks. (c) Region boundaries formed by mapping the histogram labels to pixels and finding connected components. (d) Augmented cluster set after propagating peaks from adjacent sectors. (e) Region boundaries produced by the augmented cluster set. (f) The original subimage and region boundaries during the three peak propagation steps; after two iterations of peak propagations, no further peaks are added to any sector and the process terminates. Note that the sensitivity in the peak extraction process was not sufficient to detect the sky-wall boundary.



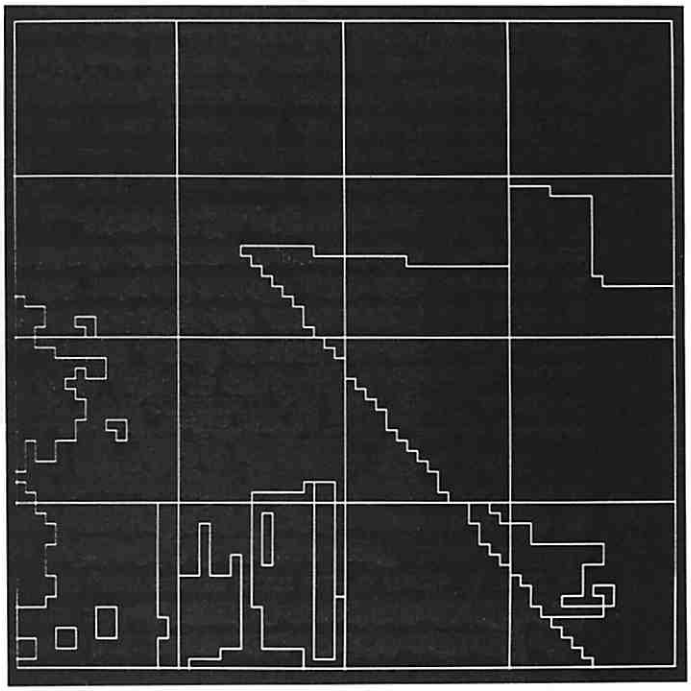
(b)



(c)

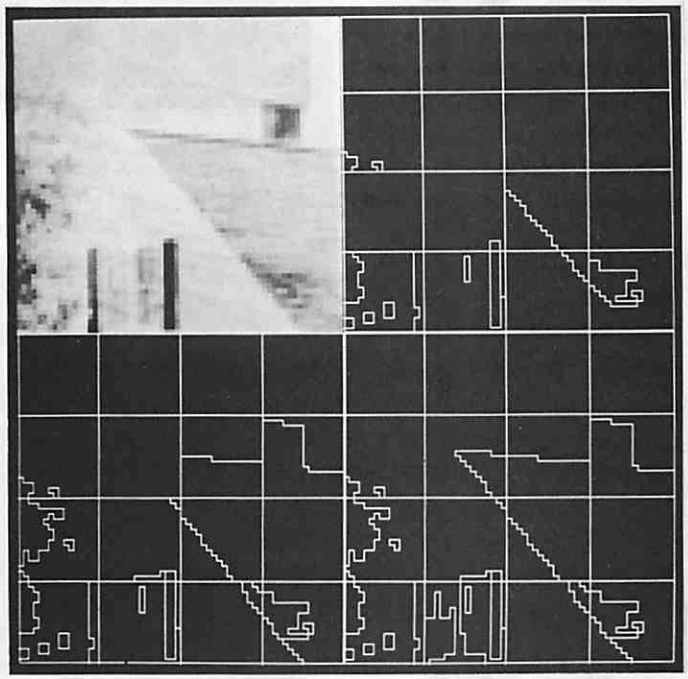


(d)



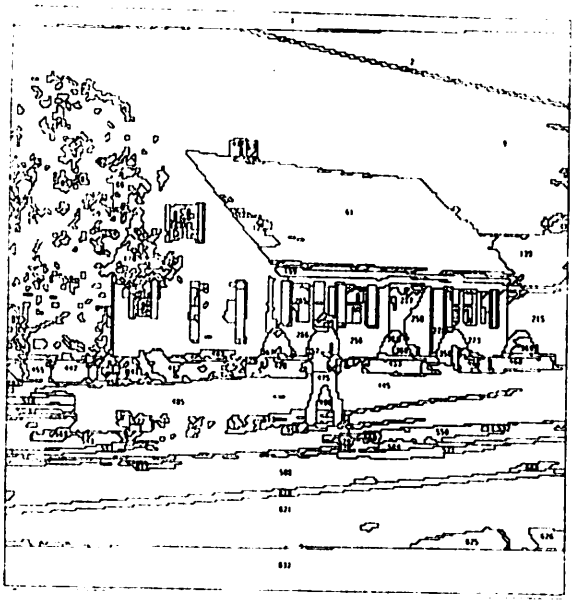
(e)

Figure 3.1 continued



(f)

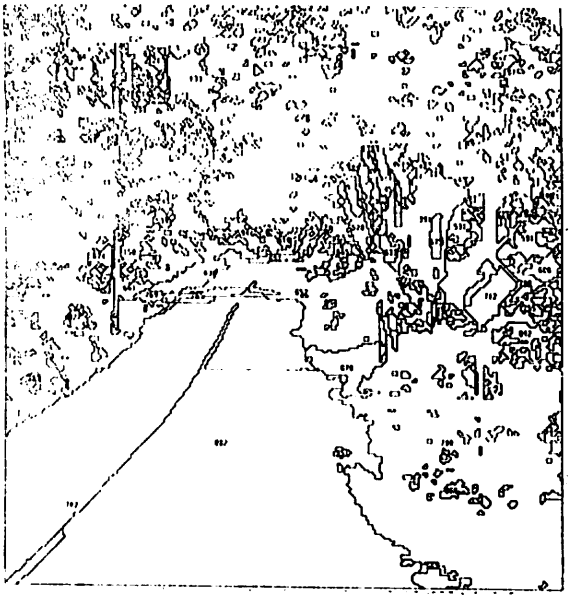
Figure 3.1 continued



(a)

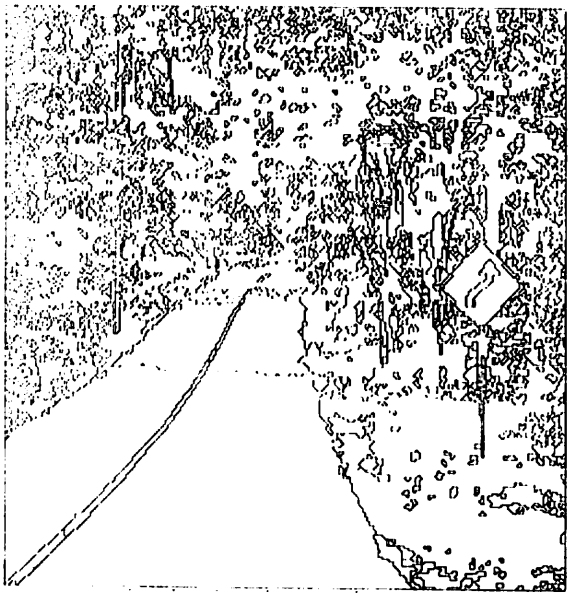
(b)

Figure 3.2 Results of Histogram-Based Region Segmentation. The algorithm has a sensitivity parameter controlling the peak extraction process and the resulting number of regions produced. In each case the parameter was set to "high" or "very high" sensitivity for each of the 16 x 16 sector subimages; (a) High sensitivity intensity segmentation; (b) High sensitivity segmentation of the red component (R from RGB); (c) Intensity segmentation at very high sensitivity; (d) Red color plane segmentation at "very high" sensitivity; (e) Intersection of R,G, and B region segmentations at very high sensitivity (or equivalently the union of R,G, and B boundaries). Note the recovery of additional interesting boundaries beyond the intensity segmentation at the expense of additional fragmentation across the image.



(c)

(d)



(e)

Figure 3.2, continued

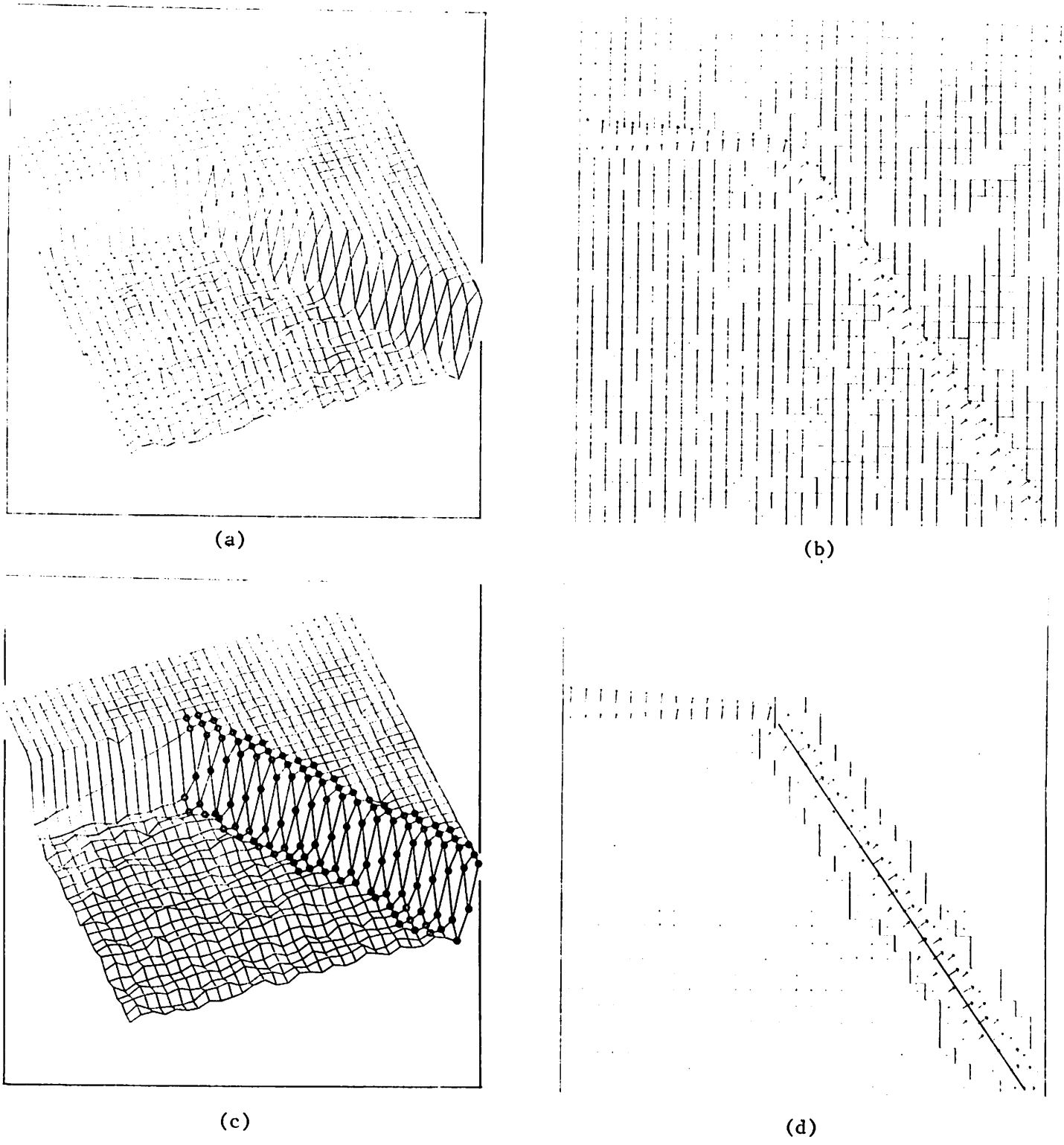
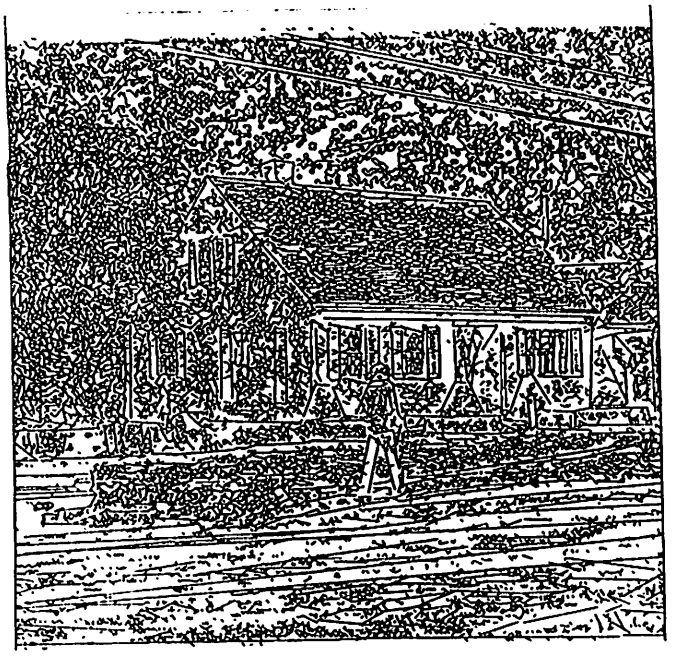
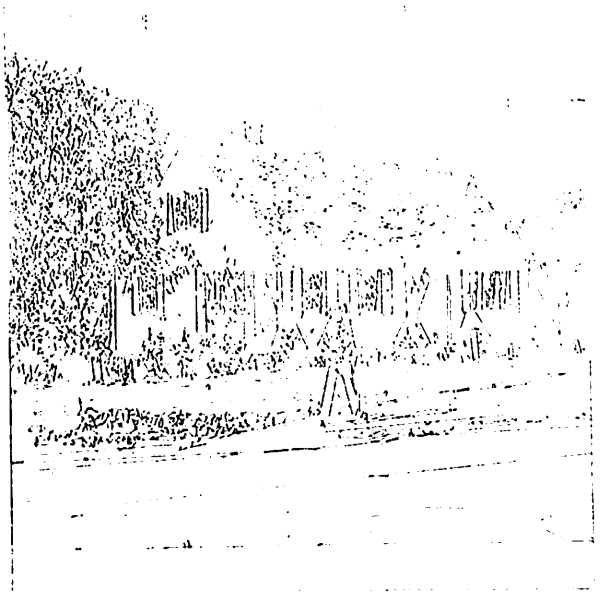


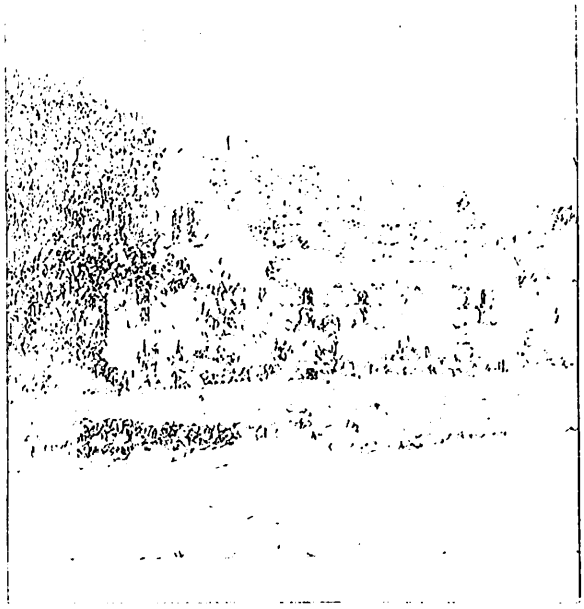
Figure 3.3 Line Extraction from Gradient Orientation. (a) Surface plot of a relatively sharp and high contrast edge; (b) Regions produced by a connected components algorithm applied to the labels of the orientation partitions; (c) Pixels included in the line support region are highlighted by dots; (d) The resulting straight line overlaid on the set of pixels making up the line support region.



