A REGION-ORIENTED IMAGE-ANALYSIS SYSTEM BY COMPUTER

Yu-ichi Ohta

Department of Information Science

Kyoto University

March 1980

A REGION-ORIENTED IMAGE-ANALYSIS SYSTEM BY COMPUTER

Yu-ichi Ohta

March 1980

Department of Information Science
Kyoto University
Kyoto, 606, JAPAN

Submitted in partial fulfillment
of the requirements for the
degree of
Doctor of Engineering
at
Kyoto University

A REGION-ORIENTED IMAGE-ANALYSIS SYSTEM BY COMPUTER

Yu-ichi Ohta

ABSTRACT

In this thesis, we study a region analyzer for color scenes. The major issues addressed and described here are the following: (1) the role of color information in region segmentation; (2) the technique of partitioning an image into a set of regions; (3) the technique of managing the regions in a symbolic data structure; and (4) the modeling and control scheme for obtaining the "best" match between the model of a task world and the set of regions obtained from an input image.

Systematic experiments have been performed to examine the role of color information in region segmentation. A segmentation scheme, called "dynamic K. L. transformation", was developed for this purpose. We have found a new set of color features effective for region segmentation.

A powerful segmentation program was developed for preliminarily partitioning an image data into a set of regions. The result of segmentation is organized into a well-structured symbolic data network, named "Patchery Data Structure", with retrieving facilities.

The knowledge of the task world is represented as a set of rules. Bottom-up control and top-down control are combined in the rule-based region analyzer. A plan is generated by the bottom-up control as a representation of the rough structures in an input scene. A symbolic description of the scene is made in the top-down analysis. The top-down process is constructed by using a production system architecture.

Outdoor scenes including sky, trees, buildings, and roads have been successfully analyzed by the system.

ACKNOWLEDGMENT

I would like to express my sincere appreciation to Professor Toshiyuki Sakai for supervising this research with adequate guidance.

I am also grateful to Associate Professor Takeo Kanade for his continuing advice and for having had enlightening discussions with me.

I am indebted to my colleagues in Prof. Sakai's Research Group for helpful discussion and co-operation. Mr. Lee Soo Youn, Mr. Ryoya Mori, and Mr. Toyoaki Nishida provided me with the conveniences for text editing and pretty printing.

Last but not least, I wish to thank my wife Kiyomi for her constant encouragement.

TABLE OF CONTENTS

Chapter I

INTRODUCTION

An image is a two-dimensional (2-D) projection of a three-dimensional (3-D) scene. The task of an image analysis system is to make a description of the scene from its image. A complete 3-D description of the scene, of course, can not be reproduced from the image data alone. The information contained in an image is not sufficient to determine all the "parameters" necessary for the reproduction.

In order to make the reproduction actually solvable, we must reduce the number of the parameters. One way is to restrict the world of the scene in which the image analysis system works. The restricted world is called a task. Within the task world, we can find a lot of "rewriting rules" from the image features to the scene description. These rewriting rules are called the "knowledge" of the task world and they are represented in the model.

This thesis is devoted to develop a scheme for model representation and control structure for image analysis. As well as these "higher level" problems dealing with representation and use of knowledge, we have paid much attention to the "lower level" ones in the image analysis; such as signal-based region segmentation or representation of segmented regions.

We employed the region analysis method to organize our image analysis system. The region analysis technique and the edge analysis technique are two of the major methods employed in analyzing pictures; regions and edges are complementary to each other. Yet, region analysis has not been studied as well as edge analysis. Recently, the region analysis technique are capturing attentions for several reasons. One of the most significant reasons is that the regions can solve some problems which are difficult for the edges.

In this thesis, we study a region analyzer for color scenes. Outdoor scenes are mainly used as a task of the analyzer. This is because the outdoor scenes include such objects as trees, sky, etc., which are naturally defined by the properties of regions rather than of edges. The placement relations among the objects can be dealt with easily by using the regions. The major issues addressed and described in this thesis are the following: (1) the role of color information in region segmentation; (2) the technique of partitioning an image into a set of regions; (3) the technique of managing the regions in a symbolic data structure; and (4) the modeling and control scheme for obtaining the "best" match between the model of a task world and the set of regions obtained from an input image.

I-1. Aspects of Region Analysis

A region analysis system covers a wide range of subjects varying from the digitization of pictures to their semantic interpretation. In this thesis, we assume that an input color image is given as a set of digitized intensity arrays corresponding to the red, green, and blue components of color.

(1) Color Information for Region Segmentation

The subject of color image segmentation might be studied as a simple extension of the segmentation for black and white images. Many researchers have recognized the importance of color information in image segmentation. However, how effectively we can use chromatic information in the segmentation process and what color coordinate systems are most appropriate has not been much studied. We have tried to obtain a solution for these problems in the case of region segmentation. A set of color features is derived through an experiment of computing effective color features by means of the Karhunen-Loeve transformation at every step of segmenting a region. Comparisons are made among the segmentation results obtained by using the various sets of color features which are usually used in image analysis. Probably, this is one of the first systematic experiments actually performed to search the effective color information in segmentation. Chapter II describes this issue in detail.

(2) Region Segmentation Technique

The region segmentation is a process which partitions an image into a set of regions. Each region is characterized by some consistent features such as color or texture, and such regions are often called "coherent" or "homogeneous" regions. Regions in an image correspond to surfaces of objects in a real world. The region segmentation is thus based on the assumption that characteristics of a surface are usually consistent. The "coherent" regions are not always equal to "meaningful" regions in many cases, but they are the atomic elements for constructing a description of the scene.

Region segmentation techniques can be divided into three classes: (1) region splitting, (2) region merging, and (3) combined use of splitting and merging.

The region splitting technique partitions an image into regions in a top-to-bottom manner. It starts with the entire image and works toward the set of "coherent" regions. The distribution of image features in the spectral domain is often used to obtain criteria for the splitting operation [Tomita et al., 1973; Ohlander, 1975]. The method is suited to extract global structures in the image, but is usually weak in detecting detailed ones.

The region segmentation based on region merging operations is alternatively called region growing. It starts with atomic regions, e.g. pixels or tiny square regions, and works toward the set of "coherent" regions in a bottom-to-top manner. Local spatial features, such as the contrast at boundaries between regions, are used as criteria for the merging operation [Brice et al., 1970]. The method is good in extracting detailed structures in the image, but is rather sensitive to noise.

The spatial resolution of segmented regions can be defined as the ratio of the size of the atomic region to that of the whole image. The region growing with high spatial-resolution tends to require a far more computational cost than the region splitting which achieves the same resolution.

The split-and-merge method [Horowitz and Pavlidis, 1974] aims to gain computational efficiency preserving the merits of both the splitting and merging methods. A pyramidal data structure [Tanimoto and Pavlidis, 1975] provides a working environment for this split-and-merge method [Pavlidis, 1979].

If the regions are used as atomic elements for image analysis, they must have sufficient spatial resolution to evaluate shape parameters of objects. There are a lot of algorithms to obtain a set of regions, but actually only a few algorithms can produce the segmentation of an image with satisfactory resolution. The segmentation algorithm developed in our system is of the splitting type. It can extract the detailed structures in the image as well as the global ones with a sufficient spatial resolution. Chapter III

describes this issue.

(3) Symbolic Representation of Regions

A region segmentation process produces as a result a two-dimensional array which indicates the region numbers: the points in the same region retain the same number. Any data about the segmented image can be derived from this array and the original image. The computation needed for the derivation, however, is time consuming, because it must deal with the image arrays directly.

When regions, not pixels, are used as atomic elements for analysis, we can perform the analysis without dealing with the two-dimensional image arrays at all. The data needed for the analysis can be described by using the regions as descriptive elements. In order to support the high-speed retrieval of any kind of data about the segmented image, the description should be a well-organized data structure. Various database schema, such as the network model or the relational model, can provide a model of the data structure. Data retrieving facilities for the data structure are also essential to enable the flexible retrieval of pictorial data.

We have defined a structured data network, named "Patchery Data Structure", together with a set of retrieving functions. Regions, boundary segments, vertices, etc. are used as the descriptive elements. This subject is also described in chapter III.

(4) Modeling and Control for Region Analysis

The model in an image analysis system represents the knowledge of the world in which the system works. Objects and various concepts are defined in the model by using the similar terms in the real world. But it must be noted that the knowledge in the model is valid

only in the world given as a task. That is, the objects and the various concepts in the model are defined within the restricted task world, and they are not always usable in the unrestricted real world.

It is well known that there are two complementary methods for model representation: procedural and declarative.

In the procedural method, the knowledge is embedded in the program which performs the image analysis. The control structure of the method is defined implicitly in the control of the program. Given a task, the procedural method provides a flexible scheme to construct an efficient image analysis system. But the structure of the analysis mechanism is rather unclear, and a slight change of the task often demands complete changes of the system. A successful example of the procedural method can be found in the face-analysis program developed by Kanade [Sakai et al., 1972; Kanade, 1977].

In the declarative method, the model is represented as a collection of the descriptions of properties of or relations between the objects. The model has a modular structure and the control structure is clear. However, it is very difficult to develop a modeling and control scheme in a declarative fashion when the task world is rather complex.

Recently, rule-based architectures, such as production systems [Davis and King, 1975], are often employed to construct expert systems in Artificial Intelligence. In these architectures, the model is represented as a collection of simple modules, and the control structure is also simple. They aim to combine the merits of both the procedural and declarative methods. We have employed a rule-based architecture in region analysis.

The control structure in an image analysis system defines the way to search for the "best" match between a model and an input image. In region analysis, the input image is preliminarily partitioned into a set of "coherent" regions based on intensity information. If the coherent regions have one-to-one correspondence with the objects defined in the model, i.e., the coherent regions are

the meaningful ones, it is an easy problem to search for the "best" match between the regions and the objects. It is almost impossible, however, to obtain a set of meaningful regions by using only the "low-level" image information (i.e. intensity or color), and the control structure must search for many-to-one correspondence between the regions and the objects.

The depth of the search tree in the image interpretation is determined by the number of regions. The branching factor is determined by the number of objects. The search space is prohibitedly large, but the situation is a little better than the ordinary tree search in game problems. Scenes have a favorable property which we call "locality". By applying this property to the search scheme, it is possible to reduce the search space drastically.

The details of our modeling and control scheme are described in chapter IV.

I-2. Overview of the Region Analyzer

This section provides the readers with an outline of the region analyer which we have developed. The main issues and the detailed descriptions are included in the following chapters. Figure 1-1 shows two major steps of the analysis mechanism: the preliminary segmentation and the rule-based analysis. The system receives red-green-blue intensity arrays of a digitized image and constructs a semantic description of the scene.

Preliminary segmentation --- This step is basically "nonpurposive", and it can be applied to a wide range of tasks. The primary objective of the preliminary segmentation is not

the reduction but the structuring of raw image data into usable information. It segments the input color image into a set of coherent regions based on the color information. An Ohlander-type segmentation algorithm [Ohlander, 1975] is employed with several improvements to extract detailed structures from the image data. The regions obtained by the preliminary segmentation are used as the atomic elements to make the description of the scene. They are organized into a fully-structured symbolic data network, named "Patchery Data Structure", with powerful retrieving facilities. In the rule-based analysis, all picture-processing operations are performed on this Patchery Data Structure rather than the raw image data. This enables the rule-based analysis to have a clear-cut scheme for modeling and control, and to perform various picture processing operations in high speed.

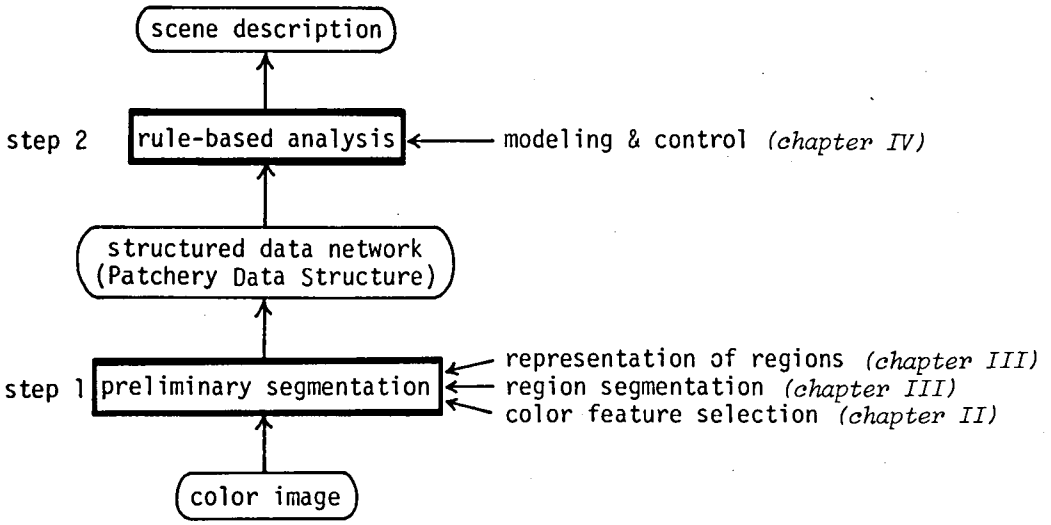


Figure 1-1. Two steps in region analysis; from an input color image to the scene description.

Rule-based analysis --- Figure 1-2 shows the schematic diagram of the rule-based architecture in our system. It employs both of the bottom-up and top-down control schemes. The knowledge of the task world is represented by two sets of rules: one is for the bottom-up process and the other for the top-down process. The rules for the bottom-up process make a plan as a rough interpretation of the scene. The rules for the top-down process make a detailed semantic description of the scene.

An approximate reasoning scheme is employed for plan evaluation to deal with the uncertainty which exists in both of knowledge and pictorial features. The plan manager controls the evaluation of plan.

The top-down analysis works in the framework of region growing. But the process is not a simple iteration of the labeling and merging operations. A scene description is built

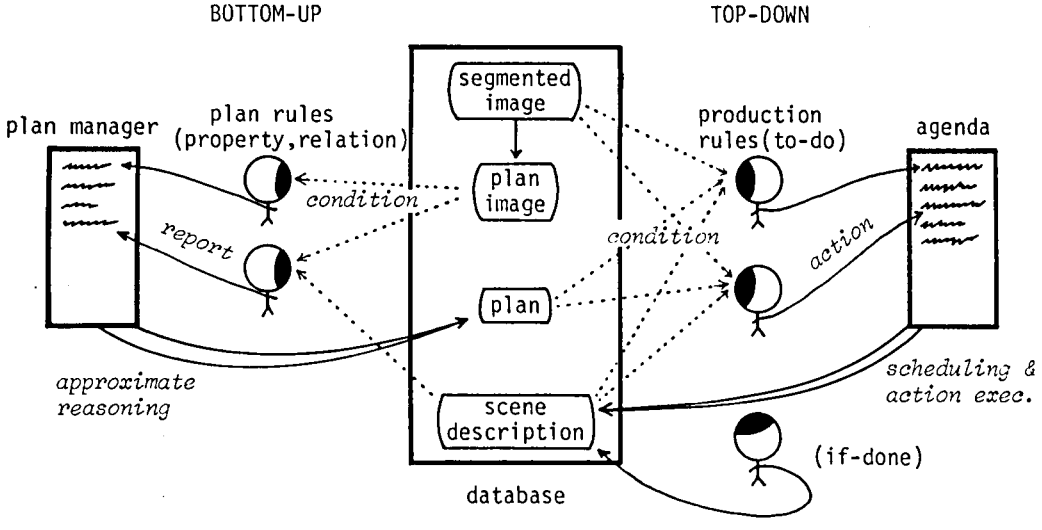


Figure 1-2. Schematic diagram of the rule-based analysis.

as the result of the top-down analysis. The knowledge used in the top-down process is represented as a set of production rules. The agenda controls the production system. Each production rule is a pair of a condition and an action. The "condition" is a fuzzy predicate [Lee and Chang, 1971]. It checks the state of the database and decides whether the associated action can be executed. Each "action" describes the operations to build the scene description. Every executable action is given a score to indicate its priority and is registered on the agenda. The action with the highest score is executed at each context of analysis, and as a result the database is changed. Production rules are activated again to examine the database. In order to reduce the computation, the agenda controls the activation of production rules according to the changes newly made in the database.

The analysis process completes when all regions are interpreted and assembled into the scene description.

I-3. Related Works

In this section we present a brief survey of analysis systems for natural scenes. Only complete systems are referred to here. The works relating to the individual problems such as color representation, segmentation, and region analysis are mentioned in the corresponding chapters. A survey of region analysis techniques in general can be found elsewhere [Kanade, 1978].

a) Barrow and Popplestone [1971] constructed a system which interpretes isolated simple objects, such as cup, spectacles,

cylinder, etc. The image data is first partitioned into a set of regions. The set of regions is then described in the form of a graph which represents properties of and relations between the regions. The description is matched against a set of models which describe typical views of objects. A limitation of this work is that it requires the regions to be meaningful ones. However, it is very difficult to obtain the meaningful partitions based only on pictorial features.

b) Preparata and Ray [1972] tried to interpret simple outdoor scenes by using a similar graph matching scheme to that of Barrow and Popplestone. In their work, the image is manually partitioned into regions to avoid difficulties in the automatic region segmentation.

c) Yakimovsky and Feldman [1973] integrated the segmentation and interpretation phases. The input image is preliminarily segmented into coherent regions. Labels of objects are assigned to each region by using the knowledge of the task world, and the regions which are assigned with same labels are merged. The knowledge is represented by a set of probabilities. The interpretation is performed by maximizing the joint probability that regions have correct labels. They successfully analyzed road scenes and X-ray images.

d) Tenenbaum and Barrow [1976] developed a scheme called Interpretation-Guided Segmentation to integrate the segmentation and interpretation phases. In their case, the knowledge is represented as a set of constraints and Waltz's filtering algorithm is employed to search a globally correct interpretation.

e) Rubin and Reddy [1977] represented the knowledge in the form of a pixel-level constraint network. All images that are admissible in a task world are precompiled into the network. Given an input image, the system searches through the network for a path which corresponds to the "best" match between the model and the image data.

f) Bajcsy and Lieberman [1974] analyzed simple outdoor scenes by using top-down control structure. The knowledge is embedded in the procedures which extract objects from image data.

g) Sloan [1977] employed a production system architecture to represent the knowledge of outdoor scenes. Each production rule is triggered by a certain pictorial feature. It simply rephrases the facts recorded in the database or tries to extract some objects from the image data in a top-down fashion.

h) Freuder [1977] represented the knowledge by a set of modular procedures and organized them into a semantic network. Each module has its own duty and it activates other modules when necessary. A simple "hammer" scene is used as the task world.

i) Riseman et al. [1977] tried to construct a scene analysis system named VISIONS. The knowledge is represented in a hierarchical structure with layers, such as objects, volumes, surfaces, regions, etc. At each level of the hierarchy, a hypothesis-and-test paradigm is used to construct a scene description from the image data.

There are two complementary control schemes, bottom-up and top-down, to organize a scene analysis system. We can categolize the

systems described above into two groups: a) through e) employ the bottom-up scheme, and f) through i) the top-down. The first group relies on a bottom-up control structure together with global optimization mechanisms. On the other hand, the second group utilizes a top-down control scheme with or without a bottom-up mechanism to trigger the top-down scheme. Notice the big difference that the mechanisms to search for a globally (sub)optimal solution are included in the first group, whereas the second group searches for one solution in a depth first manner. Our system described in this thesis employs both of the top-down control and bottom-up control schemes with optimizing mechanisms. It aims to combine the merits of the two complementary control schemes.

Chapter II

COLOR INFORMATION FOR REGION SEGMENTATION

II-1. The Problems

In color image processing, the color of a pixel is usually given as three values corresponding to the tristimulus values R (red), G (green), and B (blue). Various kinds of color features, such as intensity (D), saturation (S), and hue (H), can be calculated from {R, G, B} by using either linear or nonlinear transformations. Each color feature has its own characteristics. For instance, the set {D, S, H} is convenient for representing the human color perception; the set {Y, I, Q} is used to efficiently encode color information in TV signals; and the normalized color set {r, g, b} is convenient to represent the color plane.

It seems that in computer processing of color images, color features which were developed for other purposes have been used in different combinations for different purposes. Nevatia [1976] extended the Hueckel operator for color edge extraction. He stated that the result obtained using intensity (D=R+G+B) and normalized colors (r=R/D and g=G/D) was better than that obtained using R, G, and B. Ohlander [1975] employed nine redundant color features R, G, B, Y, I, Q, D, S, and H for color image segmentation. He reported that H was most useful and that Y, I, and Q were rarely used.

Kender [1976] presented a very careful discussion of the behavior of the linear and nonlinear color transformations used to obtain color features such as hue, saturation, and normalized color from R, G, and B. His discussion amounts to two points: (1) Nonlinear transformations such as hue, saturation, and normalized color have nonremovable singularities, near which a small perturbation of the input R, G, and B can cause a large jump in the transformed values; (2) the distribution of the nonlinearly transformed values can show spurious modes and gaps. For these reasons and from the computational point of view, he concluded that linear transformations such as Y, I, and Q would be preferable to nonlinear ones.

It is an interesting and important problem to find color features which are suited for the segmentation of color images by computer. One way to get such color features is to execute the segmentation by using various sets of color features and to compare the results. However, this allows us to examine only predefined sets of color features. In this chapter we attempt to derive a set of effective color features by systematic experiments in region segmentation. An Ohlander-type segmentation algorithm by recursive thresholding is employed as a tool for the experiments. At each step of segmenting a region, new color features are calculated for the pixels in that region by the Karhunen-Loeve transformation of the R, G, and B data. By analyzing the color features obtained in segmenting eight kinds of color pictures, we have found a set of effective color features. The effectiveness of our color feature set is proved by a comparative study with various other sets of color features which are commonly used in image analysis.

II-2. Selection of Effective Color Features

II-2-1. Segmentation Algorithm

First of all, we briefly describe a segmentation algorithm which is employed in the experiments. The basic scheme is almost the same as the segmentation algorithm described in chapter III. Figure 2-1 shows a schematic diagram of the segmentation algorithm. The basic idea of the process is as follows: The whole image is first partitioned into sub-images each of which is a connected region; then each sub-image is further partitioned if it is possible; and this process iterates. Because of the recursive nature of the algorithm, a picture stack is used to store the region masks. A region mask represents a connected region (the area without hatching in Fig. 2-1) which is to be examined for segmentation. The arrows with numbers shown in Fig. 2-1 represent the following operations.

(0) A mask corresponding to the whole image is placed at the bottom of the stack.

(1) One mask is taken from the top of the stack. Let S denote the region represented by the mask (the area without hatching).

(2) Histograms of color features in the region S are computed.

(3) If any of the histograms shows conspicuous peaks, a pair of cutoff values which separate the peak in the histogram are determined at the position of valleys, and the image of the color feature corresponding to that histogram is thresholded using the cutoff values; thus the region S is partitioned. Otherwise, region S is not partitioned further.

(4) Connected regions are extracted. For each connected region, a region mask is generated, and it is pushed down on the stack.

Operations (1)-(4) are iterated until the picture stack becomes empty. In the operation (3), the cutoff values are selected by the following two criteria: Candidate cutoff values are selected by evaluating the shape of peaks on the histograms; bad cutoff values are rejected by verifying in the image the compactness of the spatial distribution of the pixels belonging to the peak determined by the pair of cutoff values.

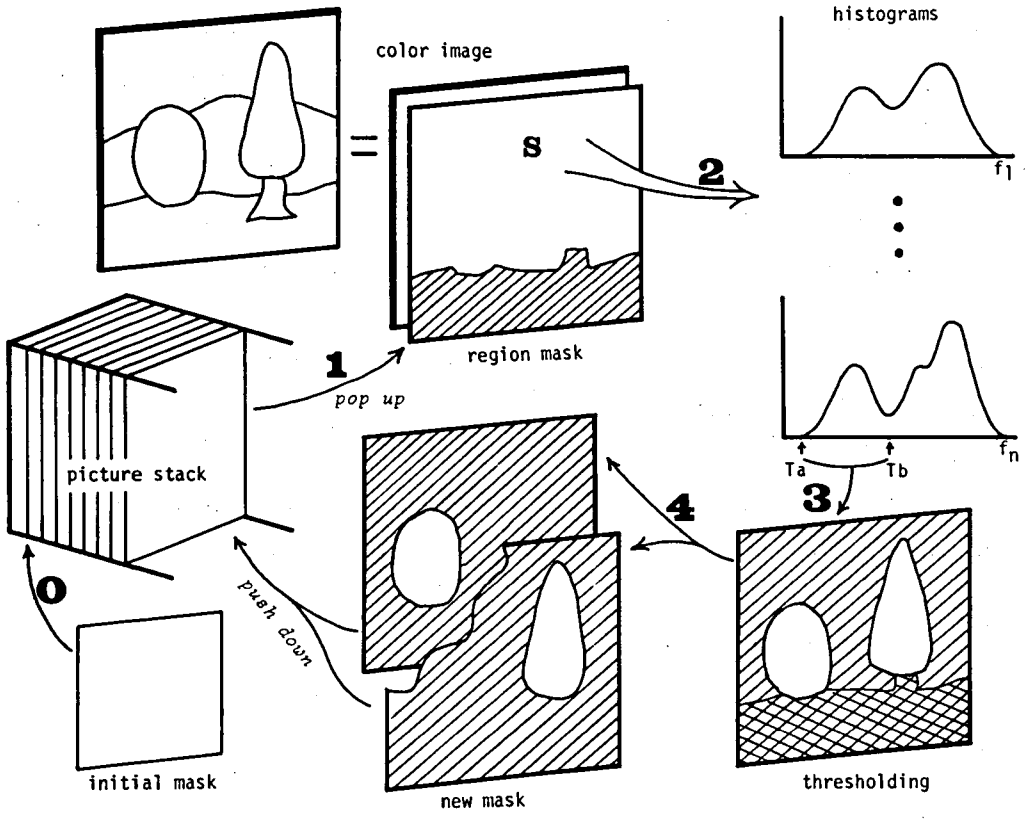


Figure 2-1. Schematic diagram of the segmentation algorithm.

II-2-2. Computation of Color Features Using the K. L. Transformation

At the step (2) in Fig. 2-1, it is important to know the histograms of what color features are to be computed. We could use features such as R, G, and B throughout the segmentation process. However, the color feature which has the deepest valley on its histogram and that which has the largest discriminant power to separate the clusters in a given region need not be the same. In feature selection of the pattern recognition theory, a feature is said to have large discriminant power if its variance is large. Thus we tried to derive color features with large discriminant power by using the Karhunen-Loeve (K. L.) transformation.

More specifically, let S be the region to be segmented, and let Σ be the covariance matrix of the distributions of R, G, and B in S. Let $\lambda 1$, $\lambda 2$, and $\lambda 3$ be the eigenvalues of Σ, and $\lambda 1 \geq \lambda 2 \geq \lambda 3$. Let $Wi=(w_{Ri}\ w_{Gi}\ w_{Bi})^t$ for i=1, 2, and 3 be the eigenvectors of Σ corresponding to λi, respectively. The color features X1, X2, and X3 are defined as

$$Xi = w_{Ri} \cdot R + w_{Gi} \cdot G + w_{Bi} \cdot B \quad (\| Wi \| = 1,\ i=1,2 \text{ and } 3) . \qquad (2\text{-}1)$$

It is well known that X1, X2, and X3 are uncorrelated, and X1 is the "best" feature in the sense that it has the largest variance (the value is $\lambda 1$). X2 is the best one among the orthogonal ones to X1. At each step of segmenting a region, three new color features X1, X2, and X3 are calculated for the pixels in that region and used to compute the histograms. We call this scheme "segmentation by the dynamic K. L. transformation".

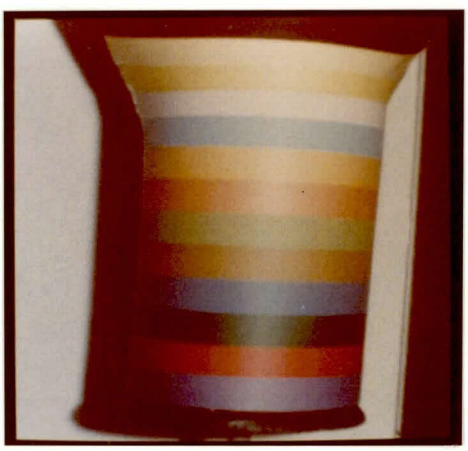
The eight scenes shown in Fig. 2-2 were used in the experiments. The names of the scenes are (a) cylinder, (b) building, (c) seaside, (d) girl, (e) room, (f) home, (g) auto, and (h) face. They were digitized with 256 x 256 spatial resolution and 6-bit density resolution for each of R, G, and B. Scenes (a), (b), and (c) in

Fig. 2-2 were digitized at Kyoto University. (d) is from Southern California University, and (e) through (h) are from Carnegie-Mellon University. Scene (a) is a cylinder with color stripes illuminated from the front. Scenes (e), (g), and (h) are the images which Ohlander used in his experiment [Ohlander, 1975], except that the size and density resolution are reduced for our system. Scene (f) is almost the same as Ohlander's "home" scene except that there are some clouds in the sky.

Figure 2-3 shows the results of segmentation by the dynamic K. L. transformation. We can notice that use of the "best" color features calculated adaptively at each step of segmenting a region gives satisfying results. In the cylinder scene, for example, the horizontal color stripes are separated almost completely, and the vertical cracks which split the color stripes vertically because of differences in intensity are relatively few. However, using the K. L. transformation on the fly requires costly computation and is not very practical. Our goal is to discover a set of color features with which we can achieve segmentations as good as those based on the dynamic K. L. transformation.

II-2-3. A Set of Effective Color Features

Table 2-1 shows the eigenvectors of Σ for the whole image of each of the eight color scenes in Fig. 2-2. It is interesting to note that W1 is approximately $(1/3\ 1/3\ 1/3)^t$ for every scene. W2 is dominated by $(1/2\ 0\ -1/2)^t$ or $(-1/2\ 0\ 1/2)^t$, and W3 by $(-1/4\ 1/2\ -1/4)^t$. Then it is possible to say that the three orthogonal color features, I1=(R+G+B)/3, I2=(R-B)/2 or (B-R)/2, and I3=(2G-R-B)/4, are important components representing color information.



(a) cylinder

(b) building

(c) seaside

(d) girl

Figure 2-2. Color scenes used in the experiments.

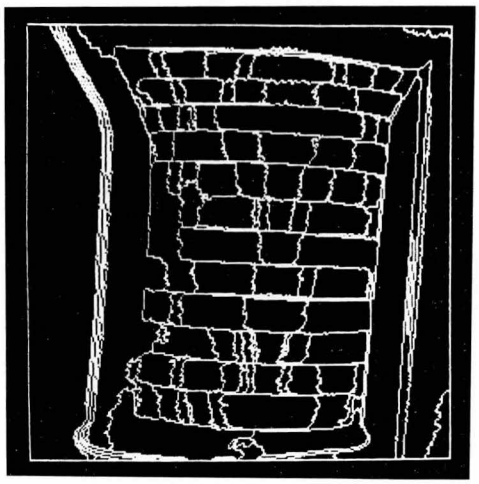
(e) room

(f) home

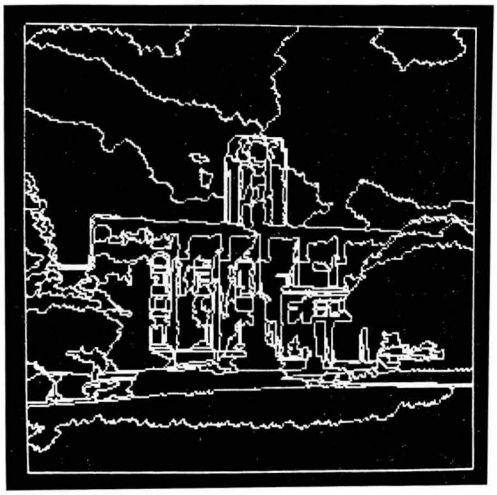
(g) auto

(h) face

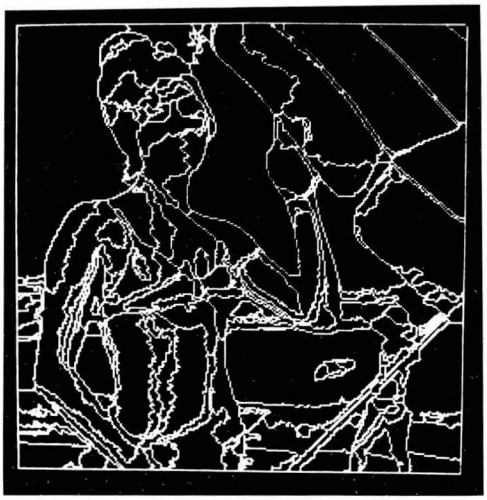
Figure 2-2. (continued.)



(a) cylinder



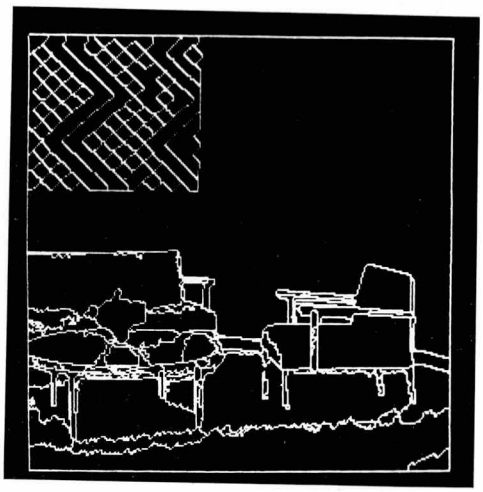
(b) building



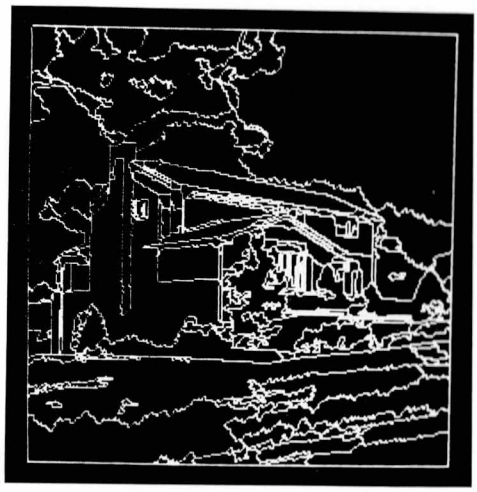
(c) seaside

(d) girl

Figure 2-3. Segmentation results by the dynamic K. L. transformation.



(e) room



(f) home

(g) auto

(h) face

Figure 2-3. (continued.)

Table 2-1. Eigenvectors of Σ for a whole image.

	w_{R1}	w_{G1}	w_{B1}	w_{R2}	w_{G2}	w_{B2}	w_{R3}	w_{G3}	w_{B3}
cylinder	0.269	0.363	0.367	0.469	0.095	-0.437	-0.308	0.461	-0.231
building	0.269	0.340	0.391	0.479	0.103	-0.418	-0.296	0.485	-0.219
seaside	0.258	0.380	0.362	-0.585	0.056	0.358	-0.176	0.464	-0.360
girl	0.336	0.354	0.309	-0.493	0.193	0.314	-0.094	0.474	-0.436
room	0.193	0.341	0.467	0.612	0.079	-0.310	-0.209	0.507	-0.284
home	0.197	0.328	0.476	0.492	0.180	-0.328	-0.313	0.484	-0.204
auto	0.304	0.317	0.378	0.239	0.309	-0.452	-0.514	0.450	0.036
face	0.175	0.411	0.414	0.523	0.128	-0.349	-0.295	0.416	-0.289

normalized by $w_G \geq 0$, $|w_R|+|w_G|+|w_B|=1$.

To prove this experimentally, we analyzed the linear combinations of R, G, and B, which are used to find the cutoff values for thresholding in segmentation by the dynamic K. L. transformation for the eight scenes. Only those cases are examined in which regions with an area larger than 1000 are split into regions larger than 200. The number of linear combinations thus gathered is 109 in this experiment. Those weight vectors are plotted on a w_R-w_B plane as shown in Fig. 2-4. The weight vectors have been normalized so that $w_G \geq 0$ and $|w_R|+|w_G|+|w_B|=1$. For simplicity, each vector is plotted with the first letter of the name of the scene for which that color feature was used. The weight vectors corresponding to {I1, I2, I3}, {Y, I, Q}, {X, Y, Z}, {U, V, W}, and {R, G, B} are indicated for reference. Contour lines are drawn to show equidistance from the reference points I1=(R+G+B)/3, I2=(R-B)/2 or (B-R)/2, and I3=(2G-R-B)/4. The curves are defined by

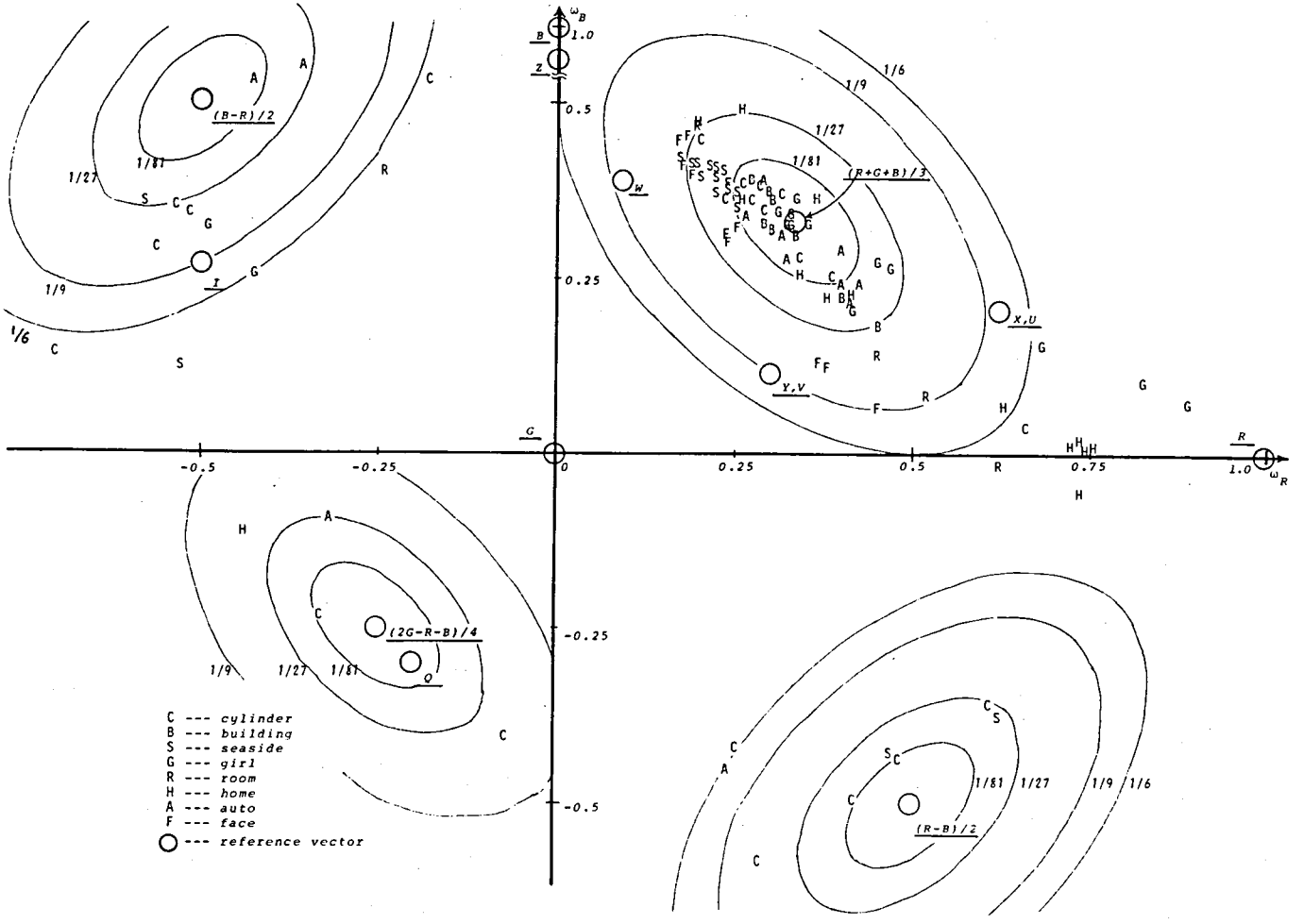


Figure 2-4. Plots of the weight vectors of 109 color features used in segmentation by the dynamic K. L. transformation.

$$(w_R-1/3)^2+(w_G-1/3)^2+(w_B-1/3)^2 = e \quad \text{(around I1)},$$
$$(w_R-1/2)^2+ w_G^2+(w_B+1/2)^2 = e \quad \text{(around I2)},$$
$$(w_R+1/2)^2+ w_G^2+(w_B-1/2)^2 = e \quad \text{(around I2)},$$
$$(w_R+1/4)^2+(w_G-1/2)^2+(w_B+1/4)^2 = e \quad \text{(around I3)},$$
$$(e=1/81,\ 1/27,\ 1/9,\ 1/6). \quad (2\text{-}2)$$

Color features in the first quadrant have weight vectors such that w_R, w_G, $w_B>0$. They correspond mainly to the intensity component and I1=(R+G+B)/3 is the most typical feature of this quadrant. In the second and fourth quadrants, w_R and w_B have opposite signs, and the color features in these quadrants represent the difference of the R and B components. Most color features are in the first quadrant. This means that the intensity is the most important feature even in color image processing.