(a)

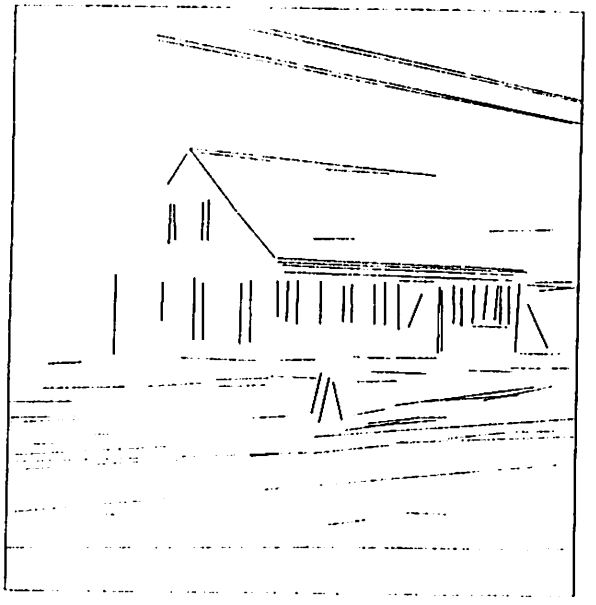


(b)

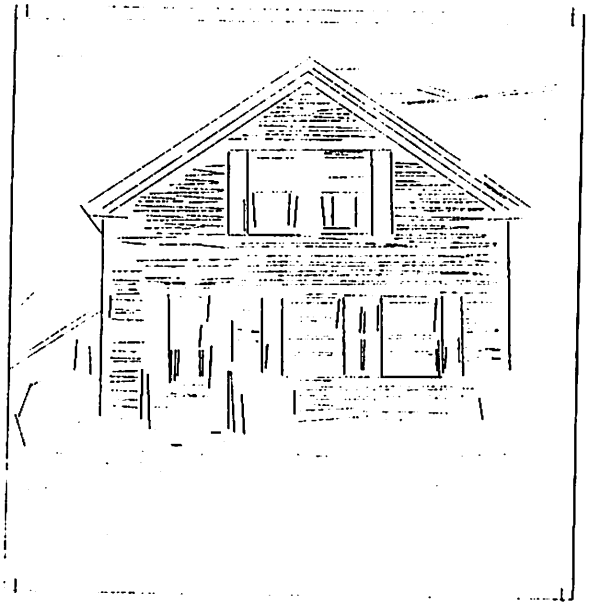


(c)

Figure 3.4 Results of Line Extraction Algorithm. (a) Initial line set; (b-e) show the results of filtering the initial line set on various attributes of the lines; (b) High gradient magnitude lines with gradient $\geq$ 10 gray levels per pixel; (c) Short high gradient magnitude lines with length $\leq$ 5 and gradient $\geq$ 10 gray levels per pixel; (d) Long high gradient lines with length $\geq$ 15 and gradient $\geq$ 10 gray levels per pixel; (e) Lines with length $\geq$ 10; (f,g) Two road scenes.

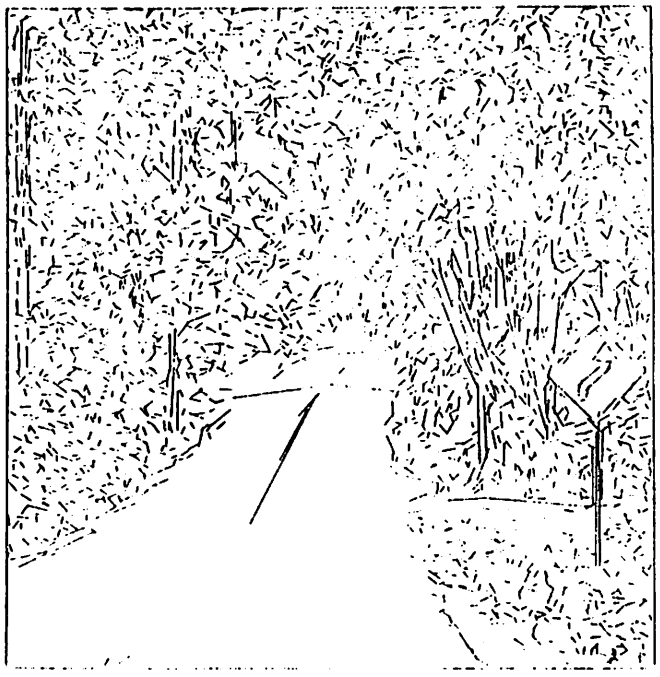


(d)

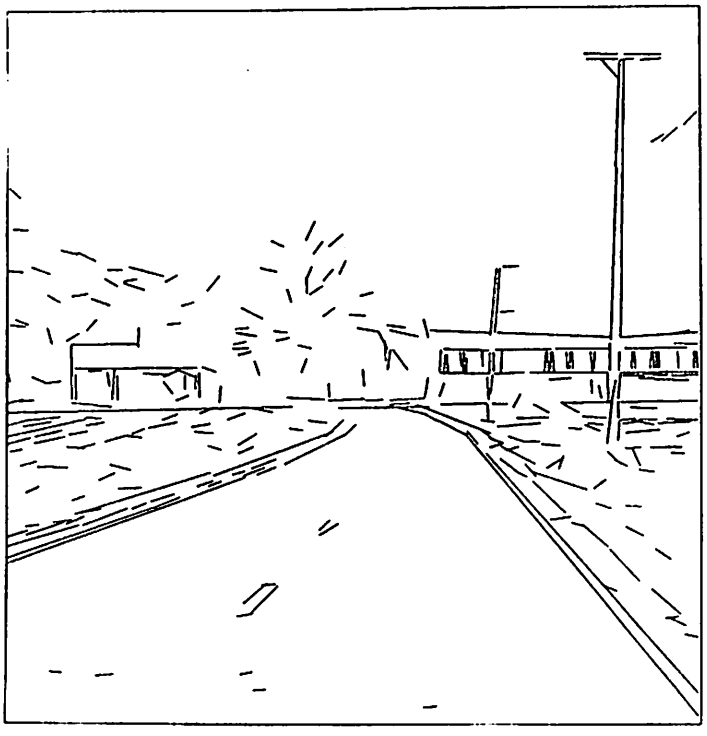


(e)

Figure 3.4, continued



(f)



(g)

Figure 3.4, continued

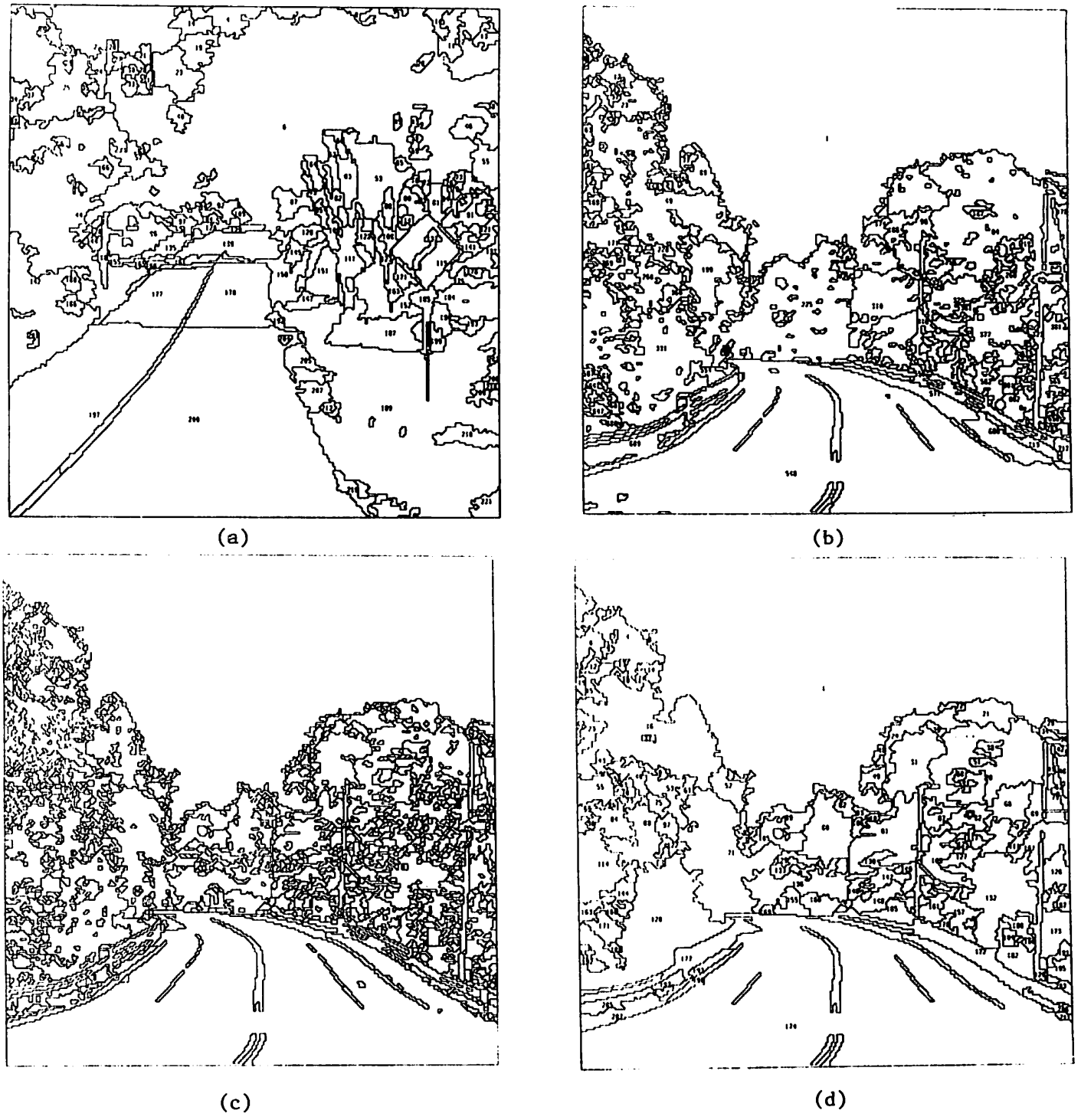
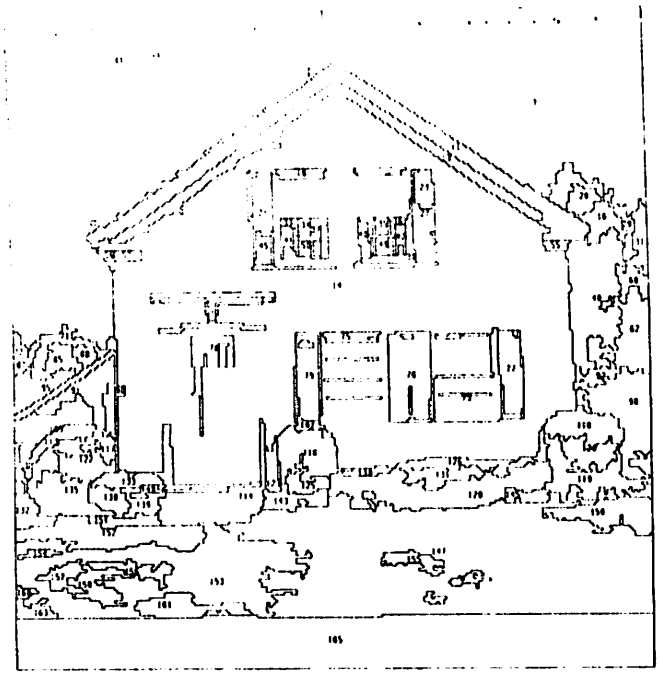
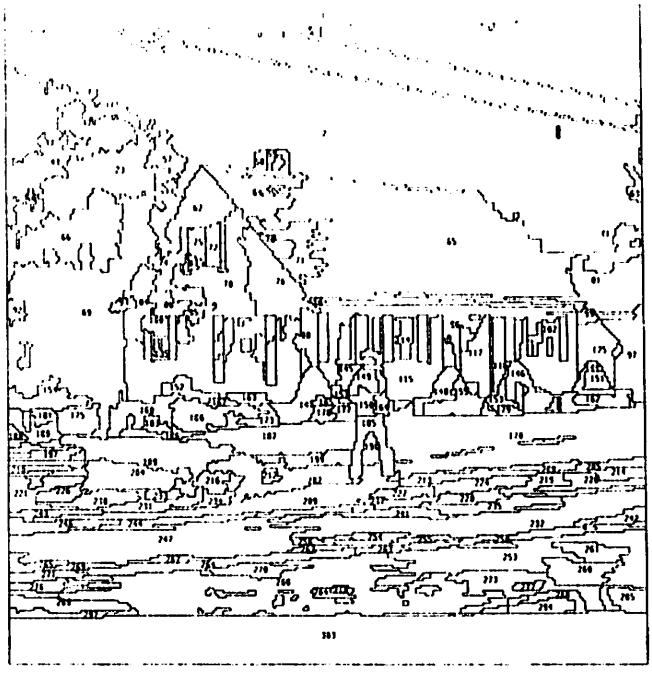


Figure 3.5 Region Merging. The result of applying rule-based region merging to the output of the histogram-based region segmentation algorithm. (a) After region merging applied to Fig. 3.2(e) which is the RGB region segmentation; this result should also be compared to both Figure 3.2(c) and Figure 3.2(e); (b) Intensity segmentation at high sensitivity of the other road scene in Fig. 1.1; (c) Three-color intersection of RGB region segmentations at high sensitivity; (d) After region merging; note that portions of the road boundary and telephone pole detail are preserved with relatively few regions in the segmentation.