The weight vector of I1 is nearly at the center of the weight vectors in the first quadrant as shown in Fig. 2-4. I2 can be regarded as being at the center of the weight vectors in the second and fourth quadrants. I3 will be a typical color feature in the third quadrant. Thus, it is possible to assume that every weight vector in the four quadrants can be approximated by the weight vectors of the three color features, I1, I2, and I3. The numbers of weight vectors in the first, second/fourth, and third quadrants in Fig. 2-4 are 83, 22, and 4, respectively. So, I1, I2, and I3 are assumed to be significant in this order.

II-2-4. Segmentation by the New Color Features

We have performed the following experiments to verify the arguments described above.

(1) Segmentation by using three color features I1, I2', and I3'

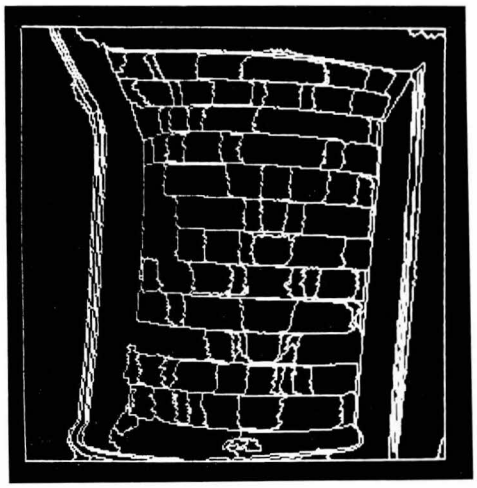
Figure 2-5 shows the results obtained by using the set of three fixed color features I1, I2'=(R-B), and I3'=(2G-R-B)/2. They seem not to be degraded compared with those obtained by the dynamic K. L. transformation. This verifies the argument that the set of three color features I1, I2', and I3' can approximate all color features which are calculated as the "best" ones at each important step in segmenting the eight color scenes.

(2) Segmentation by using two color features I1 and I2'

In Fig. 2-4 the number of color features in the third quadrant is only four. They are considerably fewer than those in the other quadrants. Thus the omission of I3' will not significantly affect the quality of the segmentation. This is verified by the results shown in Fig. 2-6 which are obtained by using only the two color features, I1 and I2'. A picture indicating the missing boundaries from Fig. 2-5 to Fig. 2-6 is shown in Fig. 2-7 for the cylinder scene in order to help visual comparison. Even in the cylinder scene which has two weight vectors in the third quadrant, the results of Fig. 2-5-a and Fig. 2-6-a are almost the same except that the fifth and sixth color stripes from top, golden yellow and orange, are not separated in Fig. 2-6-a.

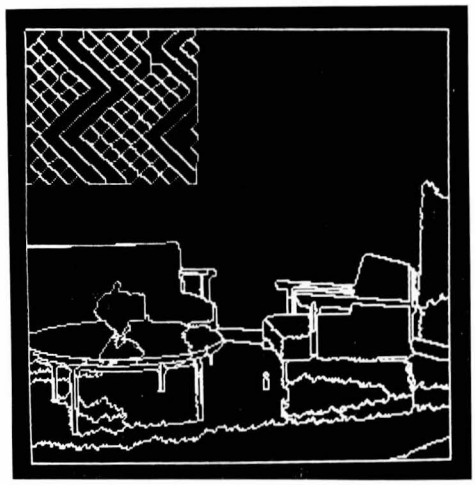
(3) Segmentation by using only one color feature I1

What quality of segmentation can be achieved by using only one color feature I1, i.e. intensity information? In the case of the cylinder scene, the fraction of the number of weight vectors in the first quadrant is 10 out of 22 weight vectors, and there are 10 weight vectors in the second and fourth quadrants. Therefore the quality of the segmentation for the cylinder scene will be considerably degraded by omitting the color feature I2'. For the



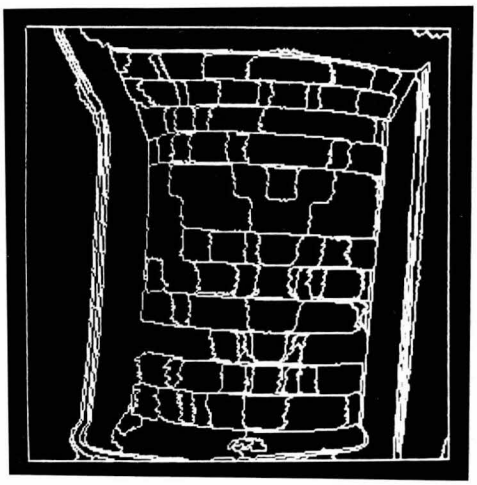
(a) cylinder

(b) home



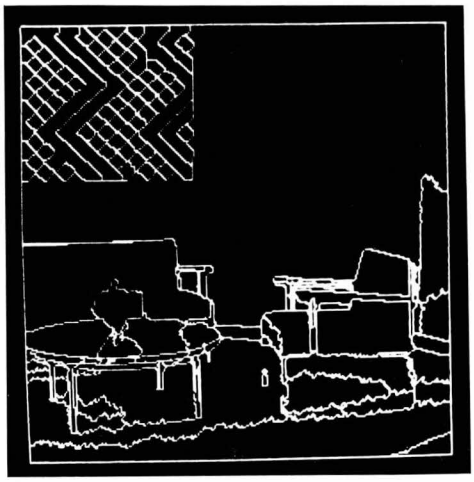
(c) room

Figure 2-5. Segmentation results by using I1, I2', and I3'. *I1=(R+G+B)/3, I2'=(R-B), I3'=(2G-R-B)/2.*



(a) cylinder

(b) home



(c) room

Figure 2-6. Segmentation results by using I1 and I2'. *I1=(R+G+B)/3, I2'=(R-B).*

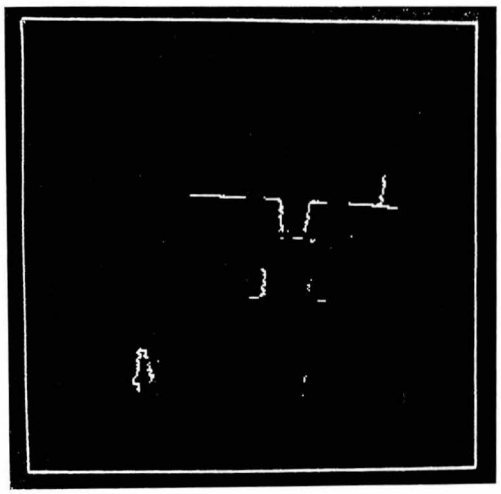
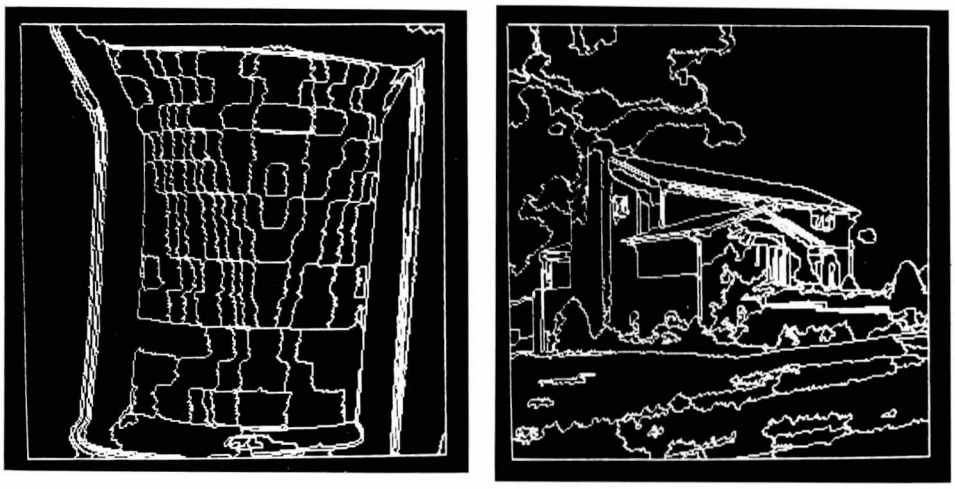
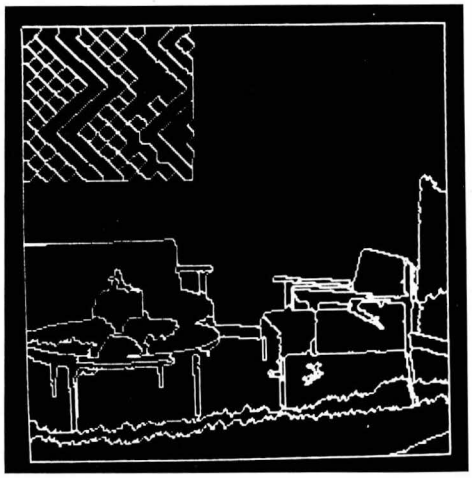


Figure 2-7. Missing boundaries from Fig. 2-5-a to Fig. 2-6-a.



(a) cylinder (b) home

Figure 2-8. Segmentation results by using only I1.



(c) room

Figure 2-8. (continued.)

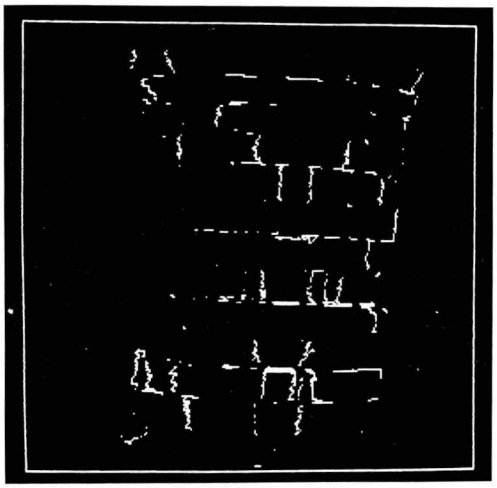


Figure 2-9. Missing boundaries from Figs. 2-5-a to 2-8-a.

other scenes, however, only a few weight vectors are in the second and fourth quadrants and the degradation will not be so significant. The results obtained by using only I1 are shown in Fig. 2-8. Figure 2-9 shows the missing boundaries from Fig. 2-5-a to Fig. 2-8-a. Separation of the color stripes in the cylinder scene is very poor as expected. It is a natural consequence of losing the distinguishability of color difference by omitting I2' and I3'. In the case of the room scene, however, the quality of segmentation is not so much degraded though there appears to be a little missegmentation. As for the home scene in Fig. 2-8-b, there is no missegmentation, in the sense that the regions which should be segmented are all segmented.

II-3. Comparison of Color Features

II-3-1. Segmentation by Various Sets of Color Features

In order to test the effectiveness of the color feature set obtained in the previous section, the eight scenes used in the previous experiments were also segmented using seven sets of color features which are commonly used in image analysis. The sets are {R, G, B}, {X, Y, Z}, {Y, I, Q}, {L, a, b}, {U*, V*, W*}, {I1, S, H}, and {I1, r, g}. R, G, and B are the original tristimulus values. X, Y, and Z correspond to the C.I.E. X-Y-Z primary color coordinate system. Y-I-Q is the color coordinate system for television signals. The L-a-b color coordinate system is designed to agree with the Munsell color system. U*-V*-W* is designed to obtain a color solid for which unit shifts in luminance and chrominance are uniformly perceptible [Platt, 1978]. I1, S, and H are the intensity,

saturation, and hue, respectively. r and g are the normalized colors. Other sets of color features such as {U, V, W}, {S, Θ, W*}, and {u, v, V} are not examined because they are similar to the sets {X, Y, Z}, {I1, S, H}, and {I1, r, g}, respectively.

{Y, I, Q} and {X, Y, Z} were calculated from {R, G, B} in our experiments by

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.500 & -0.230 & -0.270 \\ 0.202 & -0.500 & 0.298 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}, \tag{2-3}$$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.618 & 0.177 & 0.205 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.056 & 0.944 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}. \tag{2-4}$$

The transformation matrices are not the standard ones; the weights for I, Q, X, and Z are rescaled to normalize the range of the transformed values to be the same as the original R, G, and B. {L, a, b} and {U*, V*, W*} are defined as

$$\begin{aligned} L = W^* &= 25(100Y/Y0)^{1/3}-16 , \\ a &= 500[(X/X0)^{1/3}-(Y/Y0)^{1/3}] , \\ b &= 200[(Y/Y0)^{1/3}-(Z/Z0)1/3] , \\ U^* &= 13(W^*)(u-u0) , \\ V^* &= 13(W^*)(v-v0) , \end{aligned} \tag{2-5}$$

where u0=0.199, v0=0.308, u=4X/(X+15Y+3Z), v=6Y/(X+15Y+3Z), and X0, Y0, Z0 are the X-Y-Z values for the reference white. Normalized colors, intensity, saturation, and hue are obtained as follows:

$$r=R/(R+G+B),\ g=G/(R+G+B),\ b=B/(R+G+B),$$
$$I1=(R+G+B)/3,$$
$$S = 1-3(\min(r,g,b)),$$
$$H = \arctan2(\sqrt{3}(G-B),(2R-G-B)). \qquad (2\text{-}6)$$

There are two problems in using these color features for region segmentation. One is the instability of nonlinear transformations. Normalized color, U*, V*, and saturation become unstable and meaningless when R+G+B is small. Therefore, in segmenting a region they are not used to compute histograms if R+G+B is less than 30. Hue is unstable when saturation is near zero, and is not used if S x (R+G+B) is less than 9. The other problem is caused by the fact that the input R, G, and B data are digitized. The histograms of the transformed values from digital input may have a comb-like structure. In order to avoid this, the input R, G, and B values are "undigitized" by adding a random number uniformly selected from the unit interval [Kender,1976].

Figures 2-10 through 2-16 show the results of segmentation obtained by using the seven sets of color features. Comparison of these results will be given in the succeeding section.

II-3-2. Comparison of Color Features

The effectiveness of a set of color features used in the segmentation process can be evaluated in terms of the quality of segmentation results and the behavior of the transformation from the input tristimulus values R, G, and B.

Evaluation of the quality of segmentation results is very difficult. No quantative evaluation procedure has been established for the segmentation of natural scenes. We adopted "eyeballs" as the most reliable tool at present. Pictures which indicate the difference

between a pair of segmentation results are generated to help visual comparison as shown in Figs. 2-7 and 2-9. We think that the under-segmentation (failures in splitting the regions that must be separated) affects later processings more seriously than the over-segmentation (failures which split the regions that need not be separated). So the evaluation criteria are set more severely against under-segmentations than over-segmentations.

{R, G, B}

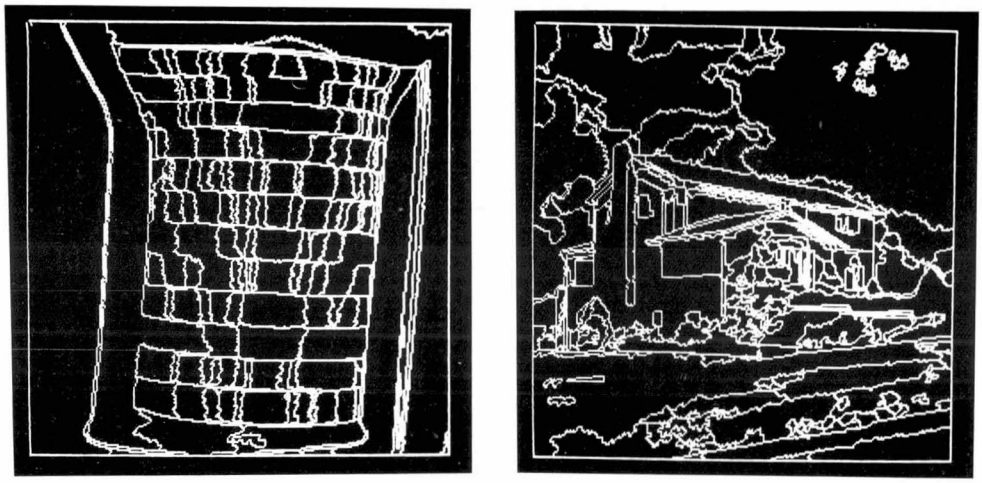
Use of the color feature set {R, G, B} for segmentation requires no transformation. But, R, G, and B have a strong factor of intensity and are heavily correlated. Thus, spurious segmentations tend to occur because of differences in intensity. This tendency is clearly observed in Fig. 2-10-a. It is noted that the vertical splitting of color stripes occurs more frequently in Fig. 2-10-a than in Fig. 2-5-a which is segmented by using the set of I1, I2', and I3'.

{X, Y, Z}

The weight vectors of X, Y, and Z are located in the first quadrant of the w_R-w_B plane as shown in Fig. 2-4; i.e., all have a strong factor of intensity. This implies that the use of this set will results in the similar segmentation to that obtained by using R, G, and B (see Figs. 2-10-a and 2-11-a). The separation of the color stripes in Fig. 2-11-a is worse than Fig. 2-10-a, because the weight vectors of X and Y are closer to the white point (I1) in Fig. 2-4 than those of R and G.

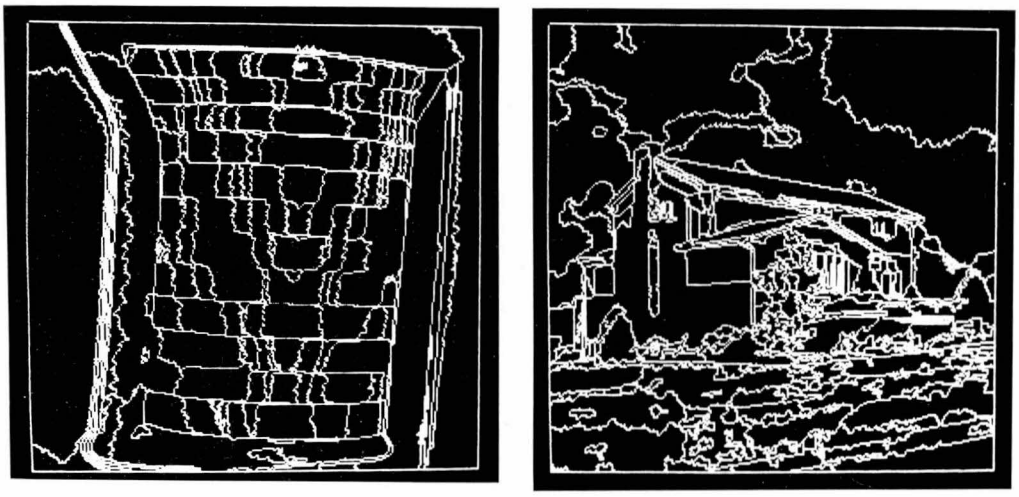
{Y, I, Q}

Y, I, and Q are in the first, second, and third quadrants, respectively (see Fig. 2-4). The segmentation results obtained by



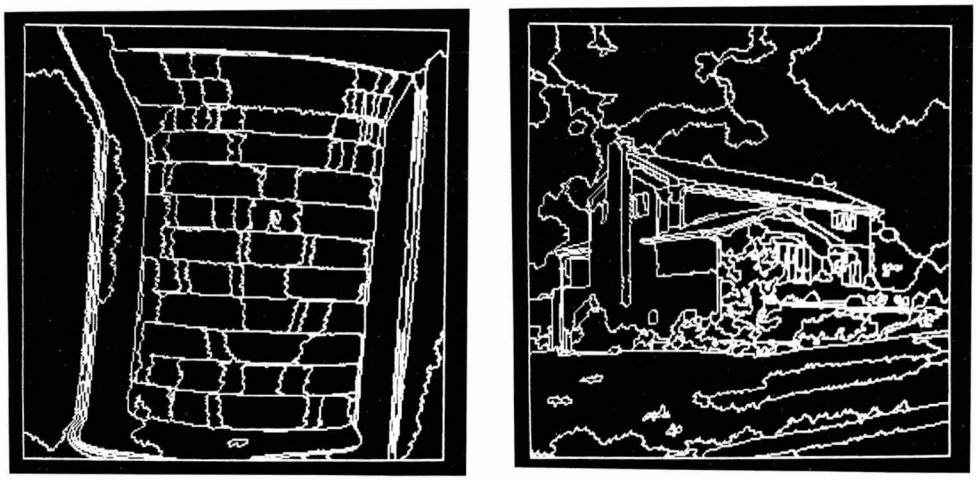
(a) cylinder (b) home

Figure 2-10. Segmentation results by using R, G, and B.