(a)

(b)



(c)

Figure 5.1 Region Segmentations for Interpretation Experiments. Each of these images was produced by the histogram-based region segmentation algorithm at high sensitivity, followed by the region merging algorithm to reduce the number of regions.

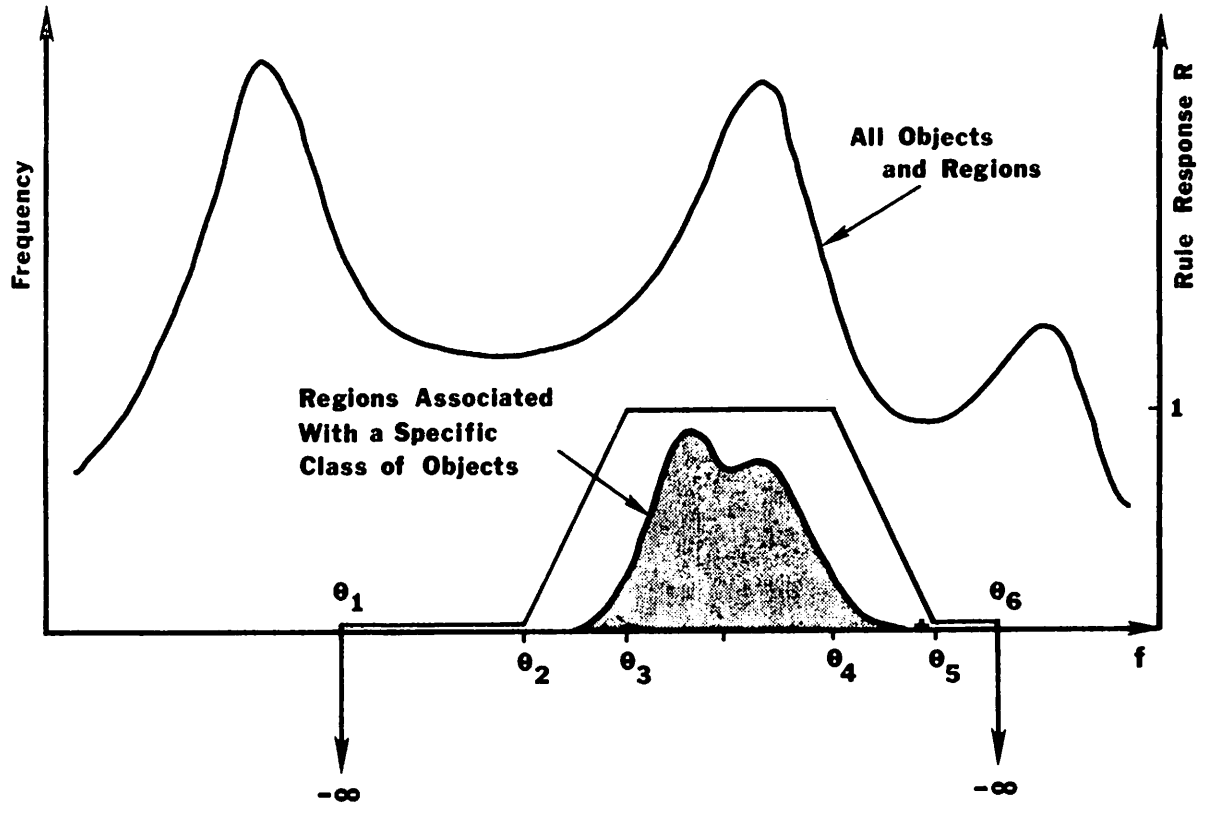
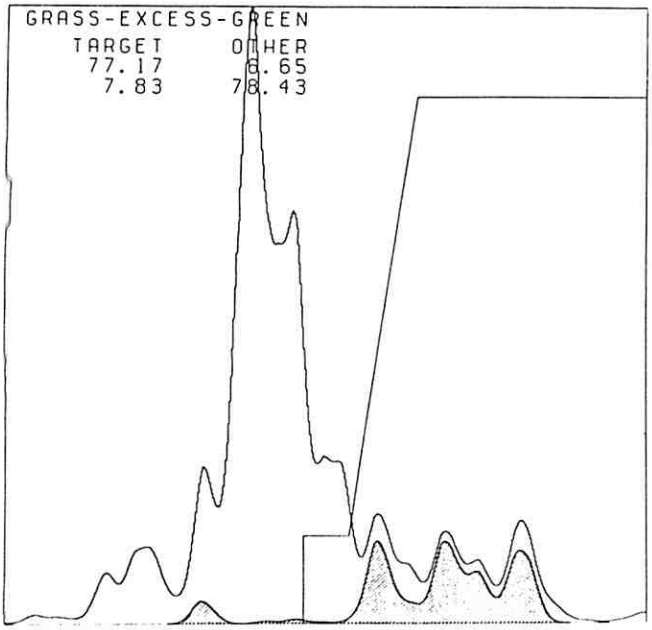
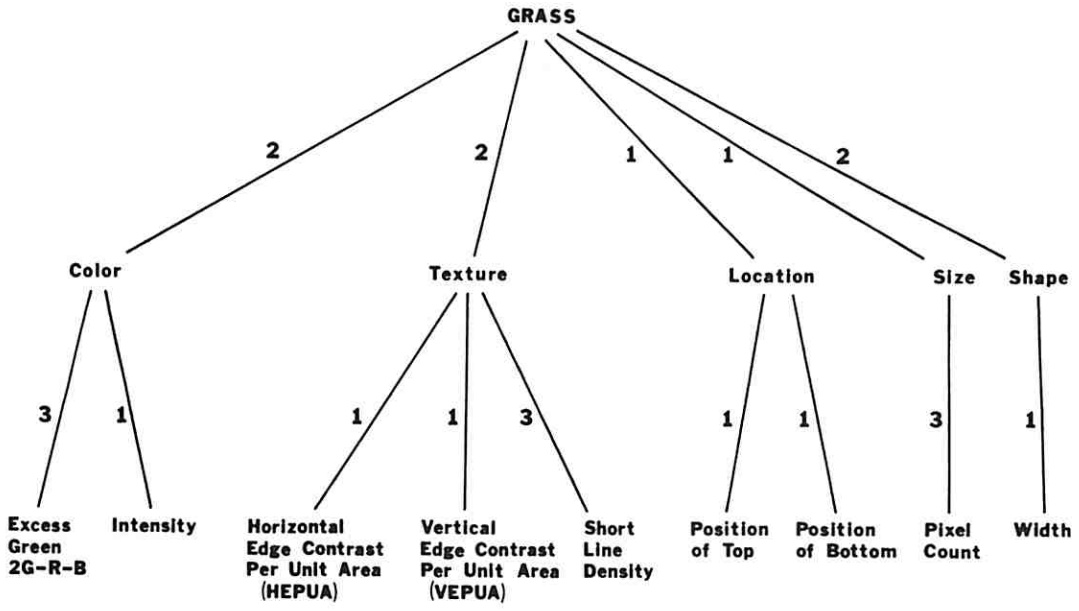


Figure 5.2. A Simple Rule as a Constraint on a Token Attribute. The two histograms obtained from a sample of hand-labelled images show the frequency of feature values of regions for all objects and the frequency of feature values of regions associated with a specific class of objects. The simple rule is shown as a piecewise-linear function mapping feature value f_I into a response R (0-1 range). The object-specific mapping is parameterized by six threshold values and stored in thed knowledge network.



(a)

(b)



(c)

Figure 5.3 Example of Complex Object Hypothesis rule for grass. (a) Histograms of color feature called "Excess Green" (defined as 2G-R-B), with a simple rule shown as a superimposed piecewise-linear function; (b) Response of simple rule coded as intensity (high response is bright); (c) Complex rule for grass.

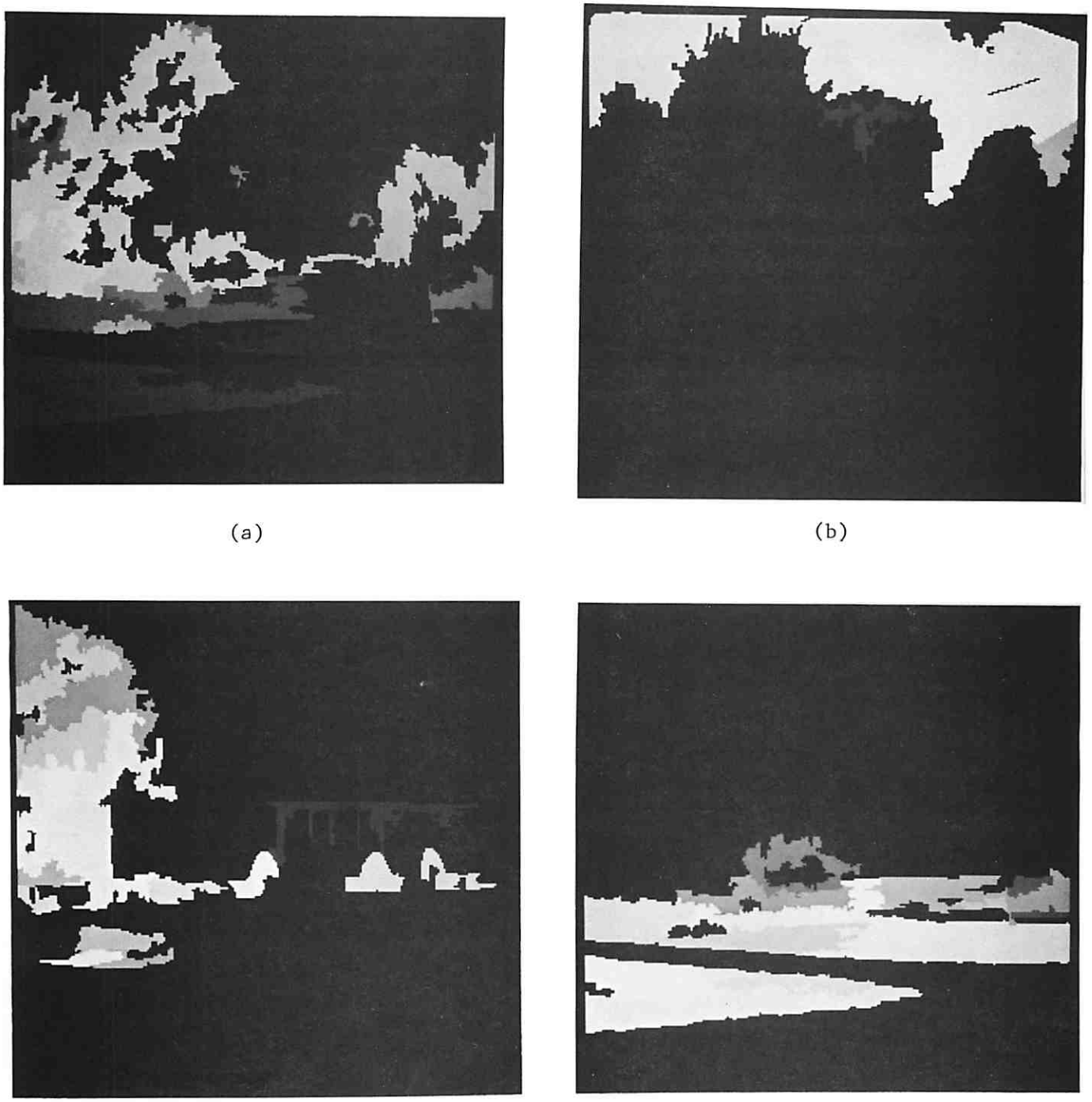
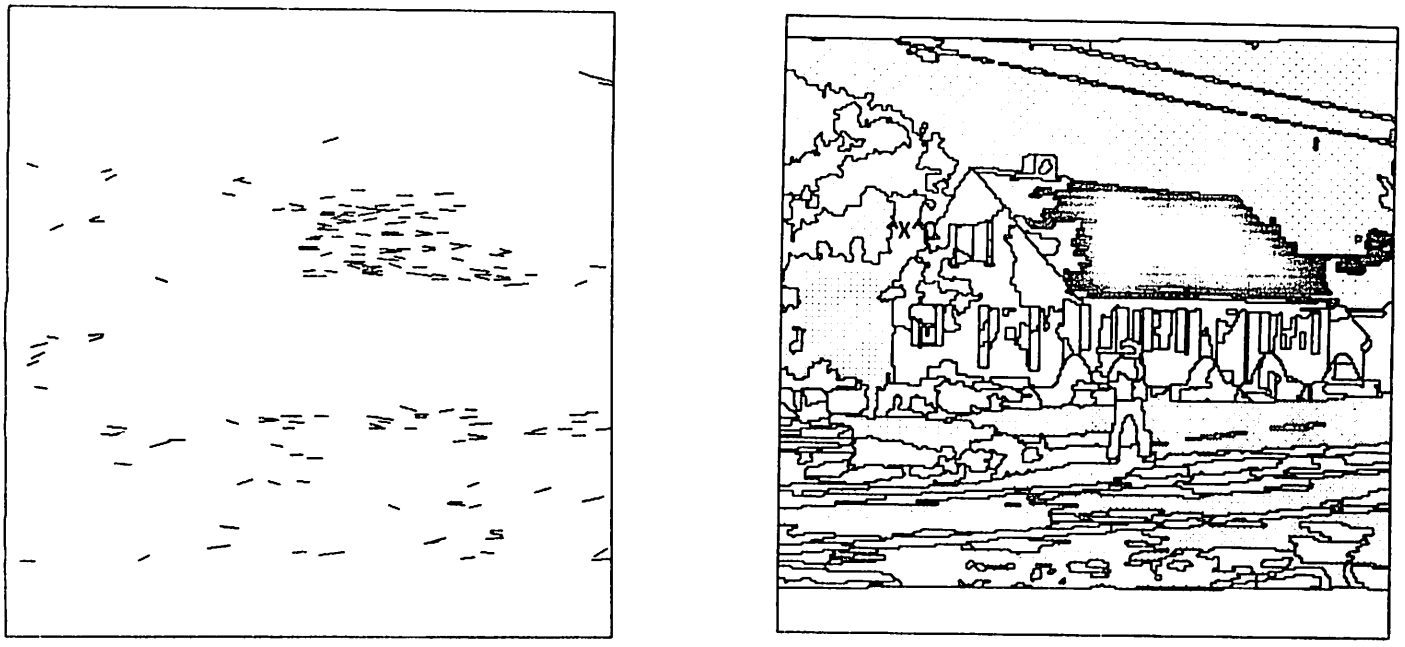
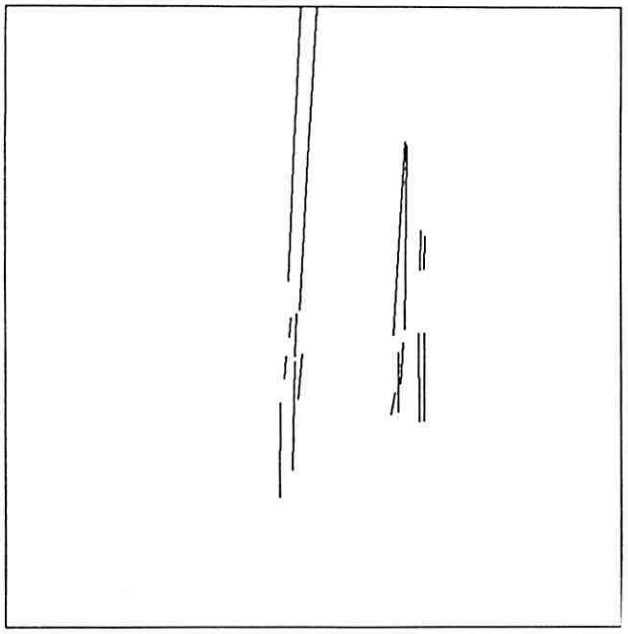