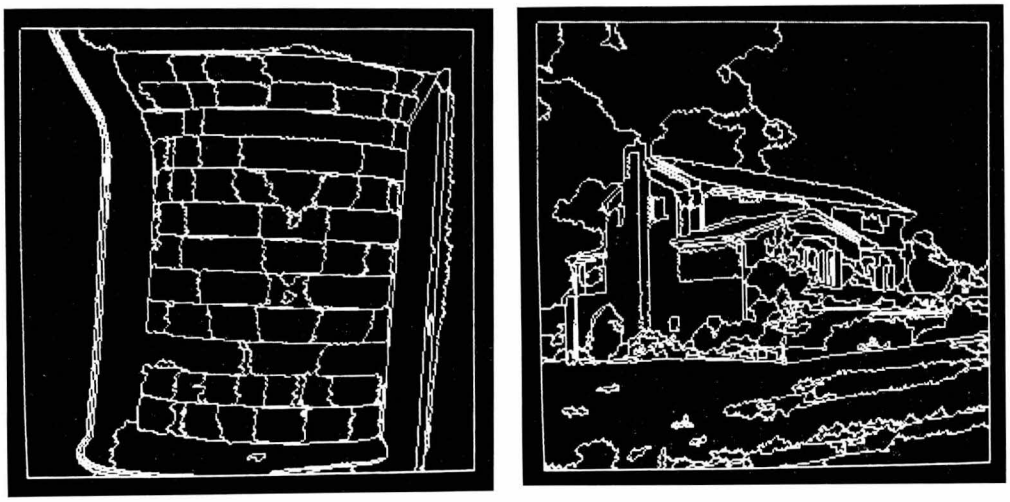
(a) cylinder (b) home

Figure 2-11. Segmentation results by using X, Y, and Z.



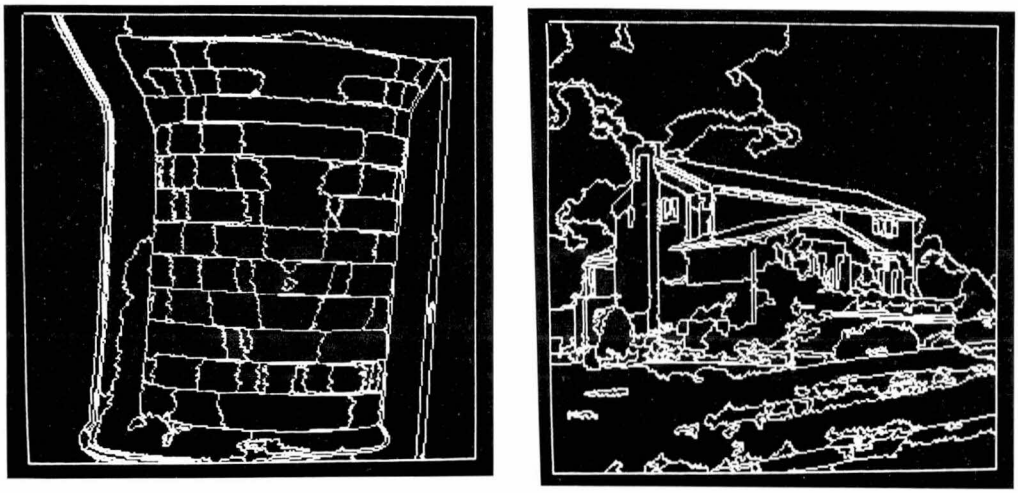
(a) cylinder (b) home

Figure 2-12. Segmentation results by using Y, I, and Q.



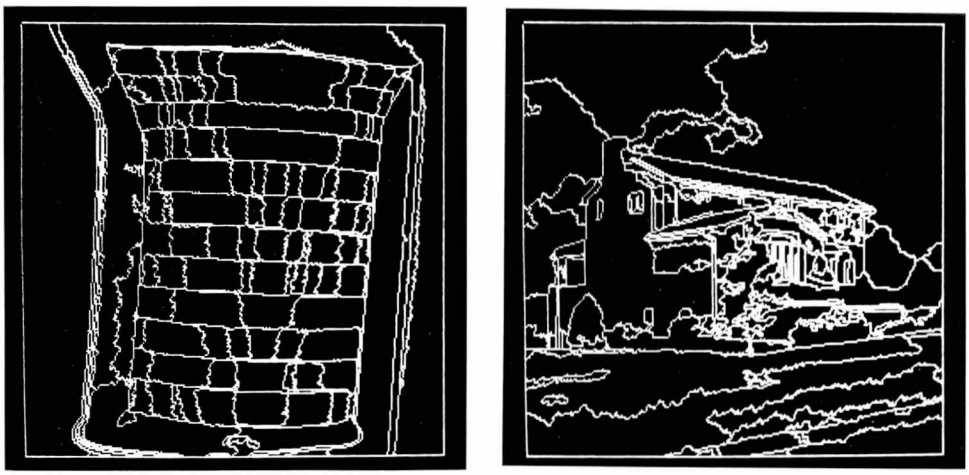
(a) cylinder (b) home

Figure 2-13. Segmentation results by using L, a, and b.



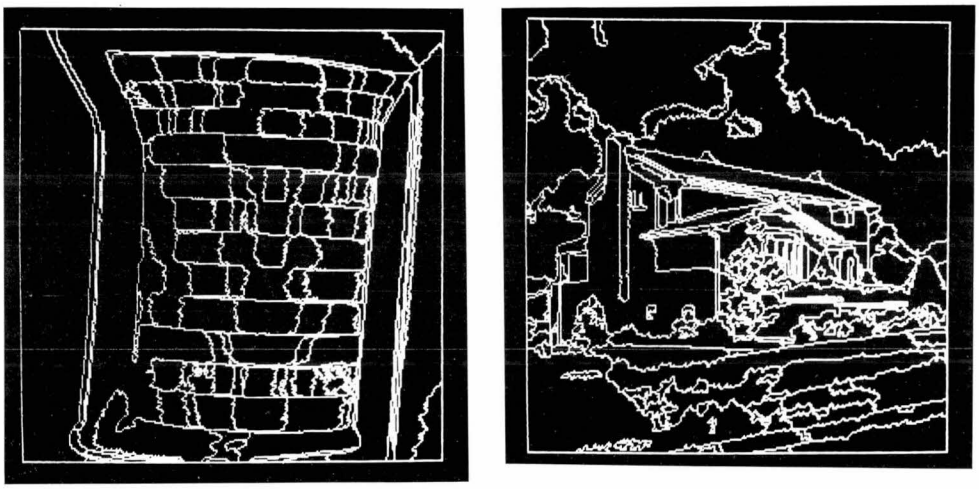
(a) cylinder (b) home

Figure 2-14. Segmentation results by using U*, V*, and W*.



(a) cylinder (b) home

Figure 2-15. Segmentation results by using I1, r, and g.



(a) cylinder (b) home

Figure 2-16. Segmentation results by using I1, S, and H.

using {Y, I, Q} (Fig. 2-12) are similar to those by {I1, I2', I3'} (Fig. 2-5). In Fig. 2-12-a the uppermost two color stripes are not separated, while they are separated in Fig. 2-5-a. The reason is that the weight vector of color feature I, located at a biased position in the fourth quadrant, is not a good approximation of the color features in the second quadrant. A more important difference between {Y, I, Q} and {I1, I2', I3'} is in the calculation of these features from {R, G, B}. The computation of {Y, I, Q} from {R, G, B} needs floating-point multiplications. Furthermore, there is the possibility that spurious combs appear in the histograms of Y, I, and Q. In contrast, all coefficients of the transformation from {R, G, B} to {I1, I2', I3'} are of the form 1/(integer). This means that a comb-like structure never appears in the histograms of I1, I2', and I3'. The calculation of {I1, I2', I3'} from R, G, B is far simpler than that of {Y, I, Q}. It can be performed by addition and subtraction of integer numbers together with shifting or simple table-lookup operations for scaling.

{L, a, b} and {U*, V*, W*}

Figure 2-13 shows the result obtained by using the set {L, a, b}, and Fig. 2-14 is the result by using the set {U*, V*, W*}. Both color coordinate systems use cube-root features for luminance as shown in Eq. 2-5. This results in the good performance in separation of the color stripes at the left side of the cylinder where intensity is dark and gradually changes. L, a, and b are based on the Y, (X-Y), and (Y-Z) color features which are located in the first, third, and fourth quadrant, respectively. On the other hand, U*, V*, and W* are based on the u-v-V normalized color coordinate system which is derived from the U-V-W system, and U, V, and W are all located in the first quadrant in Fig. 2-4. This causes the missegmentation at the border of pale-yellow and yellow stripes and the missegmentation at the border of golden-yellow and orange stripes in the strongly

illuminated part of the cylinder surface in Fig. 2-14-a, while in Fig. 2-13-a all color stripes are separated clearly. In the case of the home scene (and in the other scenes), we do not observe significant differences among the segmentation results obtained by these two sets of features and the other sets of features.

{I1, r, g} and {I1, S, H}

The set of color features {I1, r, g} produces the result shown in Fig. 2-15. The use of color features normalized by intensity results in good segmentation at the dark part of the cylinder scene as well as the results obtained by using {L, a, b} or {U*, V*, W*}. Highly illuminated part of the border between the pale-yellow and yellow stripes is not separated as in the result for {U*, V*, W*}. Figure 2-16 is the result for {I1, S, H}. It seems to be degraded than that obtained by using {I1, r, g} shown in Fig. 2-15. One reason is that the hue can be meaningful only in limited cases. From the computational point of view, these nonlinear transformations incur far more cost than linear transformations.

Table 2-2 shows the variances, σ_{I1}^2, σ_{I2}^2, and σ_{I3}^2, of the three components of a color image represented in the I1-I2-I3 color space. The variances are scaled so that $\sigma_{I1}^2+\sigma_{I2}^2+\sigma_{I3}^2=100$ for each color image. We notice a relation of $\sigma_{I1}^2>\sigma_{I2}^2>\sigma_{I3}^2$ for every color image. This relation corresponds to the fact that (the number of color features in the first quadrant) > (the number of color features in the second/fourth quadrant) > (the number of color features in the third quadrant) in Fig. 2-4. Thus, it can be said that color features with larger variance are more useful in region segmentation of a color image.

Our experiments also say that chromatic information is not always important for the segmentation process even in case of colorful scenes which have large variance in the chromatic

Table 2-2. Variances of I1, I2, and I3 images.

	σ_{I1}^2	σ_{I2}^2	σ_{I3}^2
cylinder	92.4	6.5	1.1
building	97.0	2.8	0.1
seaside	80.6	17.0	2.4
girl	85.8	10.4	3.8
room	75.2	22.7	2.2
home	76.7	19.5	3.8
auto	89.9	6.4	3.7
face	87.7	9.6	2.8

scaled by $\sigma_{I1}^2+\sigma_{I2}^2+\sigma_{I3}^2=100$.

components. We think that the usefulness of a color feature is greatly influenced by the structure of the color scenes to be segmented. For instance, the variance of I2 is only 6.5 in the cylinder scene where I2 plays an important role in segmentation, while the variance of I2 is 19.5 in the home scene which was segmented well by using I1 alone. This phenomenon can be explained by the difference in the structures of the cylinder and home scenes. The cylinder scene consists of a curved surface, while the home scene includes mainly planar objects. The intensity gradually changes on the curved surfaces, and does not work as a useful feature for segmenting the color stripes across it. This causes the chromatic information was used frequently in the segmentation of the cylinder scene.

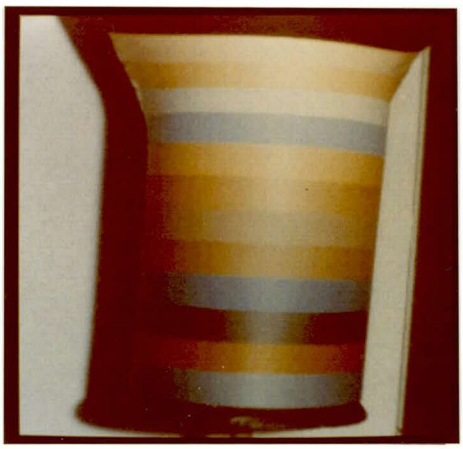
II-4. Two Component Representation of Color Images

Table 2-3 shows the eigenvalues, $\lambda 1$, $\lambda 2$, and $\lambda 3$, of the covariance matrix Σ obtained for the R, G, and B data in the whole image of each color scene. The values have been scaled such that $\lambda 1+\lambda 2+\lambda 3=100$. $\lambda 3$ is very small for every scene; the maximum value is 3.6. This implies that each color image can be approximated by two features X1 and X2 with a mean-square error of 3.6 at maximum: X1 and X2 are the linear combinations of R, G, and B with weights $\lambda 1$ and $\lambda 2$, respectively. We tried to compose color images from only two features X1 and X2 by using the imcomplete inverse of the K. L. transformation.

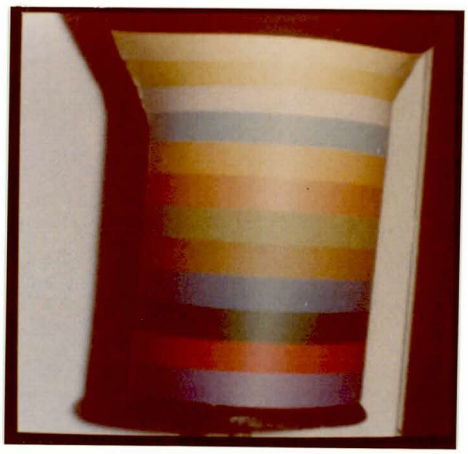
Table 2-3. Eigenvalues of Σ for a whole image.

	$\lambda 1$	$\lambda 2$	$\lambda 3$
cylinder	94.0	5.3	0.7
building	99.2	0.7	0.1
seaside	82.3	16.7	1.0
girl	86.0	11.7	2.3
room	81.8	16.1	2.1
home	84.2	12.1	3.6
auto	90.7	7.7	1.7
face	97.4	2.2	0.4

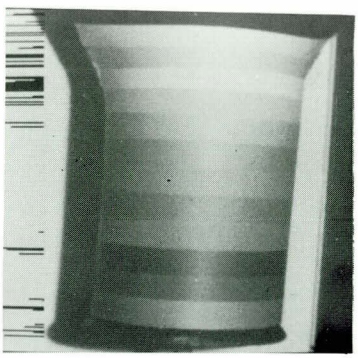
scaled by $\lambda 1 + \lambda 2 + \lambda 3 = 100$.



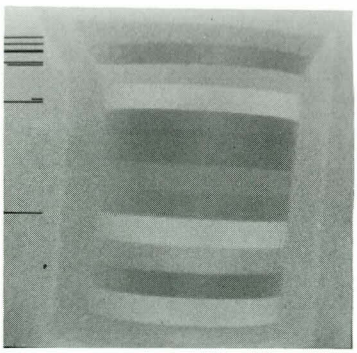
(a) reproduced image



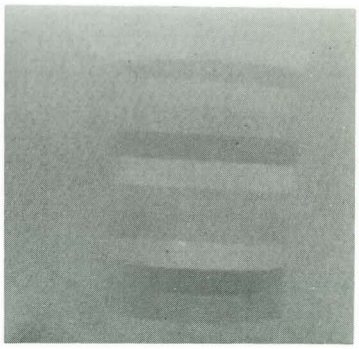
(b) original image



(c) image of X1



(d) image of X2



(e) image of X3

Figure 2-17. Reproduction of color images by using X1 and X2: Example 1. Cylinder.

(a) reproduced image

(b) original image

(c) image of X1

(d) image of X2

(e) image of X3

Figure 2-18. Reproduction of color images by using X1 and X2: Example 2. Girl.

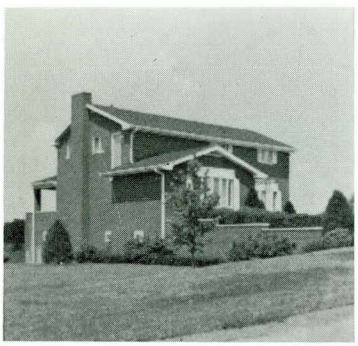
(a) reproduced image (b) original image

(c) image of X1 (d) image of X2 (e) image of X3

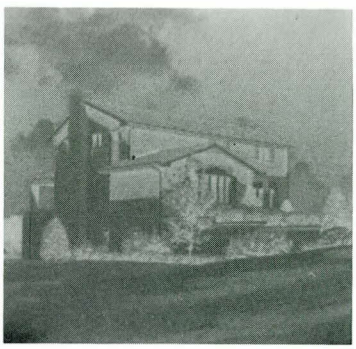
Figure 2-19. Reproduction of color images by using X1 and X2: Example 3. Room.

(a) reproduced image

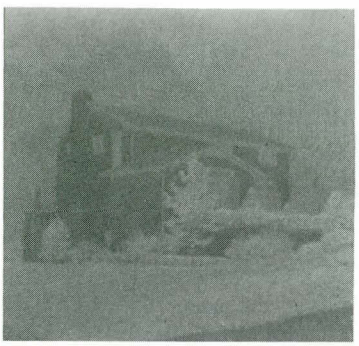
(b) original image



(c) image of X1



(d) image of X2



(e) image of X3

Figure 2-20. Reproduction of color images by using X1 and X2: Example 4. Home.

Let W be the 3 x 3 matrix for the K. L. transformation of a color image.

$$(X1\ X2\ X3)^t = W\ (R\ G\ B)^t \quad . \tag{2-7}$$

Of course this transformation is reversible and

$$(R\ G\ B)^t = W^{-1}(X1\ X2\ X3)^t \quad . \tag{2-8}$$

Now suppose we fix X3 to be M3 throughout the image, since the variance of X3 is small. Here, M3 is the mean value of X3 for the whole image. Then we can consider a reproduced color image by this inverse mapping from the two features X1, X2, and constant M3 for X3. The color components of each pixel are given by

$$(R'\ G'\ B')^t = W^{-1}(X1\ X2\ M3)^t \quad . \tag{2-9}$$

We compared the color images defined by R', G', and B' with the original color images. Figures 2-17 through 2-20 show the results of the experiments. In each figure, (a) is the reproduced color image and (b) is the original color image. Images (c), (d), and (e) represent X1, X2, and X3, respectively. The following two facts were observed: (1) Although the R'-G'-B' color images are composed by using only two spectral features, they are good reproductions of the original color images. (2) The clarity of color in a small area in the color image tends to be heavily degraded. The first fact means that the color information in the scenes we have used is almost two dimensional. The second fact can be explained by examining the images corresponding to X3. The remarkable low contrast is due to its small variance. However, there exist noticable small areas with different gray value from the average. These areas have vivid colors, and they can not be fully represented by the two principal features X1 and X2. Thus, their colors are spoiled by neglecting X3. To sum up, color

images can be represented by using only two spectral features at the cost of spoiling the clarity of the colors of small areas. The experiments by Land [Land, 1959] tells us that the colors in natural images are "perceptually" almost two dimensional; our eyes can perceive full colors by mixing two spectral stimuli in the context of natural images. The experiment described here shows that the color in natural scenes are "physically" almost two dimensional.

II-5. Conclusion

We have considered the role of color information in the region segmentation process. By means of systematic experiments in region segmentation, we found a set of effective color features I1=(R+G+B)/3, I2'=(R-B), and I3'=(2G-R-B)/2. The three features are significant in this order and in many cases good segmentations can be achieved by using only the first two. The transformation to derive them from the R, G, and B data is simple and it does not behave badly even when digitized input is used.

Comparisons are made experimentally between various sets of color features which are commonly used in image analysis. The characteristics of each set can be observed through the comparative experiments. The difference among segmentation results clearly appears in the case of the cylinder scene in which the separation of the color stripes is difficult. The color feature sets {L, a, b} and {I1, I2', I3'} give good results in our experiments. But, in many other scenes, no significant difference is observed among the results obtained by using the eight sets of color features. This seems to be

closely related to the fact that the color information in natural scenes is almost two dimensional (intensity and one chromatic feature), which was shown in the experiment of color image reproduction by using only two color features. That is, every set of color features can represent the color information for many scenes with a fairly large margin and therefore can provide enough information for region segmentation. When different sets of color features make little difference in segmentation of a scene, the calculation involved in the coordinate transformation from the R-G-B system becomes an important factor to consider the effectiveness of the color coordinate system for region segmentation. The set of color features derived in this chapter is good in this point as well as in the segmentation result. We think that it is a useful color feature set for color image segmentation.

In this chapter, the effectiveness of color feature set was discussed in the framework of region splitting. The results will be valid in other domains of color image processing such as edge extraction.

Chapter III

PRELIMINARY SEGMENTATION OF COLOR IMAGES

III-1. The Problems

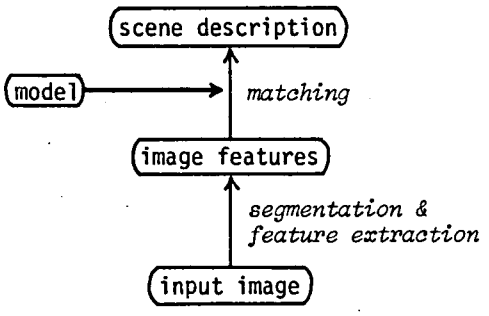
III-1-1. Nonpurposive Segmentation

We describe two topics in this chapter. One is the segmentation of a color image into a set of regions based on spectral information. The other is the organization of the segmentation result into a symbolic data structure.

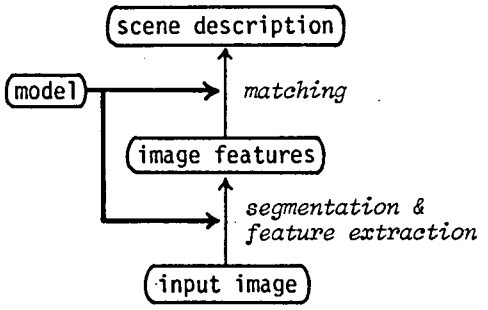
Generally speaking, two processes are necessary for analyzing image patterns: (1) The low-level process which performs segmentation of input image and extraction of useful features from the segmented image; (2) the higher-level process which performs semantic analysis of the image patterns based on the features extracted in the low-level process. In order to achieve good performance in the segmentation, it has been recognized as being useful to import task specific knowledge into the low-level processing. For this, we can consider the following four schemes.

(a) Basic scheme

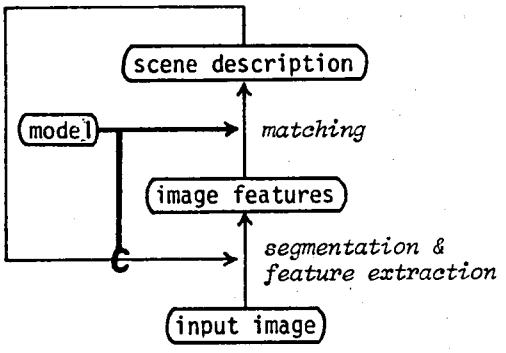
Figure 3-1-a illustrates the basic scheme. First, segmentation and feature extraction are performed on an input image. Secondly, the



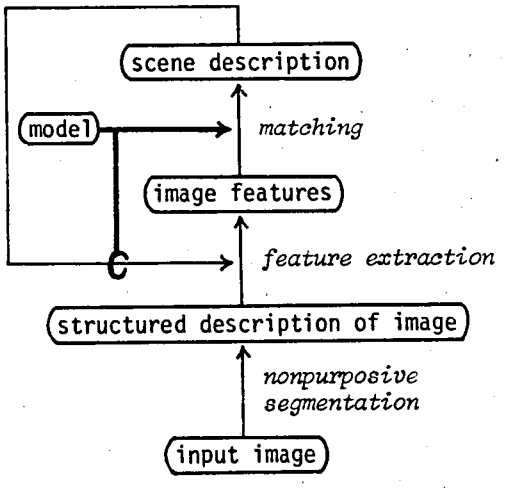
(a) basic scheme



(b) top-down scheme



(c) feedback scheme



(d) nonpurposive segmentation scheme

Figure 3-1. Four schemes importing knowledge into image analysis.

extracted features are matched with the models which describe the knowledge about the patterns to be analyzed. The task specific knowledge is used only in the higher-level matching process. When the patterns to be analyzed are complex, it is difficult to obtain a "correct" segmentation, and the feature extraction becomes unreliable.

(b) Top-down scheme

The task specific knowledge is imported into the low-level process as well as the higher-level process as shown in Fig. 3-1-b. The segmentation process can be tuned up to the patterns to be analyzed. This scheme works well when variation of input patterns is limited. But, some defects are inevitable. First, the segmentation algorithms in this scheme are special purpose ones and they often turn out to be powerless when applied to the task slightly different from the one for which the algorithms were originally designed. Secondly, it is defficult to apply this scheme to the tasks in which sufficient a priori knowledge about input patterns is not available.

(c) Feedback scheme

When the input patterns are very complicated or noisy, a simple top-down scheme cannot cope with them. It is difficult to extract the features necessary for the analysis all at once. The feedback scheme shown in Fig. 3-1-c has been developed to overcome this difficulty. The results of partial analysis are fed back to the low-level process to guide it in searching for more detailed features. This scheme works effectively when important features which determine the overall structure of input patterns can be extracted rather easily at the initial stage of analysis. Otherwise, the costly operations which deal with the raw image data are apt to be iterated wastefully due to the trial-and-errors between the higher and lower level processes.

(d) Nonpurposive segmentation scheme

The feedback scheme tries to extract the features directly from the flood of data, the raw image, by the top-down control. If the feature extraction can be performed on a well-organized database, the inefficiency of the feedback scheme will be fairly improved. One way to construct such a database is to segment the input image into a set of edges or regions and to organize them into a structured description. It keeps in a symbolic manner rich information extracted from the signals, the raw image. Figure 3-1-d illustrates this idea. The merits of making such descriptions are:

1) It is convenient to manipulate the descriptions by high-level language such as Lisp or FORTRAN. This helps in implementing the higher-level process.
2) The picture processing functions can be executed in high speed without dealing with the raw image.

Once we take this view, the main role of the segmentation is not necessarily to "reduce" the amount of data but to "structure" the data into usable information. Furthermore, the algorithm must be a "nonpurposive" one; it must be applied to a wide range of tasks. Marr's "Primal Sketch" [Marr, 1975] is one way of making such a "nonpurposive" symbolic description of image data. He extracted edge segments in the image by means of various kinds of local filtering and organized them into a set of symbols.

We have developed an image analysis system based on this scheme. Regions are employed to make the description of image data. The description is named "Patchery Data Structure".

III-1-2. Specifications for Segmentation and Symbolic Description

An image of the scene is given as red, green, and blue intensity

arrays. The "nonpurposive" symbolic description of the image is constructed by two processes: the segmentation process and the description process. The segmentation process extracts the structural information of the image by partitioning it into coherent regions using the spectral information. The description of the segmented image must be organized to facilitate flexible retrieval of pictorial information.

We have settled the following specifications on the segmentation.

(a) It deals with color images of 256 x 256 pixels.

(b) It must extract the detailed structures as well as the global structures in the image .

(c) It should apply to a wide range of tasks; i.e., it must be "nonpurposive".

(d) Segmentations based on texture differences are not considered.

(e) The textural areas should be kept from being broken into too many tiny fragments.

(f) The partition must be performed such that we can assume a single region does not span over more than one object in the scene. That is, the partition may go to the state of the "over-segmentation".

As for the description, the following specifications are settled.

(g) The description is to be constructed using regions, boundaries, vertices, etc. as the descriptive elements. The relationships between the regions must be described.

(h) High-speed derivation of the pictorial information from the description must be possible.

(i) The storage size necessary for the description should be small.

(j) The features explicitly included in the description should be limited to the basic ones. Other features are derived from the described ones when they become necessary.

III-2. Region Splitting Using Multihistograms

III-2-1. The Algorithm

We adopted an algorithm which uses multihistograms of one dimension to find the features to be used for region splitting. The basic idea of the algorithm is as follows: The whole image is first partitioned into sub-images each of which is a connected region; then each sub-image is further partitioned if it is possible; and this process iterates. This algorithm has been applied by Tomita et al. [1973] to the segmentation of artificial textural patterns. Ohlander [1975] applied it to the segmentation of color scenes.

Because the algorithm uses the histograms to find the cues for segmentation, it is suitable for the extraction of global structures in input images. On the other hand, it is weak in detecting the detailed structures. Some improvements are needed to realize a segmentation process which satisfies the specifications described in the previous section.

Figure 3-2 shows a schematic diagram of the segmentation algorithm using multihistograms. Because of the recursive nature of the algorithm, a picture stack is used to store the region masks. A region mask represents a connected region (the area without hatching in Fig. 3-2) which is to be examined for segmentation. The segmentation is performed as follows.

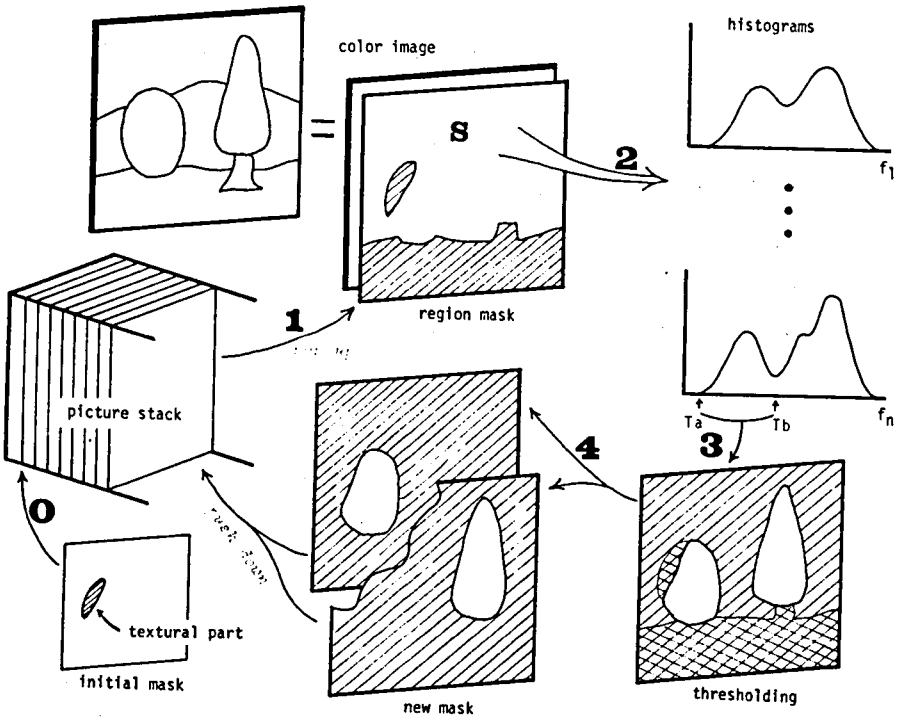


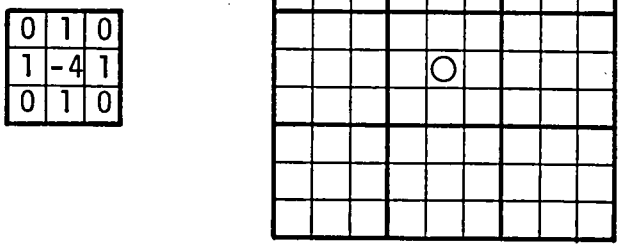
Figure 3-2. Region splitting using multihistograms.

1. The textural areas are first extracted (see III-2-2). A mask corresponding to the un-textural area is placed at the bottom of the stack. --- The arrow 0 in Fig. 3-2.
2. If the stack is empty, the algorithm stops. (Segmentation is finished.)
3. One mask is taken from the top of the stack. Let S denote the region represented by the mask (the area without hatching). --- The arrow 1 in Fig. 3-2.
4. If the region S is small (i.e., the number of the pixels in S is less than a threshold T1), no splitting is tried to S further. S is memorized as a resultant region.
   GO TO 2.
5. Histograms of color features in the region S are computed. --- The arrow 2 in Fig. 3-2.
6. If all histograms are monomodal, the algorithm skips to 9.
7. If any of the histograms show conspicuous peaks, a pair of

cutoff values which separate the peak in the histogram are determined at the position of valleys, and the image of the color feature corresponding to that histogram is thresholded using the cutoff values; thus the region S is partitioned. --- The arrow 3 in Fig. 3-2.

8. Connected regions (with area greater than a threshold T2) are extracted. For each connected region, a region mask is generated, and it is pushed down on the stack. --- The arrow 4 in Fig. 3-2.
   GO TO 2.
9. If the area of the region S is not large (less than a threshold T3), the region S is memorized as a resultant region, and GO TO 2.
10. Extraction of detailed structures in the region S is tried by the window scanning method (see III-2-5). If it succeeds, the region S is partitioned, GO TO 8. Otherwise, the region S is memorized as a resultant region.
    GO TO 2.

The threshold T1 is determined to avoid exhaustive trials for splitting small regions. The threshold T2 is used to reject noisy regions and to prevent meaningless fragmentation. The threshold T3 is determined in connection with the size of the small windows. In the case of segmenting color images with 256 x 256 pixels, the thresholds T1, T2, and T3 are set to 50, 8, and 1536, respectively. Admitting the slight over-segmentation, the quality of the segmentation results is not significantly influenced by the change of the threshold values.



(a) Laplacian operator (b) 9 x 9 window operator

Figure 3-3. Laplacian operator and 9 x 9 window operator.

III-2-2. Pre-extraction of Textural Parts

The textural parts (busy parts) in input images are apt to be divided into a lot of meaningless fragments. In order to prevent this, the textural parts are first extracted from the image and the segmentation process is applied only to the parts without textural property. The extraction of the textural parts is performed by the following steps.

(1) The Laplacian operator (Fig. 3-3-a) is applied to the green image to produce a Laplacian image. Thresholding is executed at a cutoff value T to yield an edge image. The green image is used because it is the most similar to the black-and-white image among the three component images. The cutoff value T is determined from the mode value and the standard deviation of the histogram for the Laplacian image.

$$T = (\text{mode value}) + (\text{standard deviation}) \times 1.4 \qquad (3\text{-}1)$$

(2) Utilizing the 9 x 9 window, shown in Fig. 3-3-b, on the binary picture, if more than 8 of 9 subwindows (3 x 3) have at least one "1", the central pixel is considered to be textural. The textural pixels which form connected regions with large area are determined to be textural parts.

This method detects the areas with scattered edge segments. It is not sensitive to the sharp edges at the boundaries between objects. Figure 3-4 shows the sequence of textural parts extraction. Fig. 3-4-a is an input image, Fig. 3-4-b is the edge image, and Fig. 3-4-c shows the textural parts extracted.

III-2-3. Selection of Cutoff Values Using Histograms

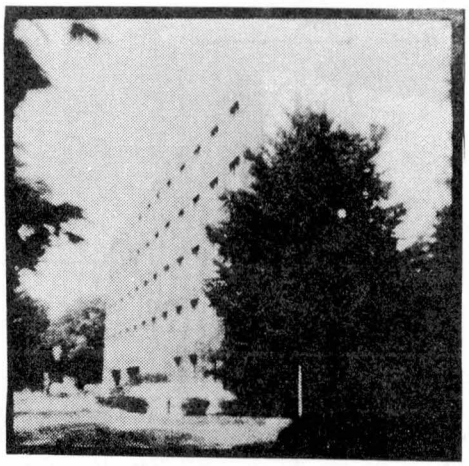
A pair of cutoff values used to partition the image at step (7) of the segmentation algorithm are selected from the positions of valleys in the histograms.

After a smoothing operation all peaks and valleys in the histograms are detected and a score is calculated for each peak (see Fig. 3-5).

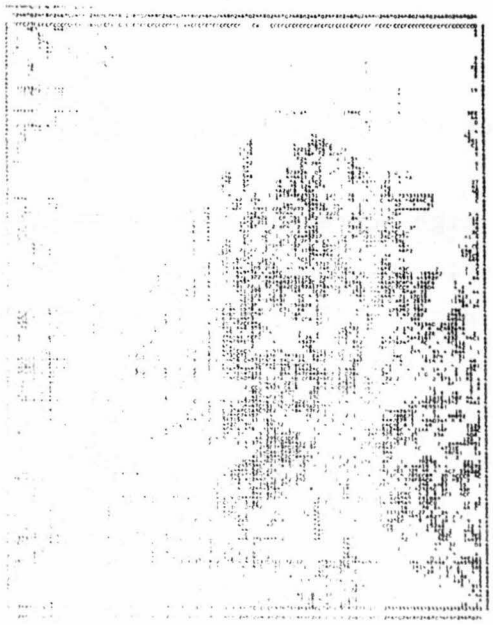
$$\text{score} = ((2P-Va-Vb)/2P) \times ((W-Np/P)/W) , \qquad (3\text{-}2)$$

where, Np is the number of pixels contained in the peak.

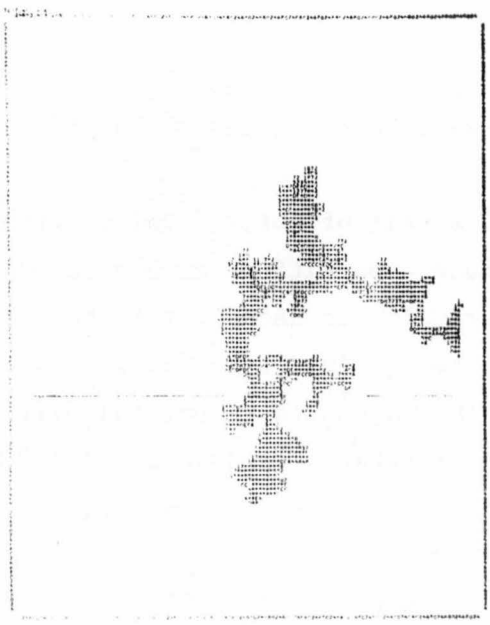
The first term represents the relative depth of the valleys and the second the sharpness of the peak. The deeper valleys and the sharper peak are more desirable. The best three peaks are selected through all the histograms and the positions of their valleys are used as the candidate cutoff values.



(a) input image



(b) edge image



(c) textural parts

Figure 3-4. An example of textural parts extraction.

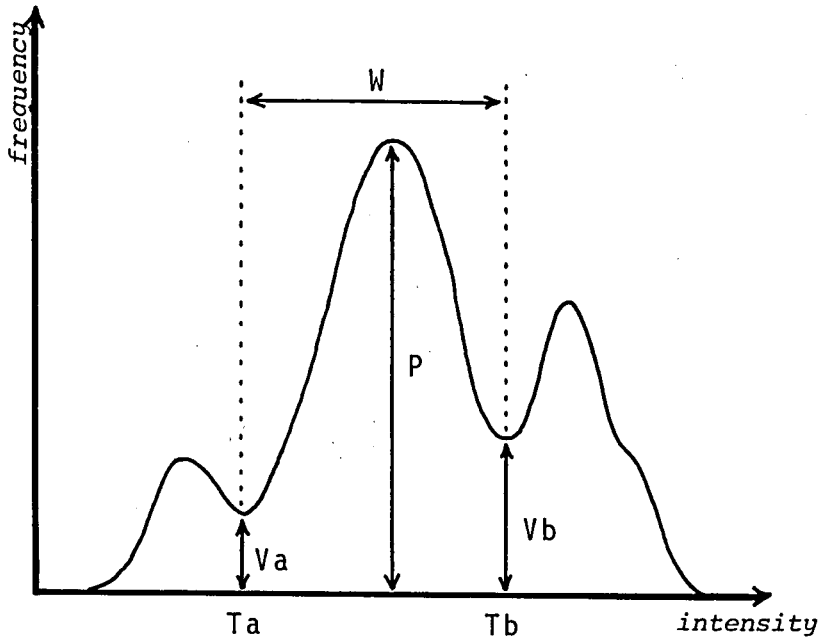


Figure 3-5. Score for a peak in a histogram.

III-2-4. Verification of Cutoff Values by Spatial Evaluation

A pair of cutoff values with a high score does not necessarily produce a partition with good quality; an accidental partition may occur. It is necessary to test the quality of the partitions that will be obtained by using the candidate cutoff values selected in the histograms and to reject bad partitions.

We define "looseness" of a binary mask which represents the regions extracted by the thresholding.

$$\text{looseness} = ((NF-NO)/NO)^2 + ((NO-NS)/NO)^2 , \qquad (3\text{-}3)$$

where, NO is the number of 1's in the binary picture,
NF is the number of 1's after "fusion" operation,
NS is the number of 1's after "shrinking" operation.

Roughly speaking, the looseness is related to the ratio of the number of boundary points at which 1's neighbor with 0's, to the total number of 1's; but it can be calculated faster than the strict ratio. The way to calculate the looseness is illustrated in Fig. 3-6. For each partition, the looseness is calculated for both binary pictures. The partition which has the smallest sum of the looseness values is selected as the best.