Figure 5.4 Examples of Applying Complex Rules to Generate Object Hypothesis rules. Each object has a complex rule which can be used to rank order a type of token in a focus-of-attention process. Each image shows the rule response for regions coded in intensity (high $\equiv$ bright).

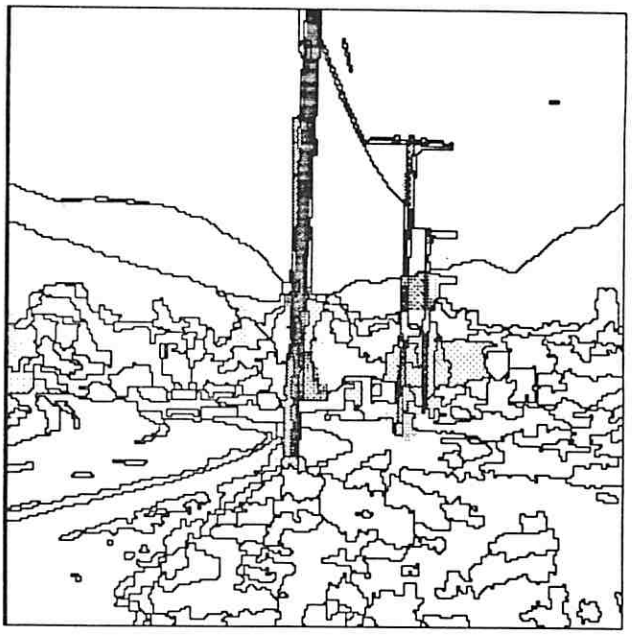


(a) (b)

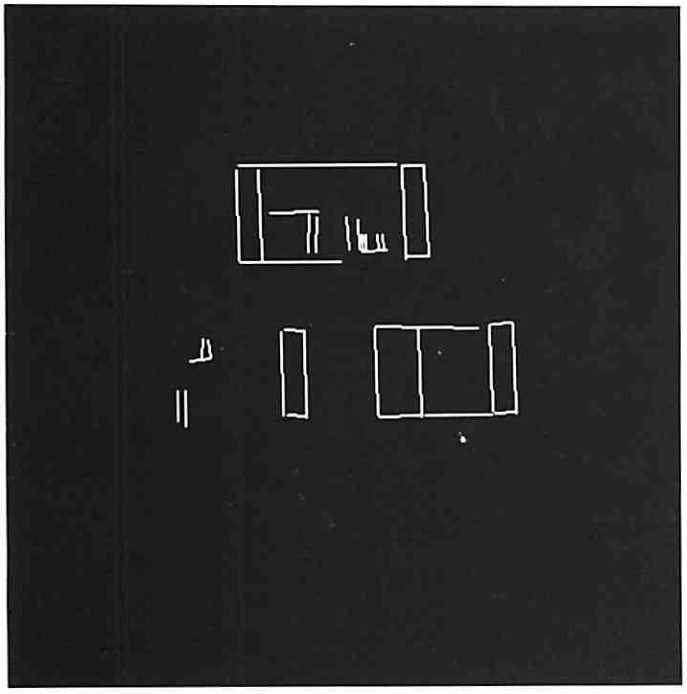
Figure 5.5. Examples of Relational Rules and Information Fusion. (a) A texture measure is formed by extracting short, horizontal lines which are INTERIOR to a region. The resulting set of lines are shown; (b) The density of extracted lines can be used to form a region attribute for rank ordering the regions; the density of shading depicts the region rating; (c) Telephone pole hypotheses can be generated by finding regions which are BOUNDED by pairs of parallel vertical lines; the line pairs formed by extracting vertical lines then selecting BOUNDING vertical lines, and finally PARALLEL pairs of BOUNDING vertical lines are shown; (d) The region scores are shown mapped back to the image (hatched regions have no long vertical lines and are vetoed). Note that further analysis of the regions with high scores will be necessary to fully and accurately extract the telephone poles; (e) Vertical and horizontal BOUNDING lines are extracted; (f) The region-line aggregations provide a focus-of-attention on rectangle hypotheses that form the geometric structure of the shutters/windows of the house.



(c)



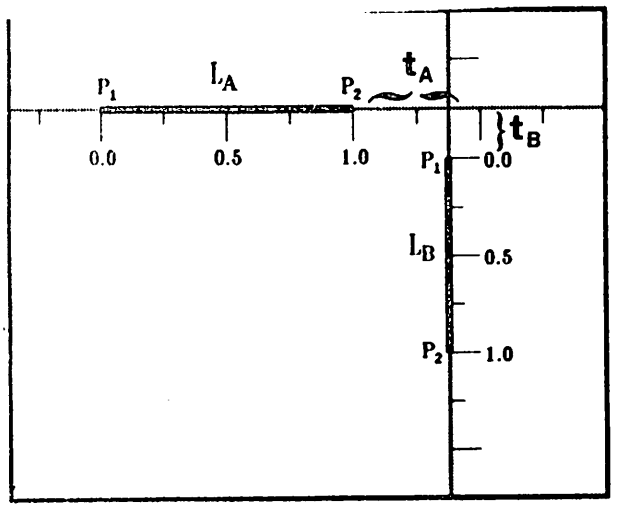
(d)



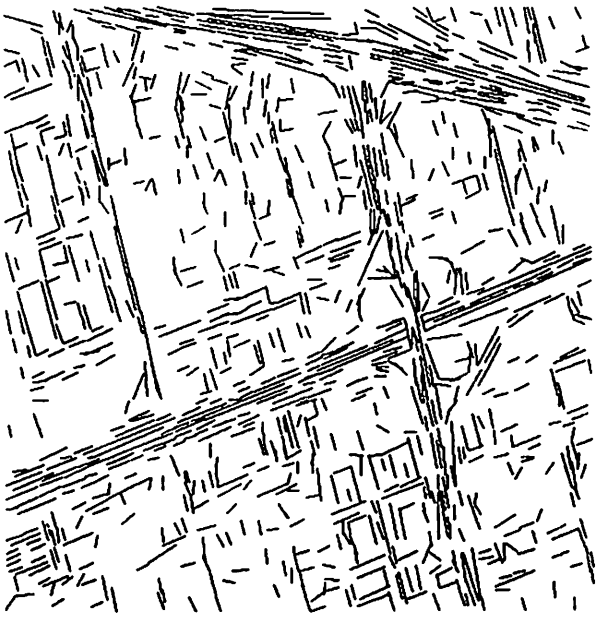
(e)

(f)

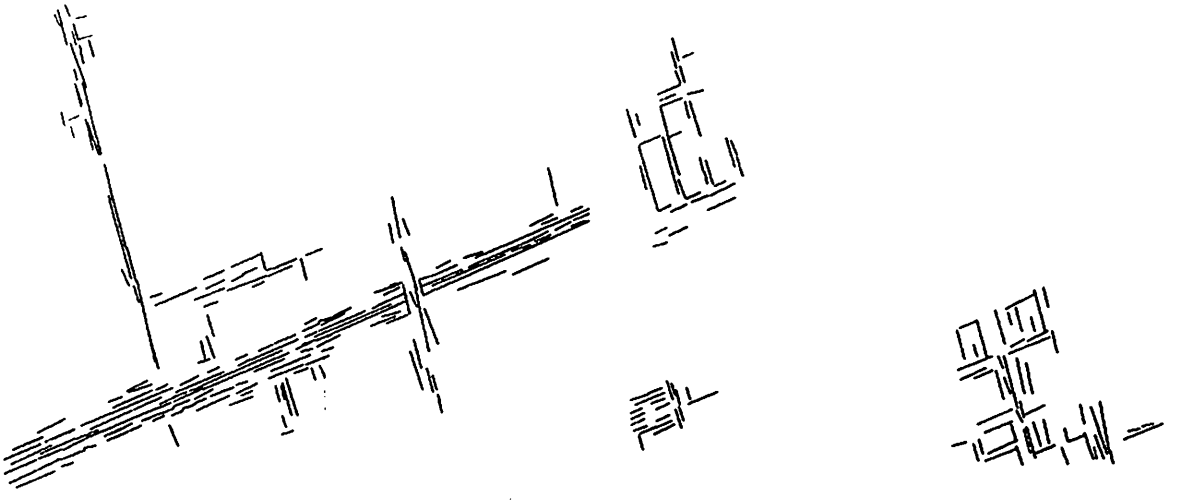
Figure 5.5, continued



(a)



(b)



(c)

Figure 5.6 Grouping via Geometric Relationships. (a) A graphical illustration of spatial proximity for orthogonal lines. Both the degree of proximity defined via parameterized distances t_A and t_B and the degree of deviation from orthogonality will contribute to the SPO measure for a pair of lines; (b) All lines of length $\geq$ 5 (pixels) extracted from Fig. 1.1(h); (c) Four example groupings obtained from a connected components algorithm for all lines which satisfy any of the SPO, SPP, and SPC relations.

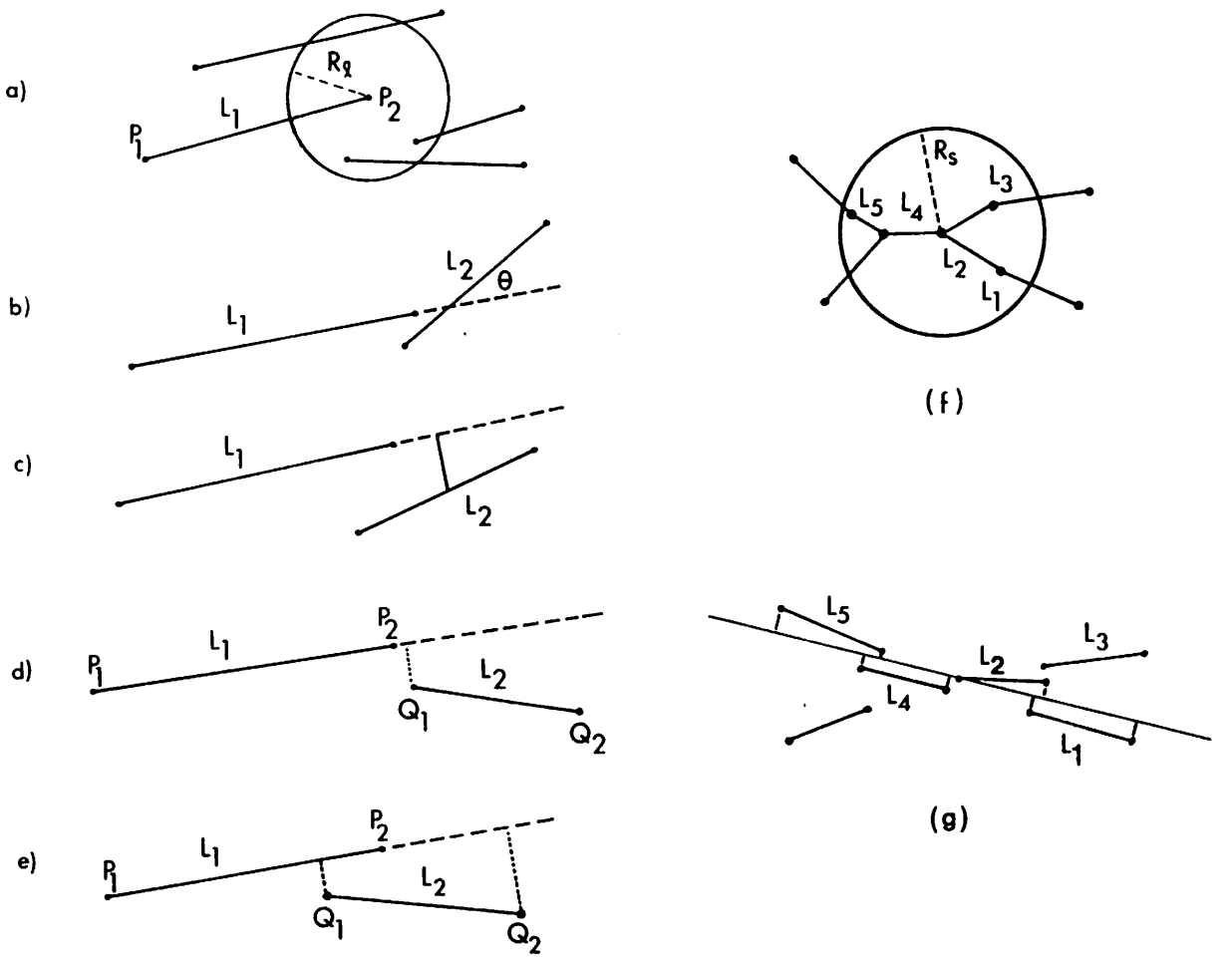


Figure 5.7 Straight Line Grouping (a)-(e) Grouping Relations; (a) Proximity - lines are candidates for grouping with L_1 if they are within the linking radius R_l from endpoint P_2, where R_l is a function of the length of L_1; (b) Orientation - lines must have similar orientation; (c) Lines must be close in the lateral direction; (d) Endpoints - the endpoint P_2 must be close to the projection of endpoint Q_1; (e) Overlap - the lines must not overlap too much; P_2 must be closer to the projection of Q_1 than to the projection of Q_2; (f) Link Graph - for each line there is a candidate group of lines which are possible extensions from each endpoint; (g) Straightness test - each sequence of lines is tested and the straightest is selected if it passes a threshold test.

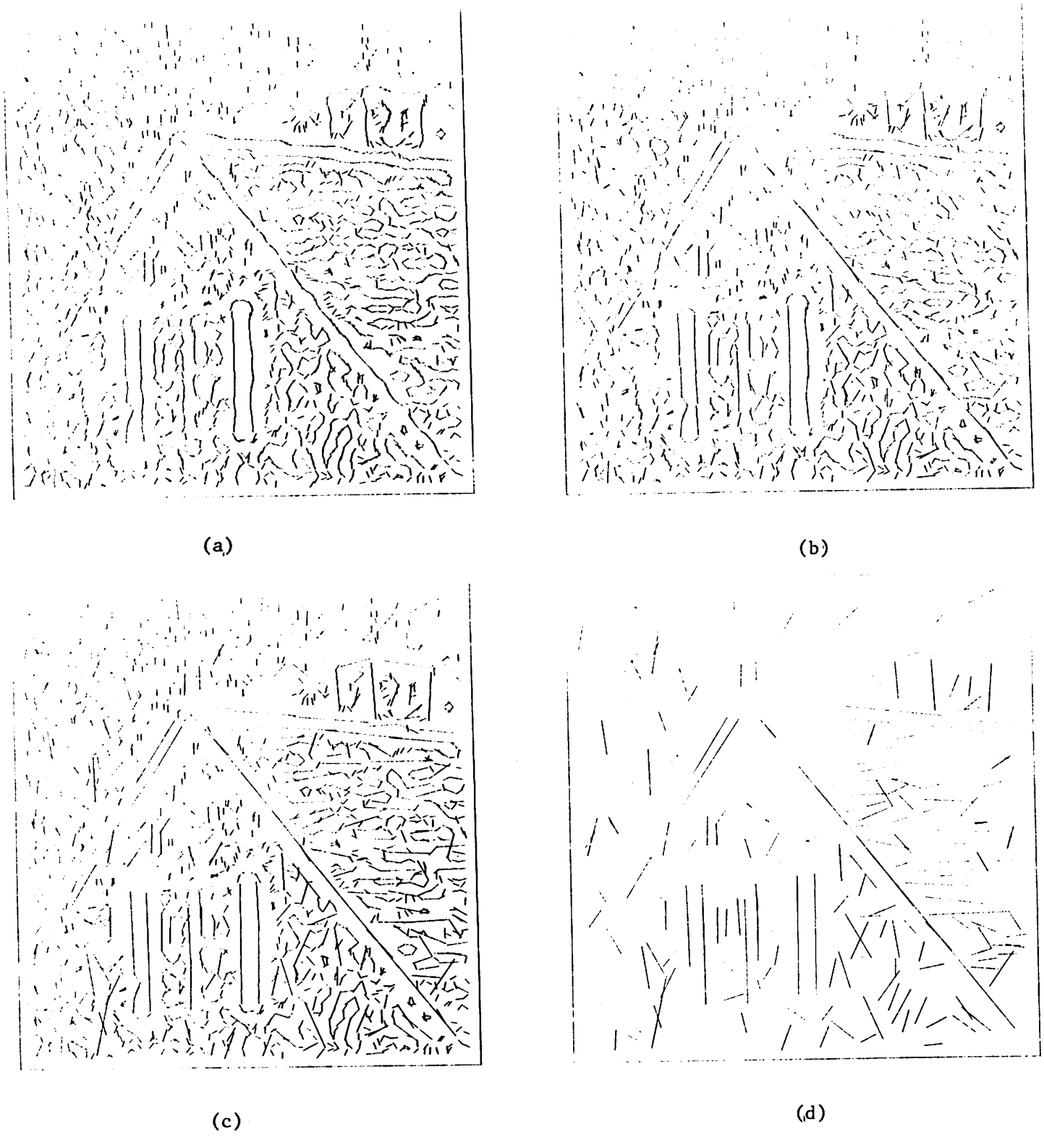
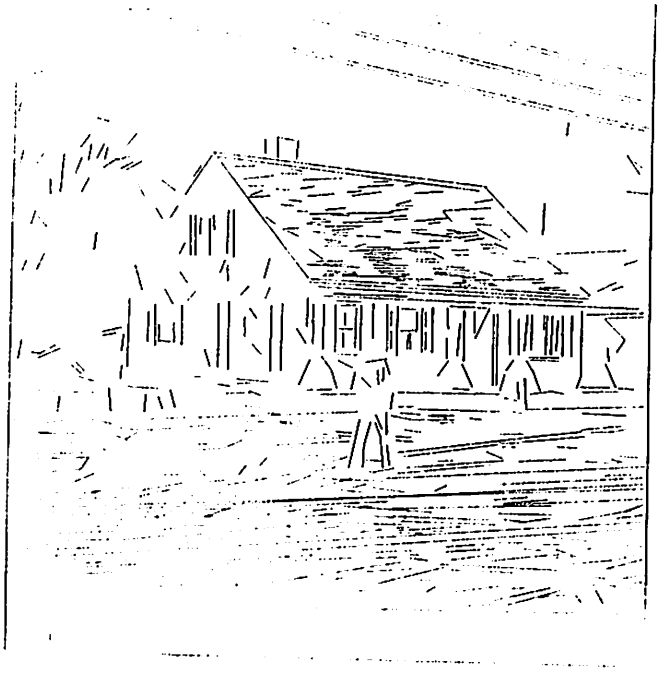


Figure 5.8 Hierarchical Grouping of Co-Linear Segments. (a) Initial line segments in subimage of chimney; (b) After two cycles of grouping; (c) After four cycles; (d) After six cycles; (e) Final results on whole image; lines of length 1 were filtered out; (f) After filtering on length and contrast.

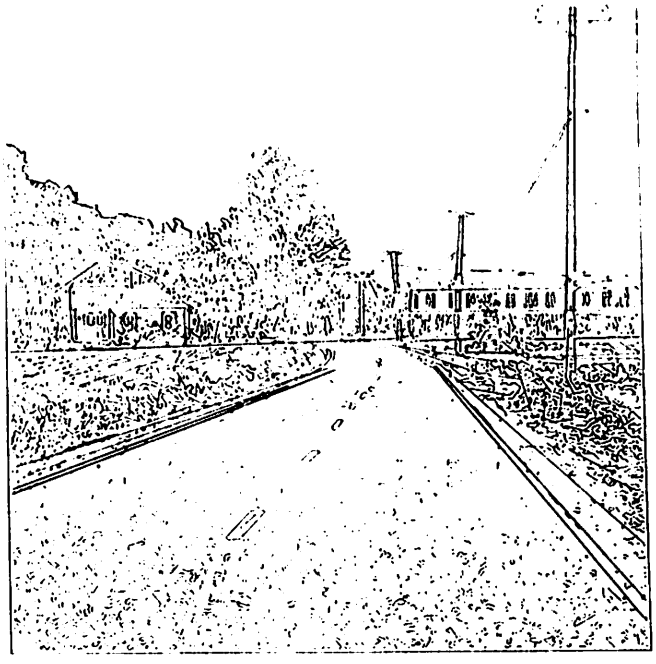
(e)



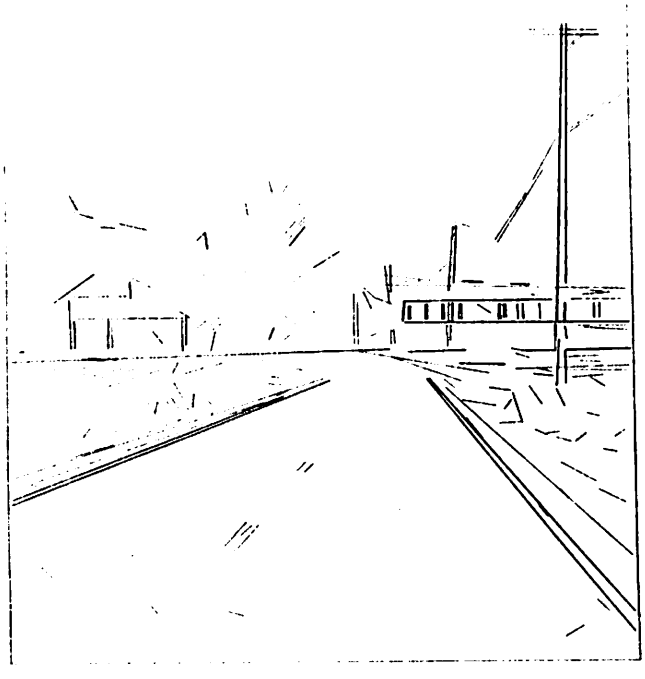
(f)

Figure 5.8 continued

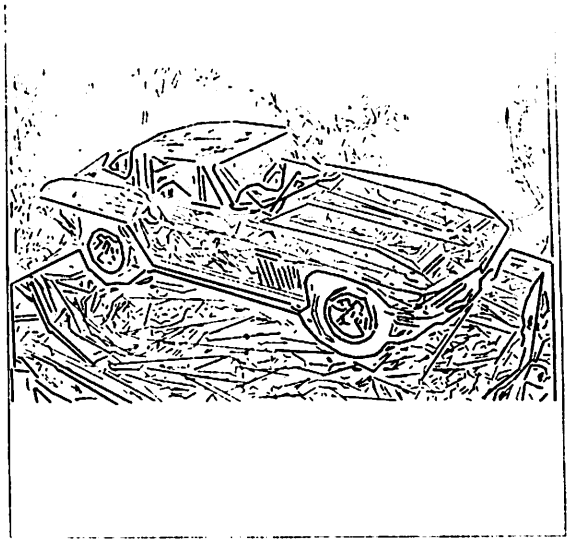
(a)



(b)



(c)



(d)

Figure 5.9 Additional Examples of Hierarchical Grouping of Co-linear Segments. The algorithm can be applied everywhere as a bottom-up line extraction process or applied in local areas under top-down control.

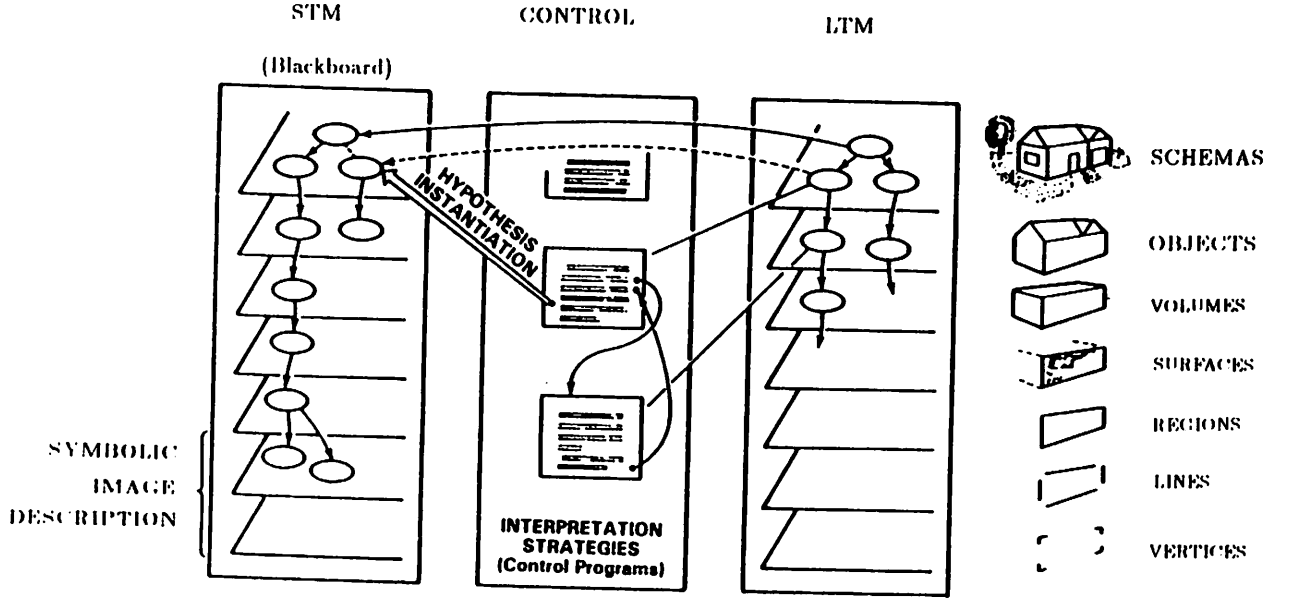


Figure 6.1 Short-Term Memory (STM) and Long-Term Memory (LTM) in the VISIONS System. STM consists of the data structures which form an interpretation of the scene. It is a hierarchical multi-level directed graph representation, where the lower levels are the symbolic tokens extracted from the sensory data and the higher levels are the inferred concepts of object and scene labels, as well as surface and volume hypotheses. LTM consists of stored knowledge about the world. LTM is organized hierarchically via the PART-OF and IS-A relations over the scene and object schemas. The schemas contain interpretation strategies that are responsible for matching the expected structures in the scene to the data in STM, and for creating the image-specific instance of the schema.