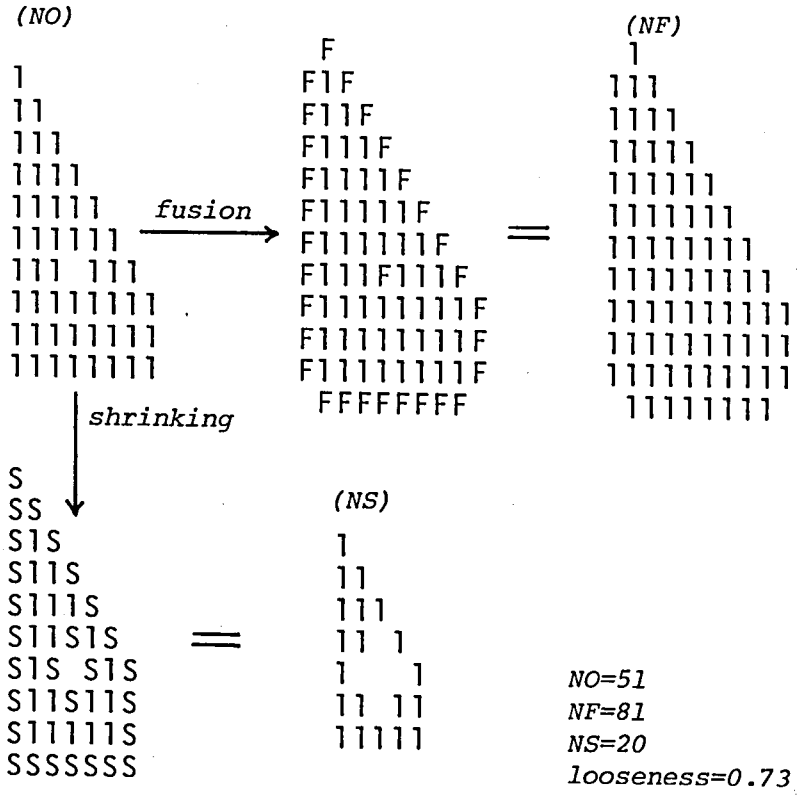


Figure 3-6. Calculation of "looseness" for a binary mask.

NO is the number of 1's in the mask,
NF is the number of 1's after "fusion" operation,
NS is the number of 1's after "shrinking" operation.

III-2-5. Detection and Extraction of Detailed Structures

The segmentation algorithm uses histograms to detect the structures in a region to be examined. In the case of large regions, sometimes no valleys can be detected in any histogram even when the region contains some structures. This is because small valleys in a histogram are veiled by dominant peaks or because many small peaks overlap with each other. To cope with such cases, regions with area greater than a threshold are scanned using a window, and it is tested whether the histograms for each window have valleys. This operation is illustrated in Fig. 3-7. The window size is set to 32 x 32 when the image size is 256 x 256.

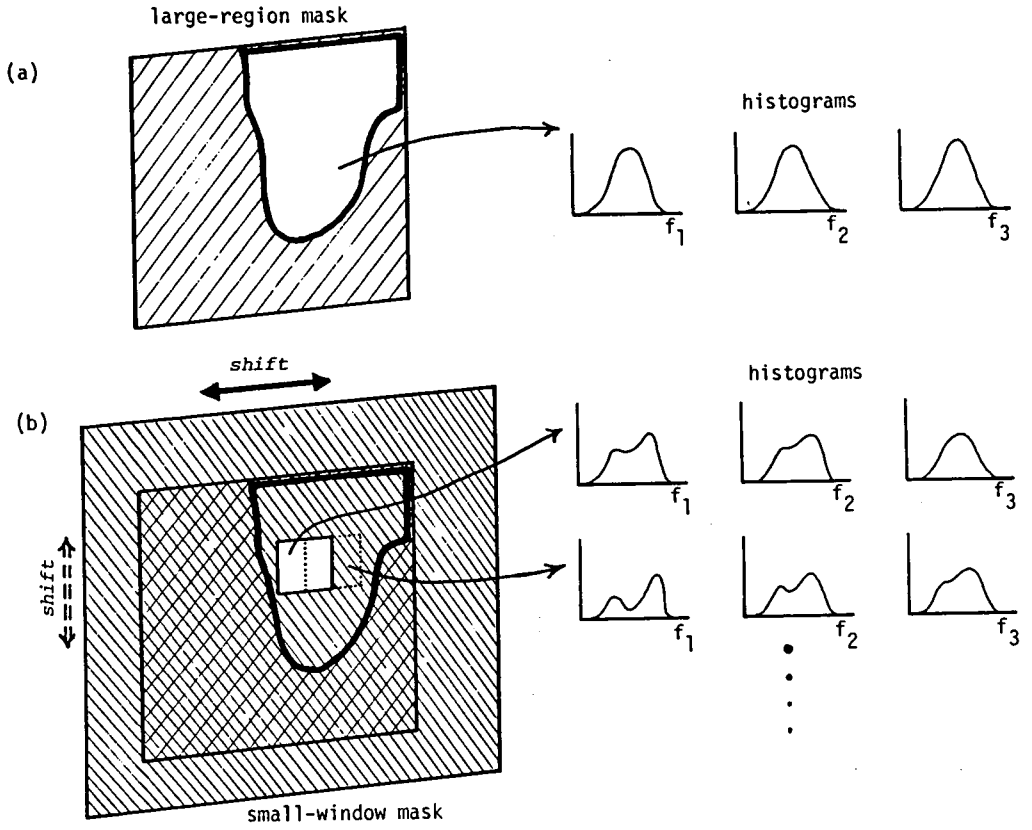


Figure 3-7. Schema of the window-scanning method.

(a) In large regions sometimes no valley can be detected in any histogram even when the region contains some structures.

(b) Large regions are scanned by a small window and the histograms for each window are tested.

This method works well in detecting the detailed structures in large regions, but the partition should be performed with caution. Strictly speaking, the cutoff values selected from the histogram for a particular window are valid only for the local area in the window. If the cutoff values are applied to the whole region, undesirable partitions may happen as illustrated in Fig. 3-8. In order to avoid this, after applying the cutoff values to the whole region, only the extracted segments which intersect the window corresponding to the histogram are picked up and partitioned.

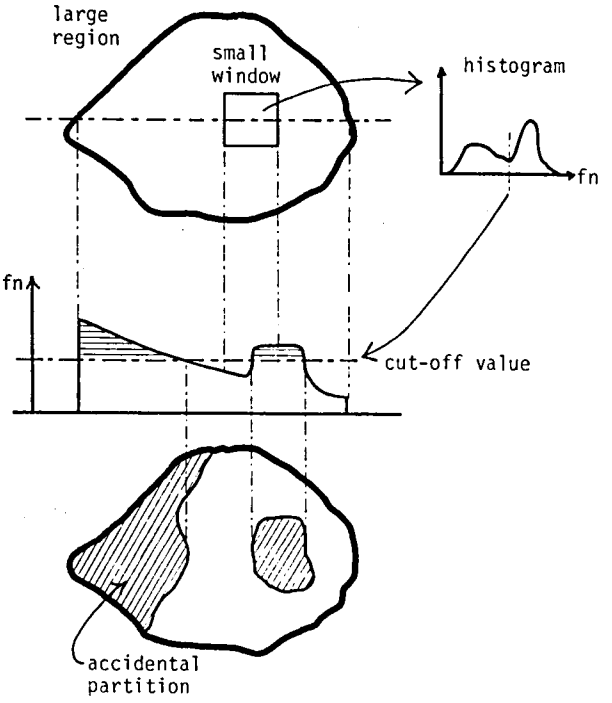


Figure 3-8. Simple application of a threshold obtained by the window-scanning method may cause accidental partitions.

The value of the feature fn smoothly changes in the left part of the region, while it changes abruptly in the right part. If the cutoff values selected on the histogram for the window in the right part is applied to the whole region, an accidental partition may happen in the left part.

III-3. Representation of Segmented Image

III-3-1. Patchery Data Structure

The regions obtained as the result of segmentation are recorded as a two dimensional image array. The role of the symbolic description is to arrange the segmentation results into a well-organized data structure which allows easy derivation of pictorial features related to the properties of and the relations between the regions. The data structure is named "Patchery Data Structure".

Few works have been reported yet on a structured description of image data for the purpose of retrieving the pictorial information. Kunii et al. [1974] reported a system which uses the relational model to represent the pictorial information. The primary aim of their system is to build an image database with an ability to retrieve images by specifying their contents. In our case, the primary aim is to retrieve features of an image based on regions. Relationships among regions must be represented by ordered sets with variable number of elements; for example, a set of boundary segments forms the contour of a region. These relations are conveniently expressed by using pointers. Regions, boundary segments, and vertices are the descriptive elements in the Patchery Data Sturcture, and various kinds of pointers are used to describe relations between the elements.

We define regions, boundary segments, and vertices as illustrated in Fig. 3-9. The connectedness of each region is considered by 4-neighbors. The boundary segments are defined on the mesh placed on the interval of the pixels. Each boundary segment corresponds to the border between two regions and it is represented by chain codes (with 4 directions) from the start vertex to the end vertex. Each vertex is a start or end point of a boundary segment between two regions. A vertex is the corner point at which three or

four regions encounter.

Holes and line segments are also the descriptive elements. A hole is a group of regions which is surrounded by another region. A line segment corresponds to a linear section of a boundary segment. The iterative end-point fits method [Duda and Hart,1973] is used to fit line segments to the chain code of a boundary segment. Figure 3-10 illustrates the line fitting operations.

III-3-2. Primary Features and Secondary Features

The description must support high-speed derivation of various kinds of features when they become necessary in the analysis process. One possible scheme is that every feature needed in the analysis process is calculated beforehand and entered in the description. But this is infeasible for the following two reasons: (1) It is wasteful to calculate the features which may not be used in the analysis process; (2) a large amount of storage is necessary to store all the calculated features.

Features used in the image analysis can be divided into two classes: primary features and secondary features. The calculation of a primary feature deals with the image arrays directly, and, generally, it is time consuming. On the other hand, the secondary features, such as compactness of a region, can be calculated from a set of primary features, and their calculation can be performed in high-speed. Based on such considerations, only the primary features are entered in our description. The secondary features will be calculated from the primary features of the descriptive elements and their relationships when needed in the analysis process.

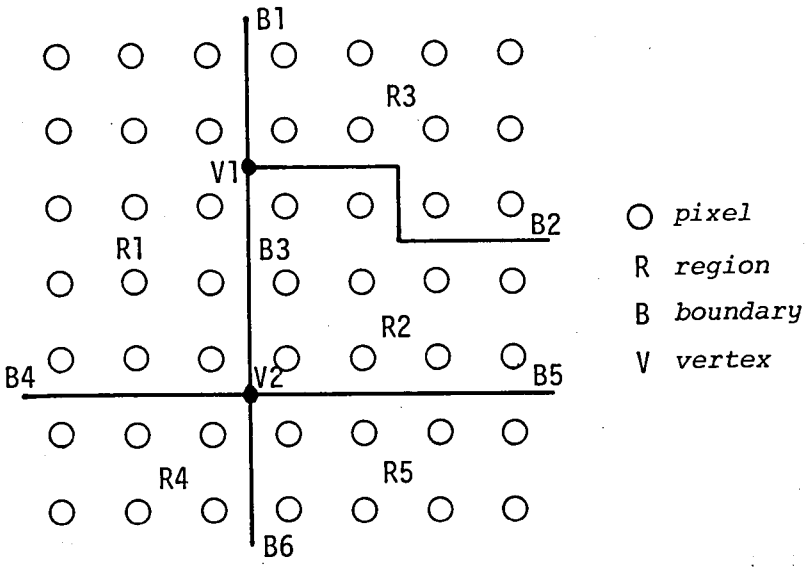


Figure 3-9. Regions, boundary segments, and vertices.

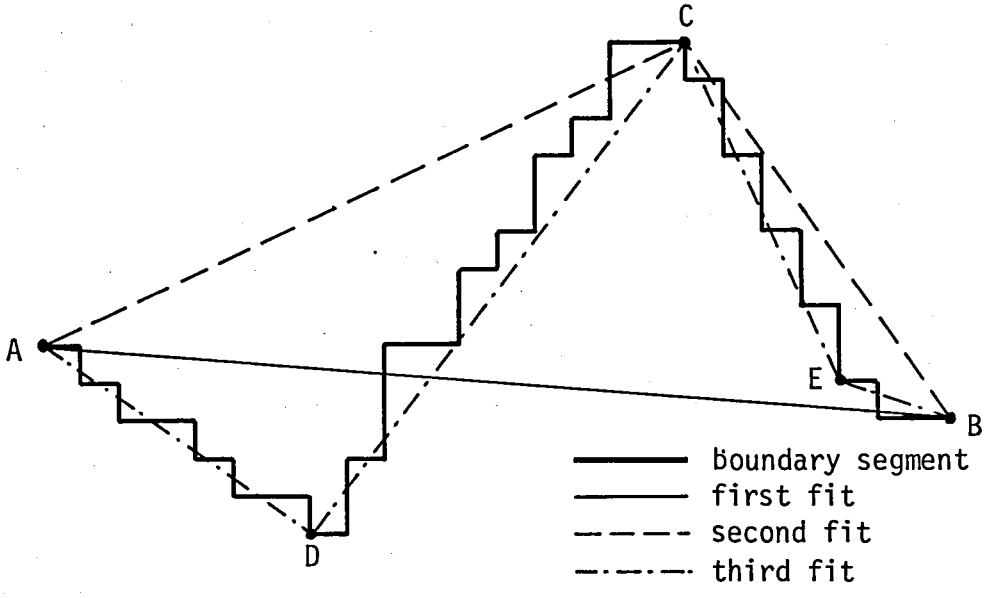


Figure 3-10. The iterative end-point fits method.

A and B are the end-points (vertices) of the boundary segment. The line AB is initially fit to the boundary. The distances from each point on the boundary to this line are computed, and if all the distances are less than a threshold the process is finished. If not, the point C, furthest from the line AB, is found and the line AB is broken into two new lines AC and CB. This process iterates.

III-3-3. Description of Properties

The properties of regions, boundary segments, vertices, holes, and line segments are described in each descriptive element. Table 3-1 shows the features used for the description as the properties of descriptive elements. For a region, the following features are described (Fig. 3-11).

(1) Area --- The number of pixels included in the region.
(2) Mean intensities of R, G, and B data.
(3) Degree of texture --- The mean value of the operator in Fig. 3-3-b.
(4) Contour lengh --- The total length of the boundaries which surround the region.

Table 3-1. Primary features for each descriptive element.

descriptive element	primary features
region	area; mean intensities of R, G, and B; degree of texture; contour length; position of the mass center; number of holes; scatter matrix of pixel positions; minimum bounding rectangle (MBR).
boundary segment	chain codes; length; contrast.
vertex	position; number of boundary segments.
hole	contour length.
line segment	distance from origin (ρ); orientation (θ); length; positions of end points.

(5) Position of the mass center --- The position vector M=(MX, MY) of mass center of a region is computed as follows:

$$M = (1/N) \sum_{i=1}^{N} Pi \ , \qquad (3\text{-}4)$$

where, N is the number of pixels in the region,
Pi is the position vector of i-th pixel.

(6) Minimum bounding rectangle (MBR) --- The MBR is defined as a set of four figures XMIN, XMAX, YMIN, and YMAX. They are the maximum and minimum values of the X and Y coordinates of the pixel positions in the region.

(7) Scatter matrix --- The scatter matrix represents the elliptical area which approximates the shape of the region. It is more useful than the minimum bounding rectangle. The scatter matrix C of a region can be calculated as follows:

$$C = (1/N) \sum_{i=1}^{N} (Pi-M)(Pi-M)^{t} \ , \qquad (3\text{-}5)$$

where, N is the number of pixels in the region,
Pi denotes the position vector of i-th pixel,
M denotes the mass center of the region.

Figure 3-12 illustrates the features described for boundary segments. The contrast is the mean difference of R, G, and B values across the boundary segment.

Figure 3-13 shows the features for line segments. The polar coordinates system (ρ-θ system) is used to represent a line segment. The ρ-θ system is convenient to aggregate co-linear line segments.

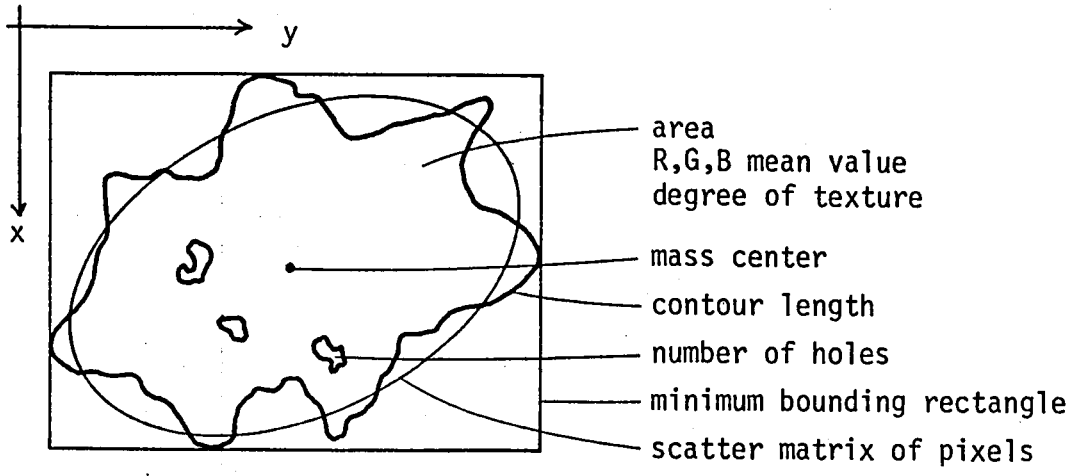


Figure 3-11. Description of a region.

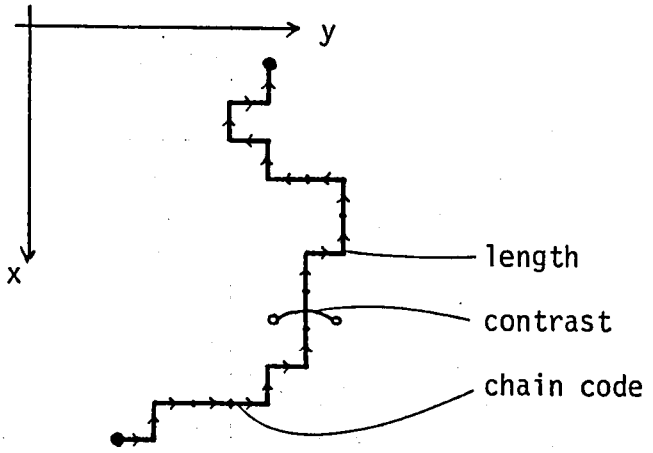


Figure 3-12. Description of a boundary segment.

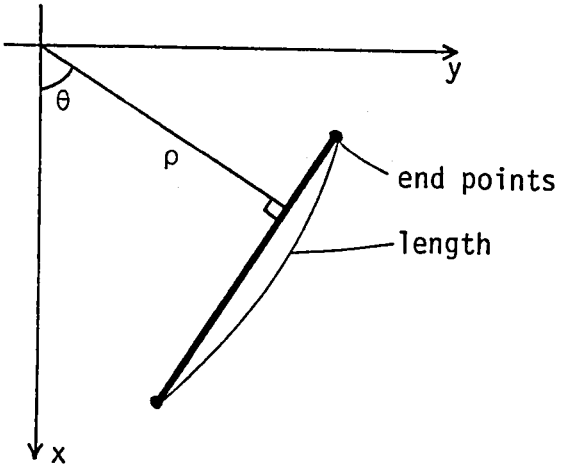


Figure 3-13. Description of a line segment.

III-3-4. Description of Relations

III-3-4-1. Topological Relations

The topological relations among regions, boundary segments, vertices, holes and line segments are expressed by the pointers between each descriptive element. Figure 3-14 illustrates the pointers used in the Patchery Data Structure. The meaning of the pointers will be clear from the figure. The boundary segments which form the contour of a region or a hole are ordered counter-clockwise. According to the considerations in section III-3-2, only the "primary relations" are explicitly represented. Other relations can be derived from the primary ones when they become necessary. For example, the set of regions touching a certain region can be obtained as follows: (1) The set of boundary segments surrounding the region is first derived; (2) the regions on the left side or the right side of each boundary segment are gathered as the objective set.

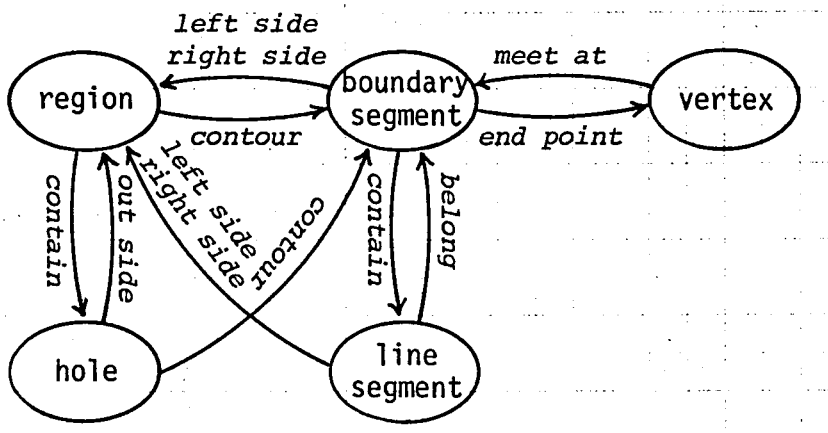


Figure 3-14. Description of topological relations.