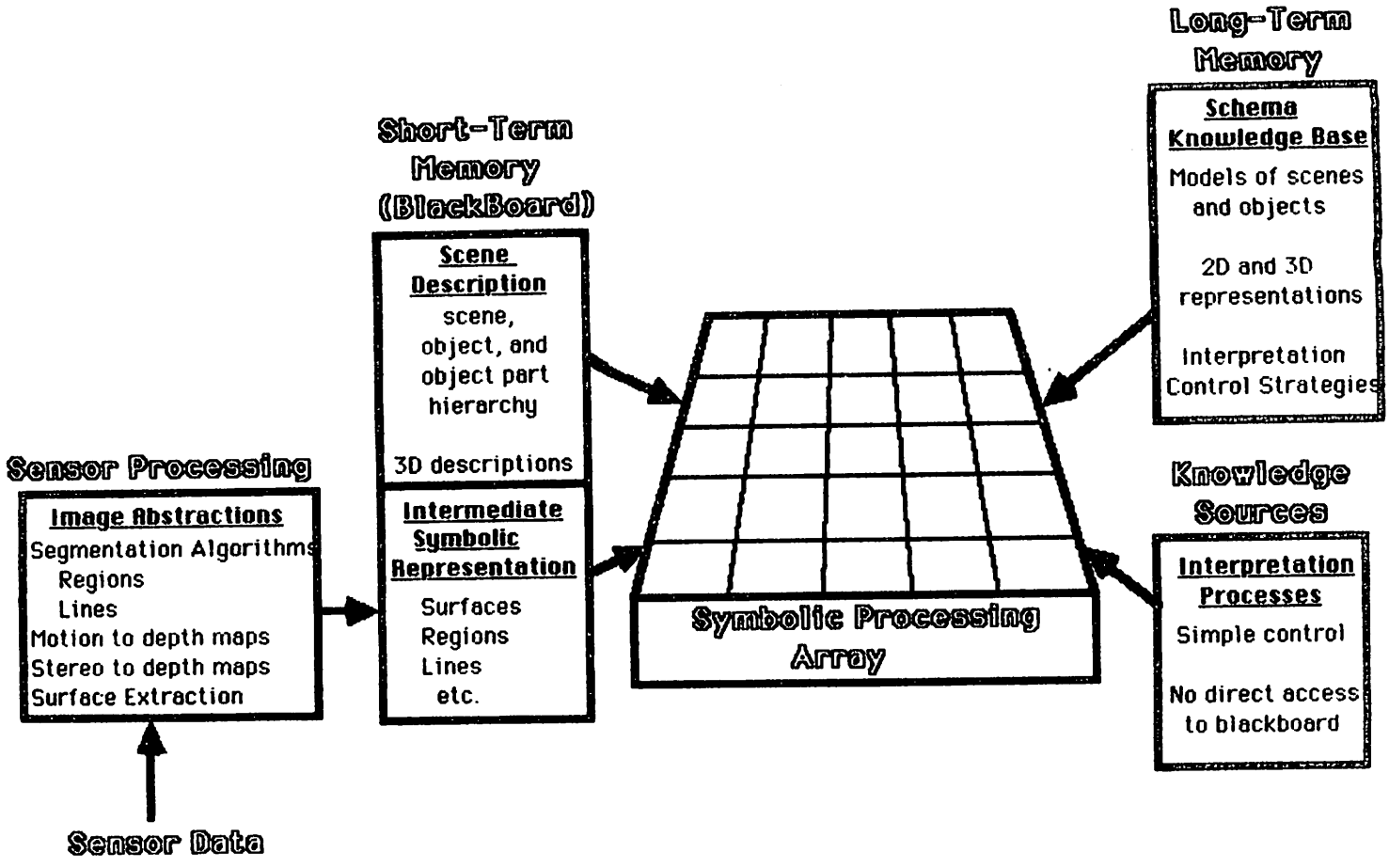
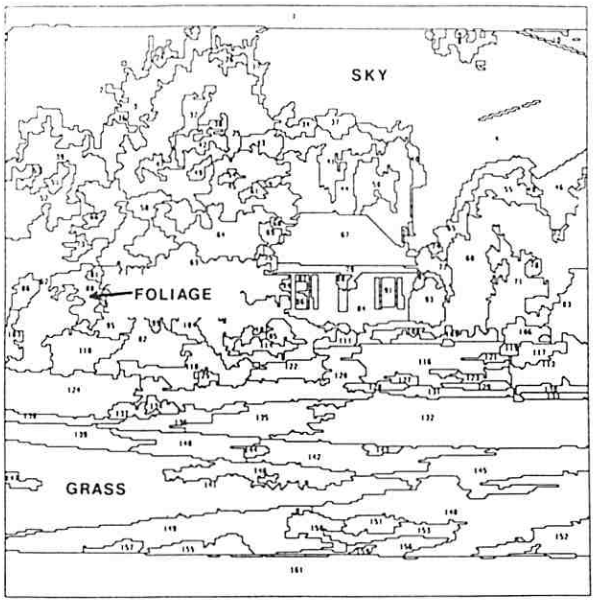


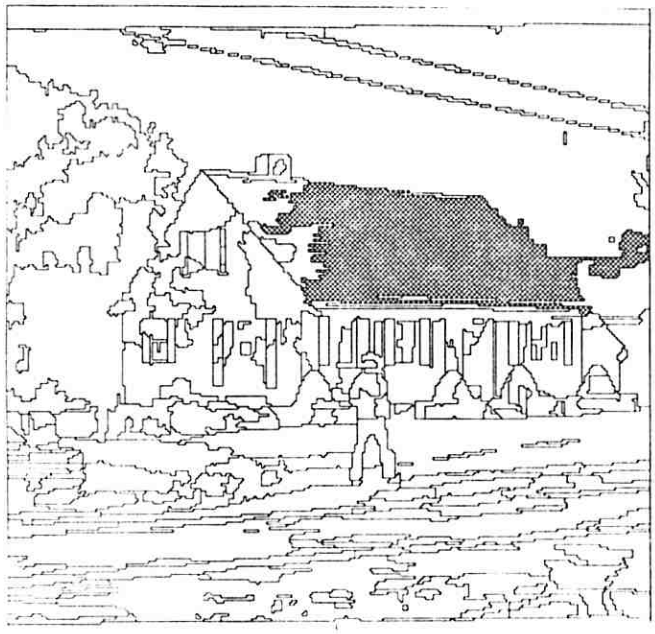
Figure 6.2 Overview of VISIONS System Components. Sensor Processes operate on sensor data producing a symbolic token in the Intermediate Symbolic Representation. Note that the short-term memory is a blackboard divided into the ISR with a large number of tokens and the inferred scene and object descriptions. Scene and object schemas form long-term memory; associated interpretation strategies can be invoked and executed in parallel on a multiprocessor (called the SPA) with each node accessing the blackboard independently. Knowledge sources are distinguished from schemas by their simplicity of control and that they do not directly access the blackboard.



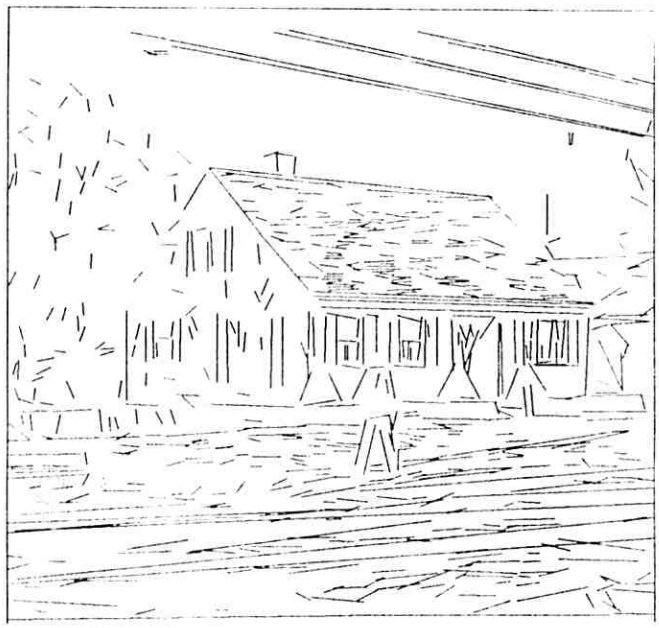
(a)

(b)

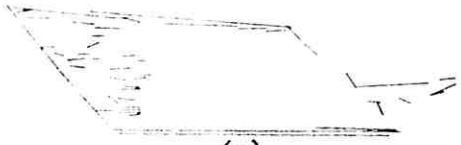
Figure 6.3 Exemplar Selection and Extension. One of the standard interpretation strategy involves the selection of exemplar tokens, which are then used as an "island of reliability" to extend the interpretation. (a) Exemplar regions for SKY, FOLIAGE, and GRASS, selected as the highest ranking regions candidates via their object hypothesis rules; (b) The similarity between the exemplar region for grass shown in (a) and all regions; brightness encodes similarity.



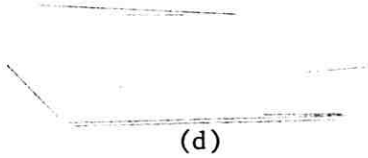
(a)



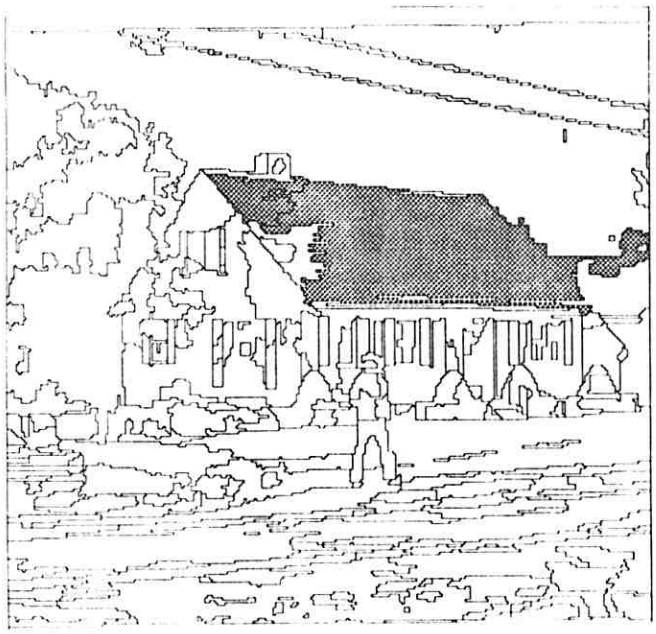
(b)



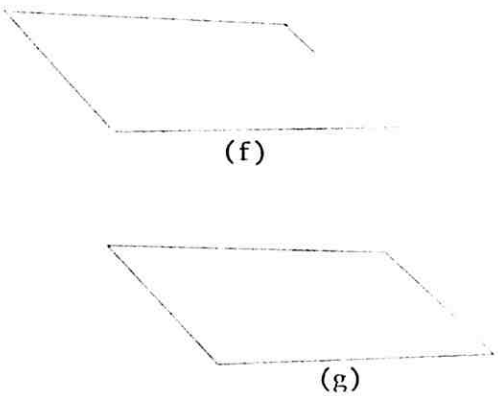
(c)



(d)



(e)



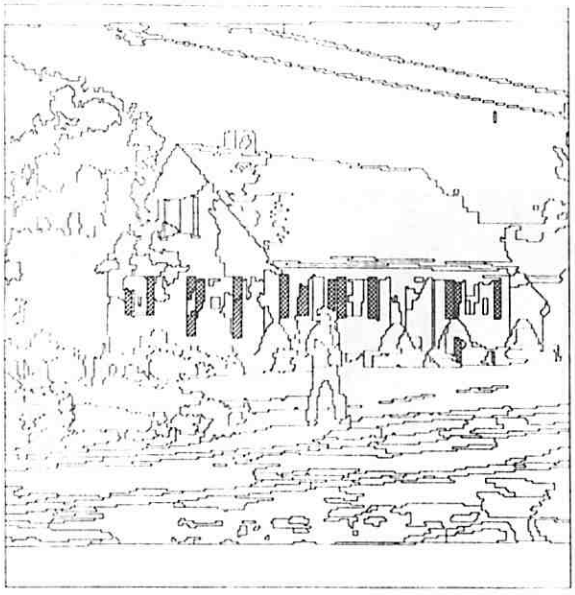
(f)

(g)

Figure 6.4. Steps in the schema directed interpretation of a roof. (a) region representation with the initial roof hypothesis region marked; note that it is fragmented because of the tree shadow; (b) the set of lines that were filtered on length and contrast to produce initial candidates for the interpretation system; (c) the set of lines intersecting the hypothesized roof region; (d) long lines bordering the boundary of the roof hypothesis region; (e) new roof hypothesis after merging regions which are partially bounded by the long lines in the previous image; (f) after joining colinear, nearby segments and removing close parallel lines; (g) the completed boundaries to forming the hypothesized roof trapezoid.