III-3-4-2. Color Relations

It is useful to describe the relationships among the regions which have similar colors. The basic operation of the segmentation algorithm using multihistograms, which we employed in our system, is to extract regions with a similar spectral property. A segmentation tree which records the history of segmentation can be used to describe the color relations among the regions. Figure 3-15 is an example of the segmentation tree. Each node in the tree corresponds to a region. The root-node corresponds to the whole image and the leaf-nodes correspond to the elementary regions obtained as the final result of the segmentaion. Every node except the leaf-node has its child-nodes which correspond to the regions obtained by splitting the region corresponding to that node. There are two kinds of child-node: sons and daughters. The sons correspond to the regions which are extracted by the cutoff operation from the parent region. The daughters correspond to the remaining regions. The cutoff values used

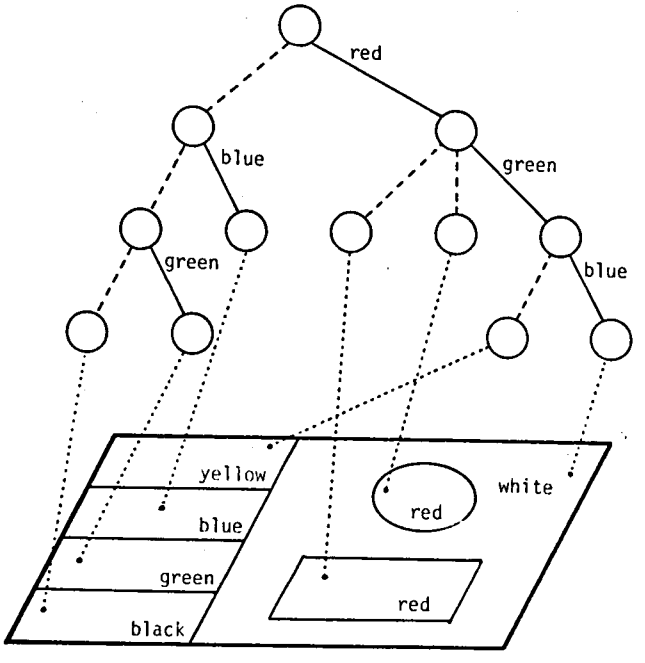


Figure 3-15. Description of color relations.

to separate the region is recorded in the corresponding node.

The distance between two nodes in the segmentation tree is defined as the path length between them. The regions which are near in the segmentation tree have similar colors. But it can not be said that distant nodes have very different colors. Furthermore, the color relation represented by the entire segmentation tree depends on the color features which are used for the segmentation process; i.e., the segmentation tree created by using R, G, and B is different from that created by using Y, I, and Q. In spite of such difficulties, we think that the segmentation tree will be a useful tool for retrieving the regions based on the similarity of color.

III-4. Manipulation of the Patchery Data Structure

III-4-1. Merging of Regions

In region-oriented image analyses, it is often necessary to merge several regions into one. In such a case, the features in Table 3-1 of a new region should be calculated from the ones of old regions without dealing with the image arrays. In our system, it can be performed as follows.

(1) Mean intensities of R, G, and B data, degree of texture, position of mass center --- The feature f of the new region can be calculated by the weighted average.

$$f = \left(\sum_{i=1}^{N} fi \cdot Si\right) \Big/ \sum_{i=1}^{N} Si \,, \qquad (3\text{-}6)$$

where, Si and fi denote, respectively, the area and the feature of the i-th old region.

(2) Minimum bounding rectangle (MBR) --- The MBR (XMIN, XMAX, YMIN, YMAX) of the new region can be obtained as the MBR of the MBRs of old regions.

$$XMIN = \min \{XMINi\},\ XMAX = \max \{XMAXi\},$$
$$YMIN = \min \{YMINi\},\ YMAX = \max \{YMAXi\}, \qquad (3\text{-}7)$$

where, (XMINi, XMAXi, YMINi, YMAXi) denotes the MBR of the i-th old region.

(3) Scatter matrix --- Let Ci, Mi, and Si (i=1...n) denote individually the scatter matrix, the position vector of the mass center, and the area of the i-th old region. Let M denote the position vector of the mass center of the new region, which is calculated by Eq. 3-6. Then the new scatter matrix C can be obtained as follows:

$$C = \left(\sum_{i=1}^{N} Si \cdot (Ci + (Mi-M)(Mi-M)^t) \right) / \sum_{i=1}^{N} Si \ . \qquad (3\text{-}8)$$

III-4-2. Derivation of Various Features

Various kinds of features are used in region analysis systems to represent the properties of and the relations between the regions. Table 3-2 shows the typical features which we used. We will show how we can derive those secondary features easily from the primary ones entered in the Patchery Data Structure. The computational time necessary to derive them from the data structure is shorter than computing them directly on the image arrays.

(1) Normalized colors (r, g, and b), intensity, hue, and saturation of a region --- They can be calculated from the mean intensities of R, G, and B of the region.

$$r = R/(R+G+B),\ g = G/(R+G+B),\ b = B/(R+G+B),$$
$$\text{intensity} = (R+G+B)/3,$$
$$\text{saturation} = 1-3\cdot\min(r,g,b)\ ,$$
$$\text{hue} = \arctan2(\sqrt{3}(G-B),(2R-G-B))\ . \qquad (3\text{-}9)$$

(2) Compactness of a region --- It can be calculated from the area and the contour length of a region.

$$\text{compactness} = 4\pi\cdot\text{area}/(\text{contour length})^2\ . \qquad (3\text{-}10)$$

(3) Crude shapes of a region such as vertically-long or horizontally-long --- They can be defined based on the VH-ratio which is computed using the scatter matrix.

$$\text{VH-ratio} = \log(C11/C22)\ , \qquad (3\text{-}11)$$

where, C11 and C22 are the diagonal components of the scatter matrix C.

(4) Effective width and effective MBR of a region --- The effective width and the effective MBR are both computed from the scatter matrix. When the scatter matrix of a region is $\begin{pmatrix} C11 & C12 \\ C21 & C22 \end{pmatrix}$, the effective MBR (XMIN, XMAX, YMIN, YMAX) is a rectangle which has a scatter matrix $\begin{pmatrix} C11 & 0 \\ 0 & C22 \end{pmatrix}$. And the effective width (XW, YW) is the width of the effective MBR.

Table 3-2. Secondary features which can be derived from the primary ones.

features of a region	normalized colors; intensity; saturation; hue; compactness; VH-ratio (crude shape); effective width; effective MBR.
features between two regions	contrast of the border; orientation of the border; linearity of the border; T-ratio (touching ratio); O-ratio (overlapping ratio); placement relations (above, below, etc.).

$$XW = 2\sqrt{3 \cdot C11}, \qquad YW = 2\sqrt{3 \cdot C22},$$
$$XMIN = XC - XW/2, \qquad XMAX = XC + XW/2,$$
$$YMIN = YC - YW/2, \qquad YMAX = YC + YW/2, \qquad (3\text{-}12)$$

where, (XC, YC) is the mass center of the region.

(5) The contrast at the border of two regions --- It can be computed by averaging the contrast of the boundary segments included in the intersection of the contours of the two regions.

(6) The degree that a region touches another region (T-ratio) --- The T-ratio of region-1 to region-2 is calculated from the length of the border between the two regions and the contour length of the region-1.

$$\text{T-ratio} = (\text{border length})/(\text{contour length of region-1}). \qquad (3\text{-}13)$$

(7) The degree of a region being surrounded by another region (O-ratio) --- The O-ratio of region-1 to region-2 is computed as the ratio of the overlapped area of the MBRs of the two regions to the whole area of the MBR of the region-1.

$$\text{O-ratio} = (\text{overlapped area})/(\text{whole area}). \qquad (3\text{-}14)$$

(8) The linearity and the orientation of the border between two regions --- These can be derived from the set of line segments included in the boundary of the two regions.

(9) Positional relationships between two regions such as above, below, left, or right --- These can be derived by using the mass centers and the MBRs of the two regions. Let (XC1, YC1) and (XC2, YC2) be the mass centers of region-1 and region-2, respectively. Let (XMIN1, XMAX1, YMIN1, YMAX1) and (XMIN2, XMAX2, YMIN2, YMAX2) be the effective MBRs of the two regions illustrated in Fig. 3-16. Then the relation "region-1 is above region-2" is defined as follows:

$$XMIN1<(XMIN2+XC2)/2 \wedge XMAX1<XC2 \wedge YMIN2<YC1<YMAX2. \qquad (3\text{-}15)$$

Actually, we defined "above" as a fuzzy predicate; the truth value is affected by the degree that YC1 is out of the range [YMIN2, YMAX2]. The other relations are defined in the same way.

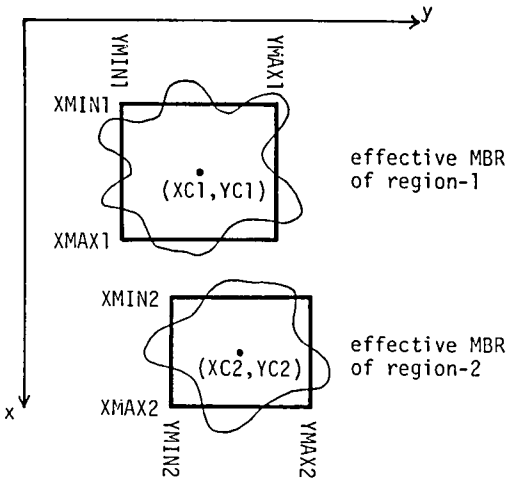


Figure 3-16. Two effective MBRs to define "above".

III-4-3. Retrieving the Regions

When the analysis is performed by means of a top-down strategy, it is necessary to retrieve regions out of the image data by specifying the properties they should have; for example, "fetch all the regions that are vertically long and have a yellow one on their left". The computation for such a retrieval is usually very large. Only a well-structured symbolic description can allow such a function to be of practical use. In our system, the following three primitive functions are for this purpose (cf. Table 4-3 in p.126).

ALL-FETCH (<to-set>, <from-set>, <fuzzy-predicate>) ,
THERE-IS (<region>, <from-set>, <fuzzy-predicate>) ,
T-FETCH (<to-set>, <region>) .

ALL-FETCH selects from a set of regions, which is specified by <from-set>, all the regions that satisfy the condition described by

<fuzzy-predicate>, and assigns them to <to-set>. THERE-IS, existential fetch, selects only one region first found. Nested use of these functions realizes arbitrarily complicated retrievals. T-FETCH selects all the regions that are touching <region>. This function, of course, can be realized by using ALL-FETCH, but T-FETCH is faster because it utilizes the relational pointers in the Patchery Data Structure. Figure 3-17 illustrates a function defined by using ALL-FETCH and THERE-IS to perform the retrieval of the above example, "fetch all the regions that are ...".

```
(ALL-FETCH  *TO-SET  *REGIONS
   (AND  (VERTICALLY-LONG  *TO-SET)
         (THERE-IS  *Y-RGN  *REGIONS
            (AND  (YELLOW  *Y-RGN)
                  (LEFT-OF  *Y-RGN  *TO-SET)))))
```

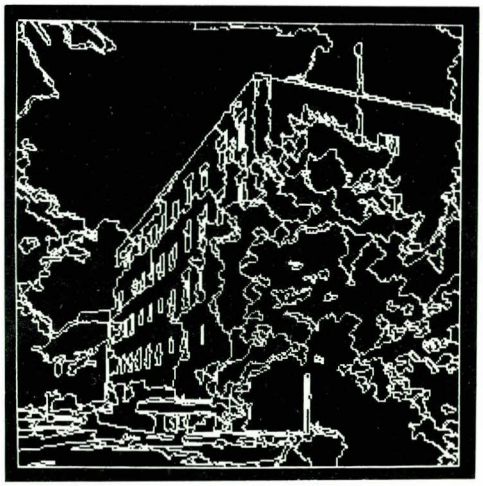
Figure 3-17. The function to "fetch all regions that are vertically long and have a yellow one on their left".

III-5. The Results

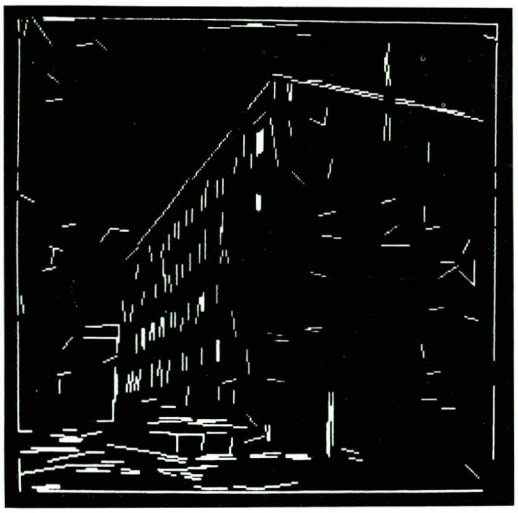
The segmentation process and the structuring process described in this chapter have been applied to a lot of color scenes, and produced successful results. This section presents the results for two scenes for illustration. Other results can be seen in the appendix.

Figure 3-18-a is an outdoor scene. The input picture is digitized with 256 x 256 size and 5-bit density resolution for each

(a) digitized color image



(b) result of segmentation

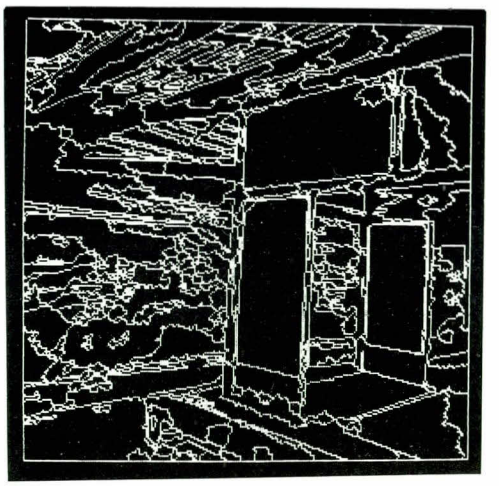


(c) straight line segments

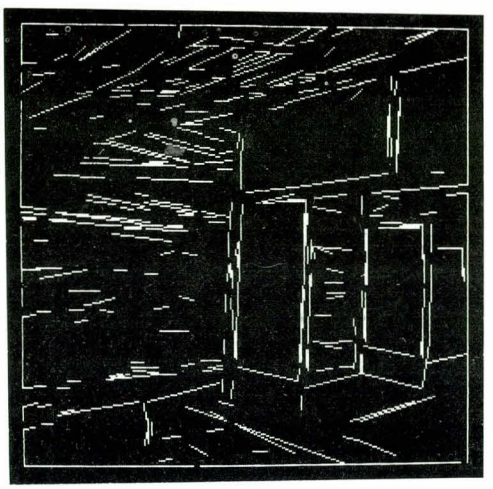
(d) reconstructed color image

Figure 3-18. Result of preliminary segmentation: Example 1.

(a) digitized color image



(b) result of segmentation



(c) straight line segments

(d) reconstructed color image

Figure 3-19. Result of preliminary segmentation: Example 2.

R, G, and B data. Figure 3-18-b shows the result of segmentation. R, G, and B are used as the color features for histogramming in the segmentation algorithm. The detailed structures, such as the windows of the building, are extracted successfully. The fragmentation of the textural part like trees is suppressed. Conceivably, the quality of this segmentation is sufficient as the "nonpurposive" segmentation. Figure 3-18-c shows the straight line segments fit on the boundary segments. Figure 3-18-d is a color image reconstructed from the structured description finally obtained. Mean intensity values of R, G, and B are assigned to each region. In this example the number of regions is 339, the number of boundary segments is 914, the number of vertices is 612, the number of holes is 37, and the number of line segments is 268. The storage necessary for this data structure is about 90 kilo-bytes.

Figure 3-19 shows the result of another color scene. In this case, the input picture is digitized with 256 x 256 size and 6-bit density resolution. Three color features (R+G+B)/3, (R-B), and (2G-R-B)/2 are used in the segmentation process. The numbers of regions, boundary segments, vertices, holes, and line segments are 391, 1121, 742, 11, and 368, respectively.

III-6. Conclusion

In this chapter, a system which makes a structured description of a color image based on regions is described. A powerful segmentation process has been developed by improving the segmentation algorithm which uses multihistograms to obtain the cues for splitting

a region. The segmentation results are organized into a structured symbolic data network. It contains rich information of the input image and supplies the analysis process with the pictorial information in a tractable form.

The features of the system are as follows.
As for the segmentation process:

(1) It can deal with color images.
(2) The global structures in the input image are effectively extracted by using the histograms as the cues for segmentation.
(3) The detailed structures can be extracted by the window-scanning method.
(4) The meaningless fragmentation of the textural area in the input scene is avoided by the pre-extraction of that area.
(5) It works well for various kinds of images.

As for the structured description:

(6) Because it is region-based, the properties of surfaces such as colors or textural features can be described in a natural way.
(7) The features entered in the description are limited to the primary ones. This realizes a compact description.
(8) Various kinds of features can be derived in high speed from the description.
(9) The region fetch functions which retrieve regions satisfying certain properties from the input image can be realized on the description for practial use.
(10) The merging operation of the regions can be performed on the description without referring to the image arrays.

There are a few unsolved problems. First, our description cannot support the splitting operation of regions. This problem, however, can be almost avoided by over-segmenting the image in the segmentation process. Secondly, the segmentation process takes a lot of time. This point will be fairly improved by employing the special purpose hardwares which can perform high-speed parallel computation for low-level operations such as histogramming and thresholding.

Chapter IV

A BOTTOM-UP AND TOP-DOWN REGION ANALYZER

IV-1. The Problems

The region analysis technique is often employed in analysis systems for natural scenes. Regions are better than edges in dealing with the properties of the surfaces and in obtaining the global structures in the scenes. A region analyzer tries to segment an input image into meaningful regions and assigns object labels to each of them. The term "meaningful" means that each region corresponds to a surface of the objects in the input scene. The homogeneity of color or texture is used as the criteria to divide the image into regions. A meaningful segmentation, however, is hardly achieved by using only such image properties. Various kinds of additional constraints have to be employed to obtain the meaningful regions. There are several attempts for this.

Brice et al. [1970] segmented block-world scenes by the region growing technique. They employed the "phagocyte" heuristic in addition to the "weakness" heuristic which evaluates the similarity of brightness. Even if the boundary between two regions is weak, they are joined only if the resulting boundary does not grow too fast. Thus, the phagocyte heuristic constrains the region growing process to obtain well-shaped regions. It worked well for the block scenes.

For more complex scenes such as outdoor scenes, it is necessary

to introduce more task-specific constraints into the segmentation process. Yakimovsky et al. [1973] incorporated the interpretation process with the segmentation process, and analyzed road scenes which include sky, roads, grass strips, trees, and cars. The input image is preliminarily divided into small fragments based on the similarity of color. Every fragment is interpreted by using the properties of and the relations between the fragments. The fragments which are assigned an identical interpretation are merged as a meaningful region. Their scheme utilizes the restrictions on the kinds of objects in the task world as the constraints to guide the segmentation. Tenenbaum et al. [1976] analyzed a room scene by a method which they call "Interpretation-Guided Segmentation". These two methods are usually called "semantic region analysis".

The semantic region analysis systems have succeeded in analyzing road scenes and simple room scenes. But, they cannot deal with the scenes which contain objects for whose recognition a certain structure has to be identified. For example, it is difficult to deal with a set of regions, such as windows of a building, which are located separately, but which are to be identified as a whole according to some placement rules. The evaluation of shapes comprising several regions is also hard. The reason for this difficulty can be understood as follows. The semantic region analysis has been developed as an improvement of the region growing technique. Thus, it still relies entirely on the bottom-up control scheme of the region growing, and the semantic information is used in the same way as the local pictorial information. In consequence, even though the segmentation is evaluated by a certain score, the usable features which evaluate the semantic constraints are limited to local ones: e.g. color, orientation of a boundary segment, crude shape of a boundary segment, etc. Such a problem stems from the fact that the semantic region analyses so far developed are based on the bottom-up control scheme.