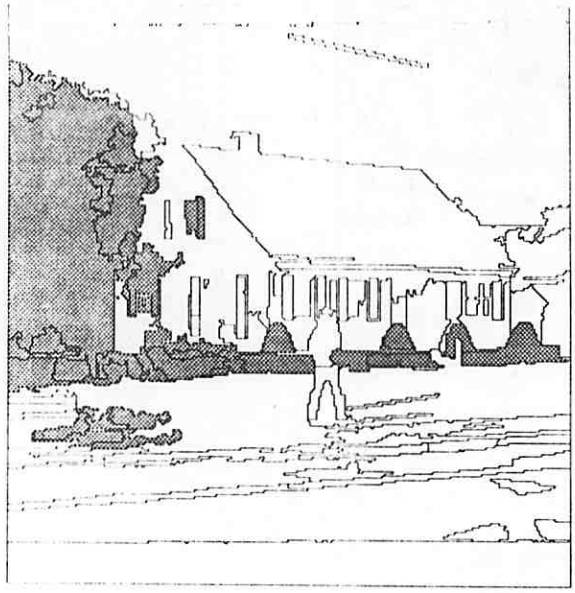
(a)

(b)

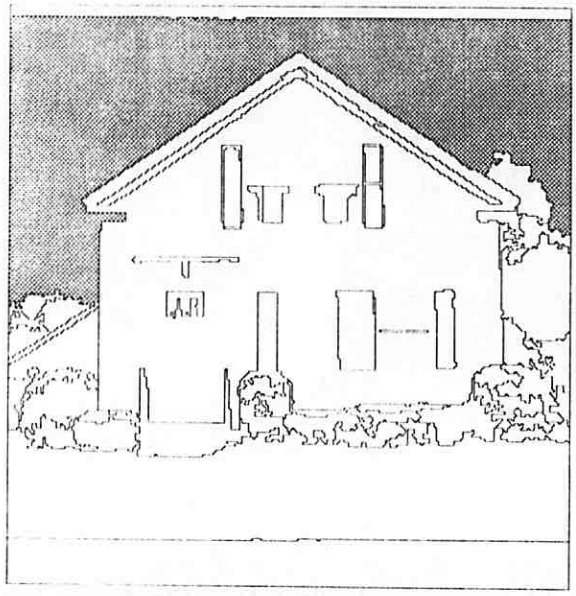


(c)

Figure 6.5. Result of Schema Interpretation Strategies for Generating Object Hypotheses. (a) roof; (b) grass; (c) shutters; (d) foliage; (e) sky. Each of the object hypotheses shown here can be applied independently and in parallel. Each object hypothesis generated must be verified and in many cases further refined. Note the shutters in (c) where some regions are incorrectly labelled and others are missing. Nevertheless, these object hypotheses forms a partial interpretation that can be further developed incrementally.

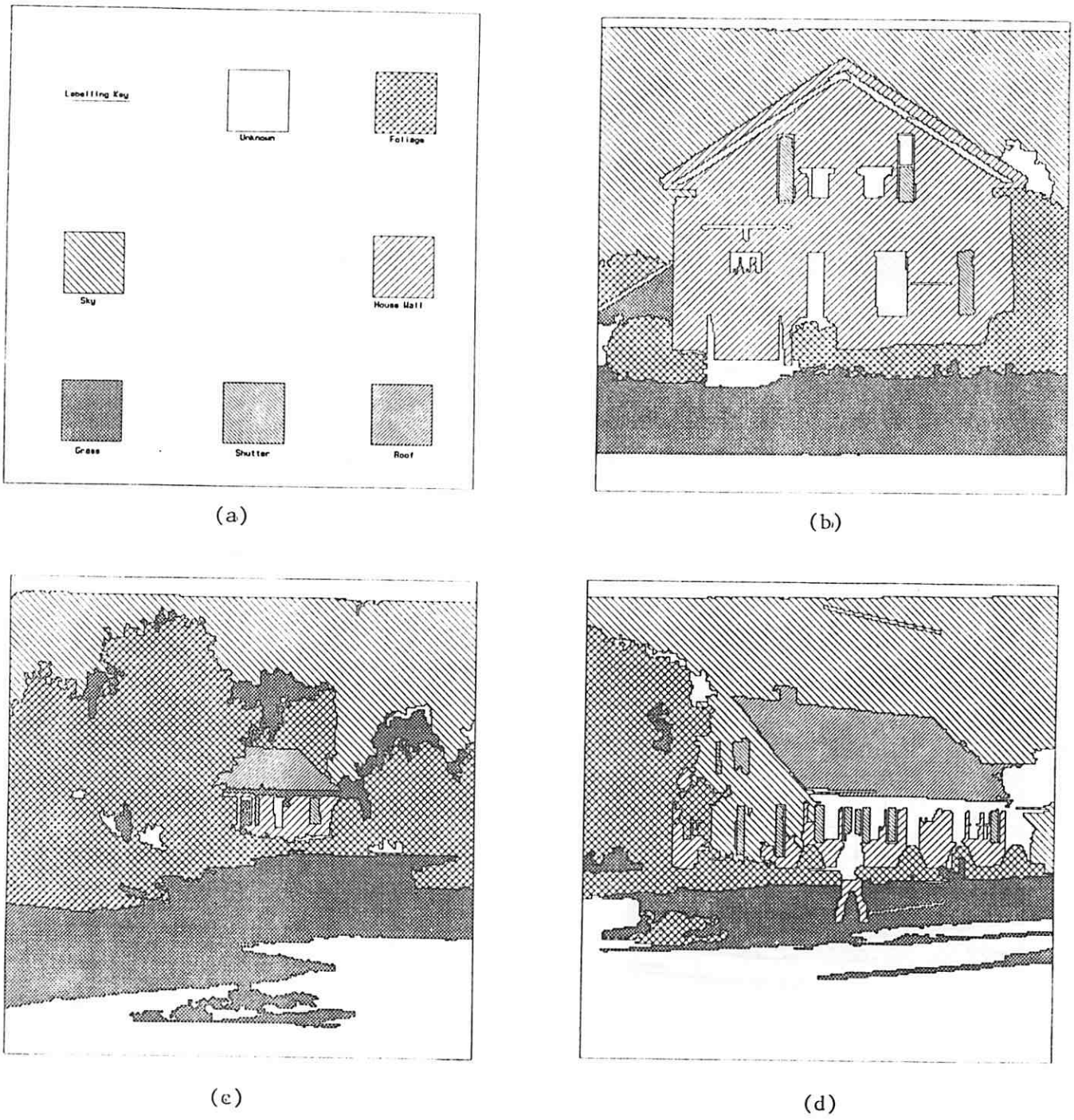


(d)



(e)

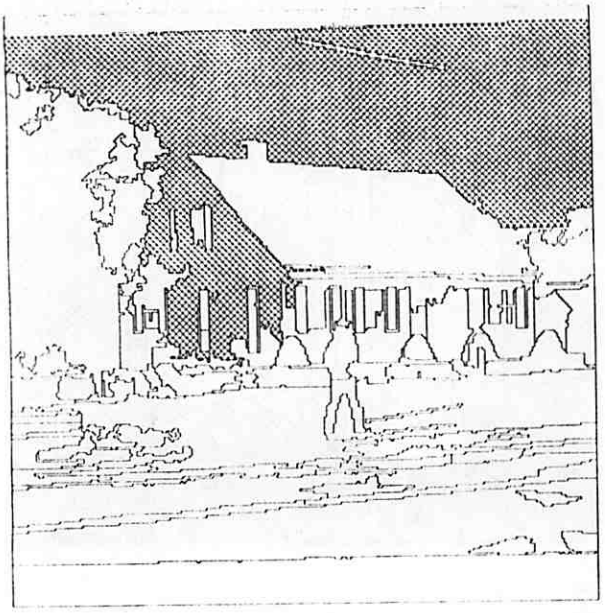
Figure 6.5, continued



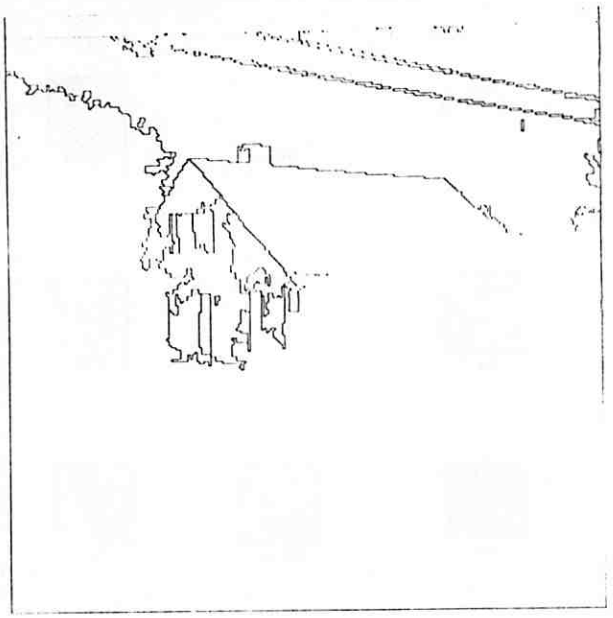
(a) (b)

(c) (d)

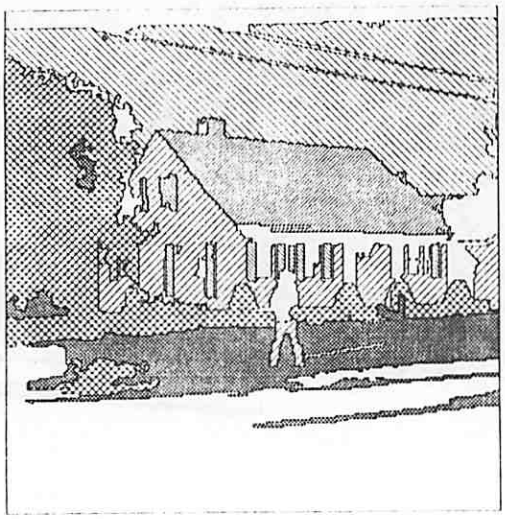
Figure 6.6. Example Final interpretations. These images show the final results obtained by combining the results of the interpretation strategies under the constraints generated from the knowledge base. (a) interpretation key; (b-d) interpretation results. In (d), the missing boundary between sky and wall results in a labelling conflict for the side wall (which is shown labelled as sky here). Note: These experiments did not significantly utilize the line information in the interpretation strategies for grouping and information fusion as discussed in Section 5; those capabilities have been more recently developed and are being integrated at a general level.



(a)



(b)



(c)

Figure 6.7 Feedback-directed Resegmentation and Interpretation. (a) There are multiple conflicting hypotheses for the shaded region; (b) the result of a resegmentation process with more sensitive setting; (c) the resulting interpretation when the resegmented regions are added to the initial segmentation and reinterpreted.

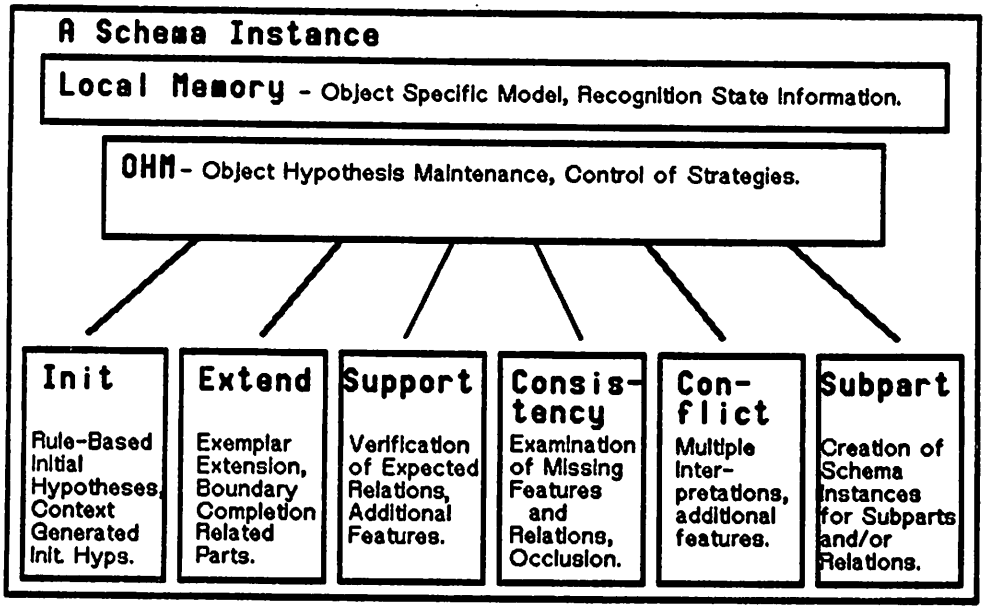


Figure 6.8 The Schema Template. The template provides a skeletal structure to guide the construction of schemas; it incorporates seven concurrent interpretation strategies which support various stages in the life of a hypothesis: creation, extension, integrating support, checking consistency, resolving conflict, and creating other related hypotheses.

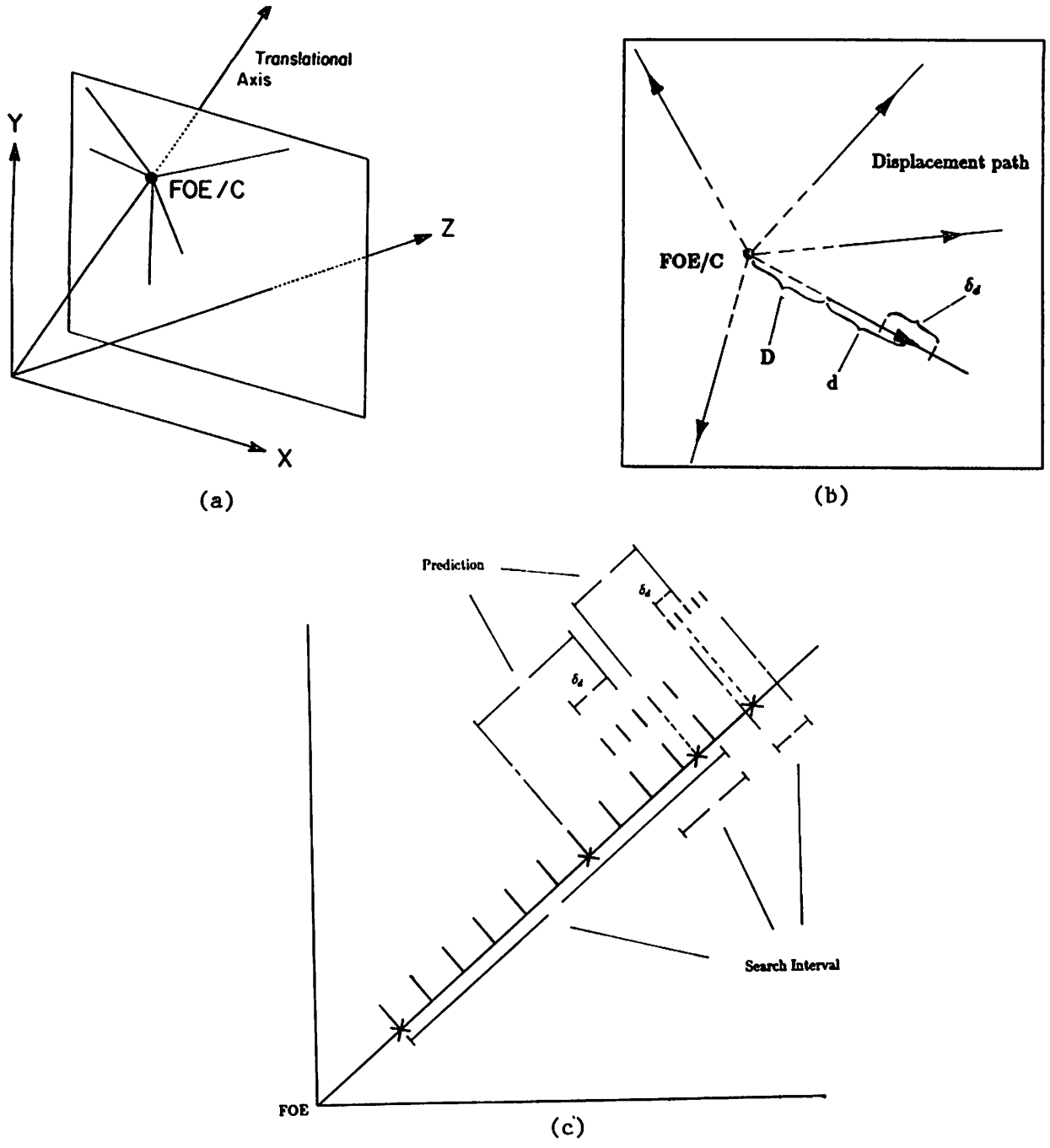
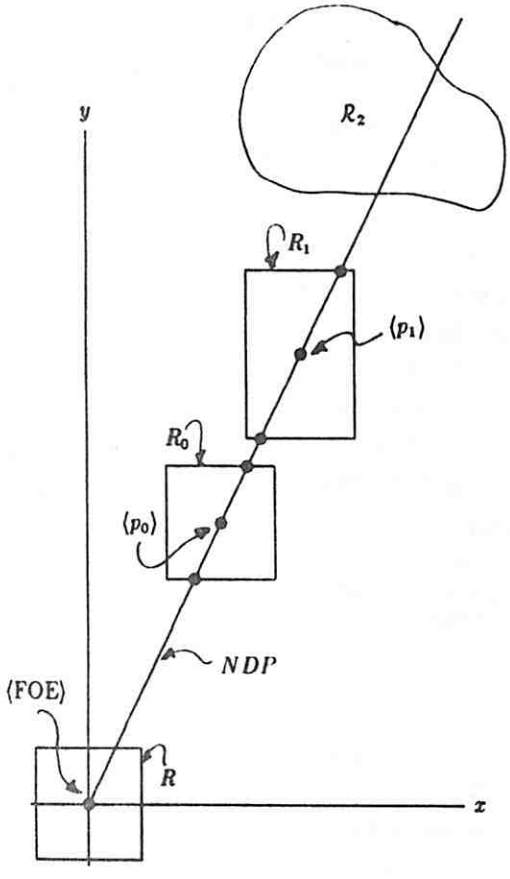
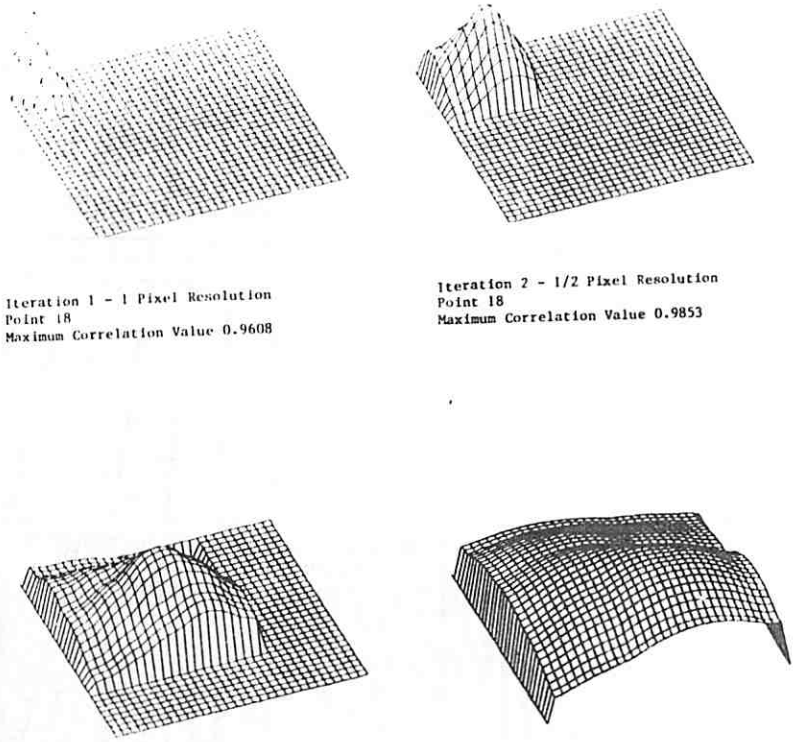


Figure 7.1 Multiframe Recovery of Depth from Translational Motion. (a) For pure translational motion, displacements of image features are constrained to move along linear paths radiating from the focus of expansion (FOE), which is the point of intersection between the infinite image plane and the axis of translation; (b) An environmental point of approximately known depth D will be displaced a distance d with an uncertainty interval of δ_d; (c) As depth is refined over multiple frames, future search windows δ_d can be smaller and matched at higher resolution; (d) The search window R_2 is actually two-dimensional because there is actually two-dimensional uncertainty R in the position of the FOE and two-dimensional uncertainty R_O and R_1 in the position of the feature points $< po >$ and $< p1 >$ respectively in two previous frames; (e) The 2D correlation surface at successive temporal matches at higher resolution.



(d)



(e)

Figure 7.1 continued

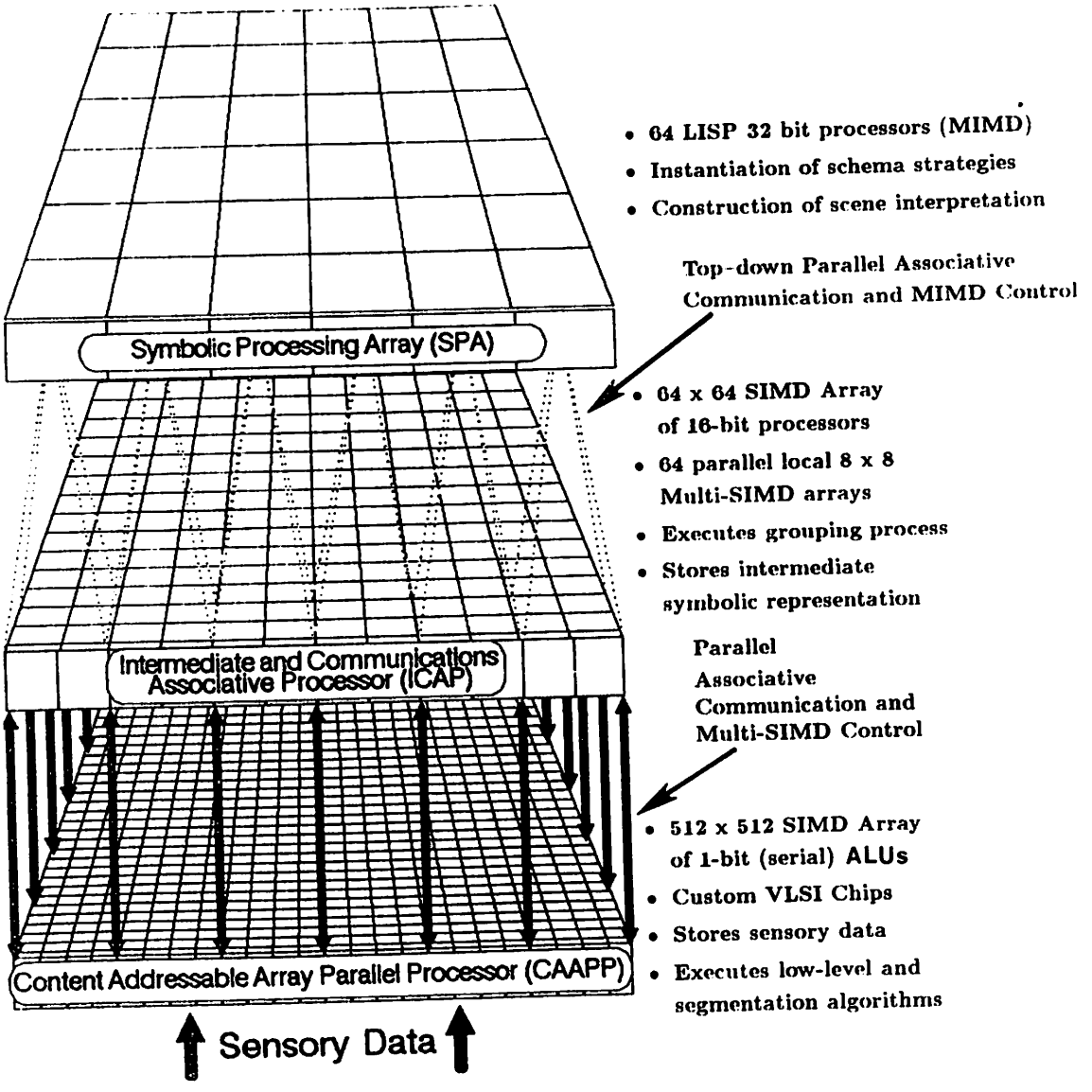


Figure 8.1 Associative Image Understanding Architecture (IUA). There are three levels of tightly coupled parallel computing elements with associative communication and control between levels.

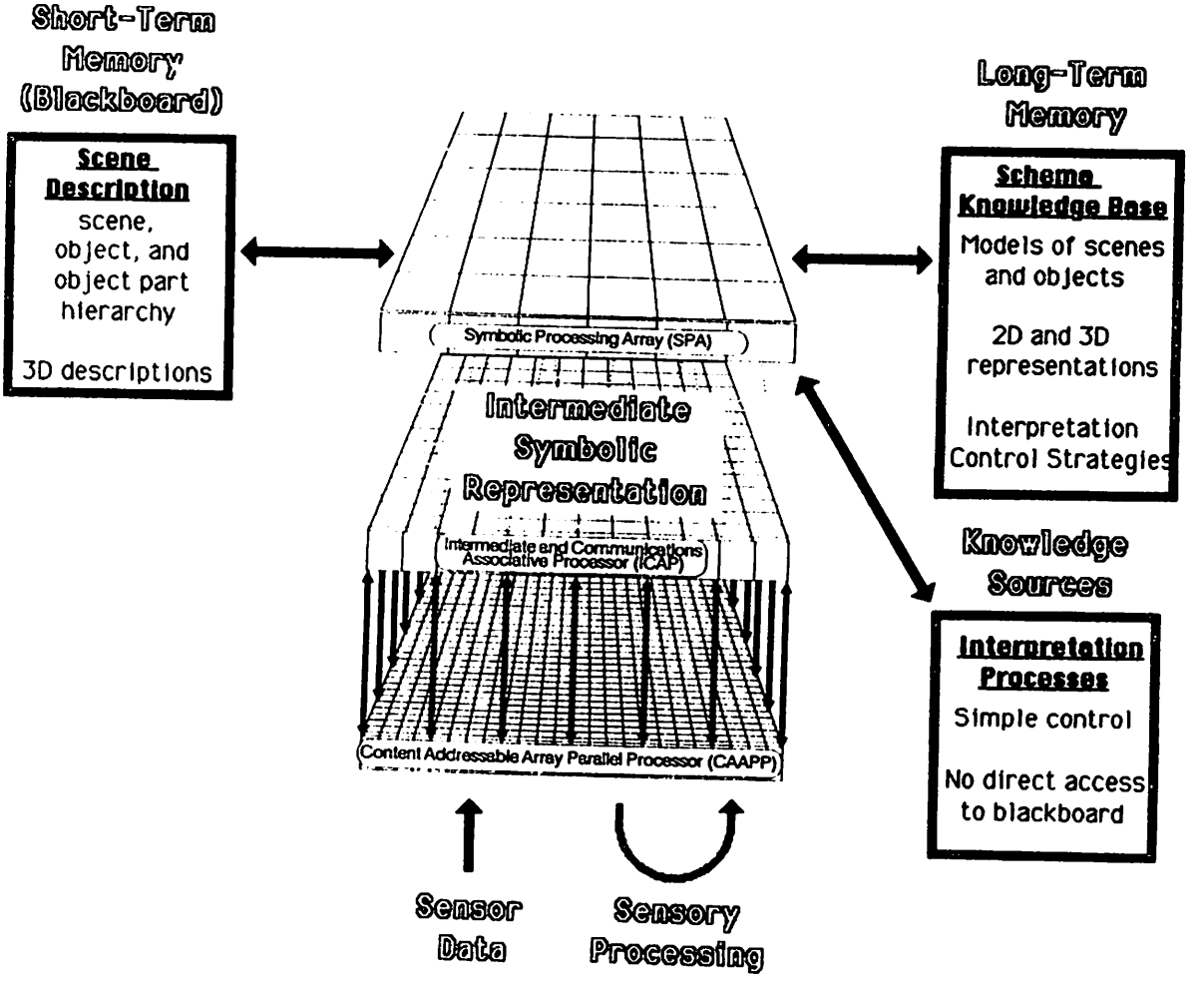


Figure 8.2 The VISIONS system mapped onto the Associative IUA. Sensory processing takes place in SIMD mode on the CAAPP, while the Intermediate Symbolic Representation is stored at both the CAAPP and ICAP level. Schemas and knowledge sources are invoked to run in parallel on the SPA, building the interpretation on the distributed or shared memory multiprocessor.