In order to resolve the problem, a model-driven top-down

approach seems essential. It is well known that, when the structure of the input scene is known, the top-down control scheme provides an efficient and reliable analysis. Under the top-down control, it becomes possible to deal with a set of regions located separately in the image such as windows of a building. But when the structure is not known, the top-down scheme is powerless. In contrast, the bottom-up control is usually not so efficient, but it is more robust; it works well even when the structure of the input image is not known. We think that we can build a powerful analysis mechanism by combining the merits of the two complementary control schemes in the following way: (1) The bottom-up control scheme extracts information about the crude structure of the input scene. (2) Based on the information provided by the bottom-up analysis, the top-down control scheme performs an efficient analysis by focussing its attention.

The task world selected is campus scenes of Kyoto University; the scenes include sky, trees, buildings, roads, windows of buildings, and cars on roads.

IV-2. A Bottom-up and Top-down Region Analyzer

IV-2-1. Patches and Regions

Figure 4-1 shows the outline of our region analyzer. An input color image is first partitioned into a set of coherent regions according to the color information. The partition is performed to the state of over-segmentation, that is, we can assume a single region does not span over more than one object, but one object might be divided into multiple regions. The segmented image is organized into

a structured data network which is called Patchery Data Structure. Regions, boundary segments, vertices, holes, and (straight) line segments are used as the descriptive elements. This process is called the preliminary segmentation.

In this chapter, the regions obtained in the preliminary segmentation are denoted by the term "patches". The term "regions" is used only for the regions which are obtained by merging the patches according to a certain criterion. The patches are the atomic elements used to make a scene description in our system.

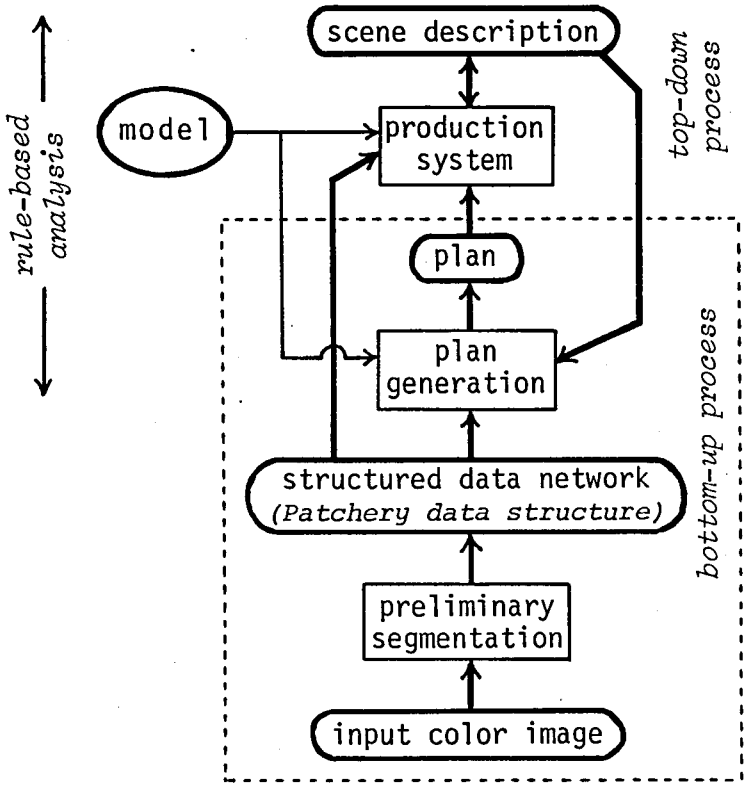


Figure 4-1. Outline of the region analyzer.

IV-2-2. Bottom-up Control and Top-down Control

Deciding where to locate the interface between the bottom-up and top-down processes is an important problem. In Fig. 4-1, the part enclosed by dotted lines is performed under bottom-up control in our system. A "plan" is generated in the bottom-up process as a representation of the crude structure of the input scene. The plan is a set of object labels and their degree of correctness assigned to each of the large patches in the preliminarily segmented image. The plan provides the top-down process with the clues concerning what knowledge can be applied to what part of the scene.

The top-down process fixes the interpretation for the large patches referring to the plan generated in the bottom-up process. Also, it analyzes the detailed structures of the scene by interpreting small patches in the context of the large patches which have been already interpreted. When the top-down process makes a significant decision (such as the position of the scene horizon) which might have an effect on the interpretation of the whole scene, the decision is fed back to the bottom-up process and the plan is re-evaluated. In this way the bottom-up process and the top-down process work cooperatively to achieve the semantic description of the scene.

IV-2-3. Rule-based Analysis

The knowledge of the task world is represented by two sets of rules in our region analyzer: one is used in the bottom-up process and the other in the top-down process. Figure 4-2 illustrates the control mechanism of the region analyzer. Because the knowledge is represented as a collection of modular rules, it is easy to add or modify the knowledge in our system. This is a useful feature in

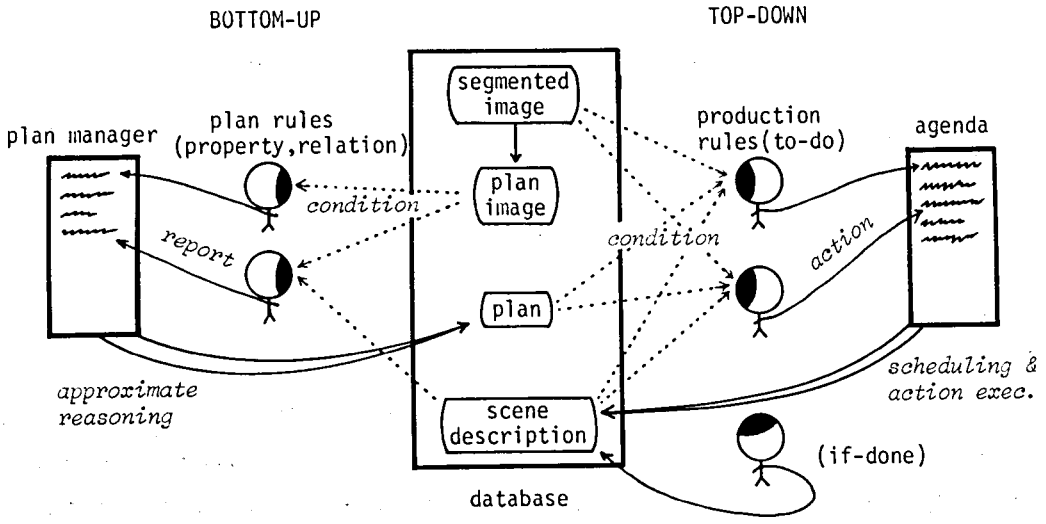


Figure 4-2. Control mechanism of the rule-based analysis.

organizing an analysis system for complex scenes, such as outdoor scenes, which include various kinds of objects.

Each rule for the bottom-up process has a fuzzy predicate which describes properties of or relations between objects. It also has a weight which indicates the uncertainty of the knowledge it relies on. The plan manager controls evaluation of the rules and performs an approximate reasoning based on the fuzzy truth-values.

The top-down process is organized as a production system. Each rule is a pair of a condition and an action. The condition is a fuzzy predicate which examines the situation of the database. The action includes operations to make the scene description. The agenda manages the activation of production rules and schedules the executable actions.

IV-2-4. Plan Generation by Bottom-up Analysis

IV-2-4-1. Plan Image

In order to generate the plan, patches with large area are first selected from the preliminarily segmented image. It is reasonable to assume that most of them correspond to large parts of objects in the scene and that they can be extracted from the image data rather stably in the segmentation process. We call those patches keypatches. It should be possible to grasp the rough structure of the scene by assigning the object labels to the keypatches. The labels have multiple values, and a score indicating the degree of correctness is assosiated to each label value; for example,

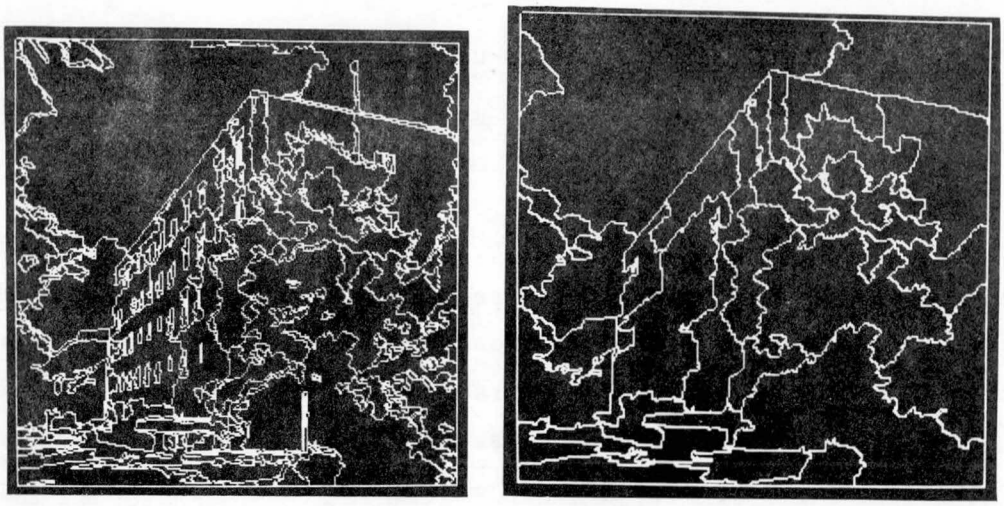
label-of-a-keypatch = [(sky=0.5)(building=0.2)(tree=0.2)(road=0.1)].

The score is computed by evaluating the properties of and the relations between the keypatches.

Small patches in the preliminarily segmented image may disturb the evaluation of the relations between the keypatches. Thus, we tentatively merge all the small patches to one of the keypatches. We call the resultant image a "plan image". The "plan" is the labeled plan image; that is, each region in the plan image are assigned with labels.

Figure 4-3-b shows the plan image generated from the segmented image in Fig. 4-3-a. It must be noted that the plan image is represented symbolically on the Patchery Data Structure, and the image (2-D array) in Fig. 4-3-b is generated only for the purpose of display.

No semantic information is used in the merge operation. When a small patch touches more than two keypatches, a score is computed for each keypatch based on the similarity of color and the compactness of the region which would be obtained if the small patch and the keypatch are merged. The compactness criterion guides the merging



(a) result of preliminary segmentation

(b) plan image generated from (a)

Figure 4-3. Generation of plan image.

operation to obtain regions with smooth boundaries: In order to evaluate the global relations between regions, the smooth boundaries are rather convenient. The keypatch which obtaines the highest score is selected. The merge operation is, of course, performed by manipulating the Patchery Data Structure.

IV-2-4-2. Rules and Plan Manager

Each rule used in the plan generation process has a fuzzy predicate which describes a property of a certain object or a relation between objects. When it is applied to a region to check whether the region can be labeled as the object, it examines pictorial features of the region and produces a fuzzy truth-value which indicates the degree of satisfaction of the property. The

predicate can also examine the scene description which is constructed in the top-down process. In this way, the information extracted in the top-down process, such as the position of the scene horizon, can be reflected in the plan generation process.

In order to evaluate the plan, the plan manager activates every rule to examine every region in the plan image. For every combination of rules and regions, the plan manager receives a fuzzy truth-value. Based on the set of fuzzy truth-values, the plan manager assigns object labels to each of the regions and computes their degree of correctness.

An approximate reasoning scheme is employed for evaluation of the plan to deal with the uncertainty existing in both of the knowledge and the pictorial features. Detailed description of this scheme is included in section IV-3.

IV-2-5. Top-down Analysis of Patches

IV-2-5-1. A Production System Architecture

The top-down analysis process is constructed by using a production system architecture. A production system consists of a set of production rules, a database, and a control structure. A production rule is the unit of knowledge representation, and the database records the facts about the input image. Each production rule is a pair of a condition and an action, and is "watching" the database. Whenever the predicate in the condition part is satisfied, the system evaluates the action part and modifies the database.

As illustrated in Fig. 4-2, the database stores
(1) the preliminarily segmented image and the plan image represented on the Patchery Data Structure,

(2) the plan, and
(3) the scene description so far obtained.

There are two types of production rules in our system: to-do rules and if-done rules. They correspond to the consequent and antecedent theorems of PLANNER [Hewitt, 1968], respectively. A to-do rule performs basic operations in the region growing process. It examines each patch which has not been interpreted yet in the segmented image by the fuzzy predicate in its condition part and determines whether the associated action can be executed for it. The executable actions are added into the agenda with a score indicating their priority, whose computation will be explained soon. The agenda controls the production system. It manages actions which are executable in a context of analysis through their attached scores. The action with the highest score is executed, and, as the result, the agenda is updated.

An if-done rule is a demon. It is triggered by the execution of a certain action of to-do rules.

In the production system architecture, interactions of the to-do rules and if-done rules as modifying the database embody a heterarchical control structure. This enables the top-down analysis process to have a flavor of data-driven control so that the order of analysis is determined according to the reliability of the interpretation of each part of the image, which is given as the plan generated by the bottom-up analysis.

IV-2-5-2. Structure of Scene Description

How to describe the analysis result is essential, especially to realize a top-down control in the analysis, because the system has to grasp the present context of the analysis exactly in order to control

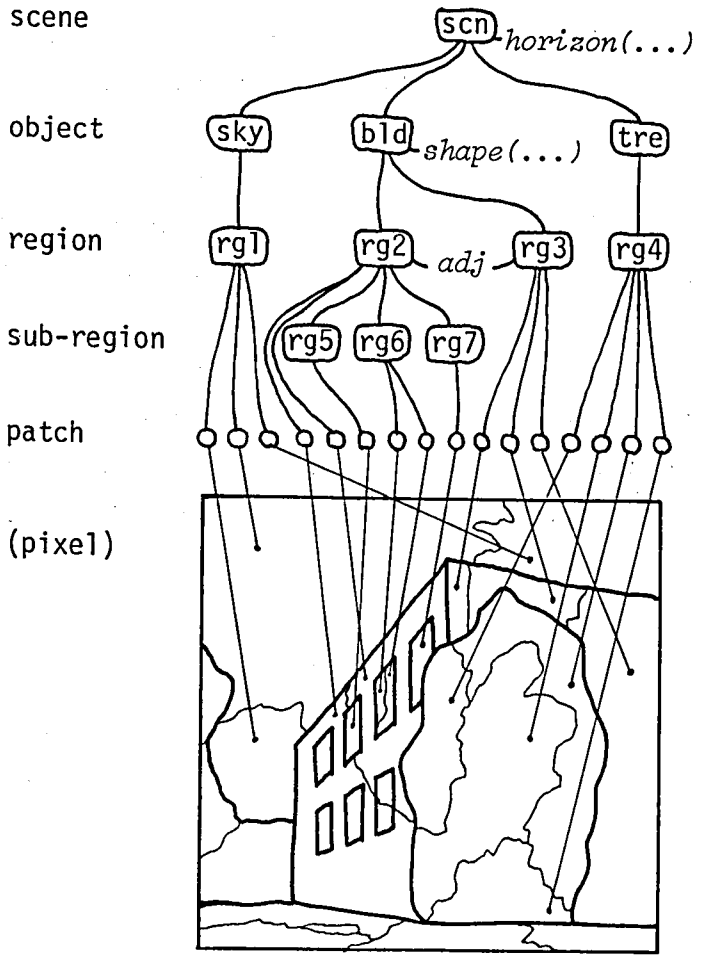


Figure 4-4. Structure of the scene description.

it according to the status of the results so far obtained. Figure 4-4 illustrates the structure of the scene description which is built as the result of the top-down analysis in our system. Scene, object, region, sub-region, patch, and pixel are the important concepts which constitute the structure of the description. The patches are the ones obtained as the output of the preliminary segmentation. They are the atomic elements in the rule-based analysis. The scene represents the whole image being analyzed. The objects stand for the objects extracted in the scene. The regions represent the main parts of objects such as the walls of buildings, and they are obtained by merging the patches which are given the same interpretation. The sub-regions are much the same as the regions, but they correspond to the subjective parts of objects such as the windows of buildings. The difference between (sub-)regions and patches is that the former are the entities given consistent semantic interpretation, whereas the latter are the entities having consistent pictorial properties. All descriptive elements are organized into a hierarchical structure by the "part-of" relations. Relations between the objects such as "adjacent" or "occluding" are described between corresponding regions.

IV-3. Modeling and Control Structure for Plan Generation

IV-3-1. Model Representation

IV-3-1-1. Knowledge Block Organization of Model

In a computer vision system, a model which describes the properties of and the relations between the objects is the most important knowledge representation about the task world. In our system, the model is organized as a semantic network as illustrated in Fig. 4-5. Each node of the network is called a knowledge block in our model. It holds a chunk of knowledge about an entity in the world; for instance, object "sky", material "concrete", property "blue", relation "linear-boundary", etc. A knowledge block for an object or material includes a set of rules which describe properties

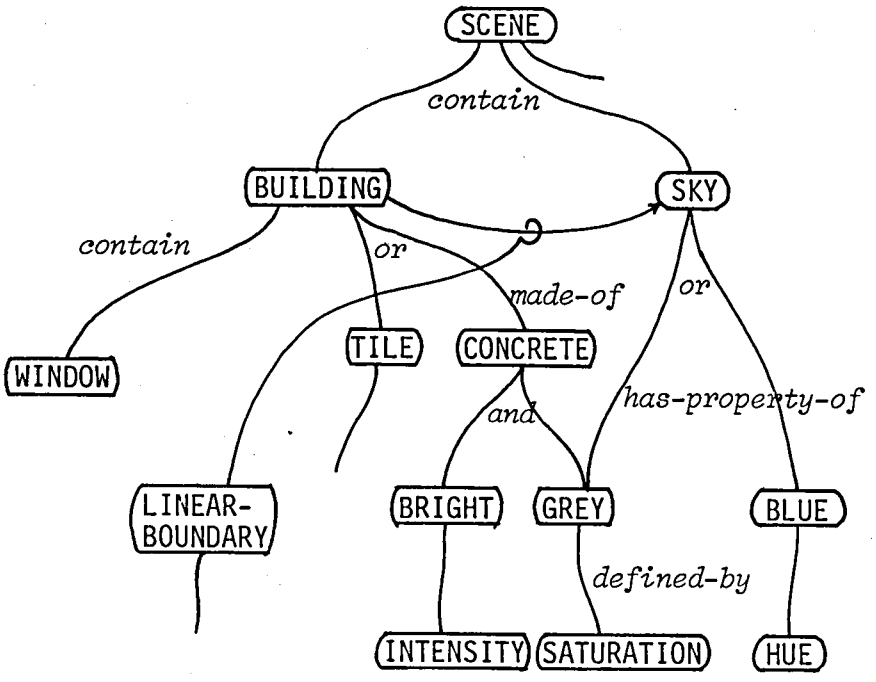


Figure 4-5. A semantic network for knowledge organization.

it must satisfy and relations it has with other blocks, and a set of production rules which belong to it. A knowledge block for a property or relation includes its definition.

The most valuable advantage of our representation over an ordinary semantic network is that it has a mechanism to represent both the universal knowledge, such as property inheritance from material to object, and the "active" rules, such as the production rules, that the interpretation process can use in a specific context of analysis.

IV-3-1-2. Rules Describing Properties and Relations

The knowledge block for an object or material holds the description of the properties which must be satisfied by a region corresponding to that object or material. It also holds the description of the relations which must be satisfied between the regions corresponding to that object and other objects. The properties and the relaltions are represented as a set of declarative rules of the following formats.

property: [(<type><fuzzy-predicate><weight>)(<var-list>)]
relation: [(<type><fuzzy-predicate><weight> FOR <label>)(<var-list>)]

The <fuzzy-predicate> defines the property or the relation itself. Its syntax is as much the same as the form in Lisp language. Figure 4-6 shows the syntax defined by the BNF notation. Some explanatory examples are given later in this section.

The <var-list> is a list of external variables used in the fuzzy predicate. There is exactly one variable in the <var-list> of property rules and two in relation rules. The control program binds

each of the external variables to the regions (or patches) which must be examined by the predicate. The fuzzy predicate evaluates the property of the region or the relation between the two regions, and returns a fuzzy truth-value.

The <weight> is a dotted pair of two figures (W1.W2). It indicates the uncertainty of the knowlegde the rule relies on. Let A be the property represented by the rule, and let X be the object to which the rule belongs. Then the first figure W1 corresponds to the a priori probability P[A] that a region satisfies the property A. W2 corresponds to the conditional probability P[A|X] that a region known to be the object X satisfies the property A.

The <type> discriminates types of knowledge the rule relies on. There are two types: GEN (GENeral) and STR (STeReotyped). The GEN-type rule corresponds to a general type of knowledge, such as "sky is blue or grey". The STR-type rule represents the knowledge about a stereotype, such as "the region which touches the lower side of the picture frame may be a part of road"; of course, we cannot say that a region is not a part of road unless it touches the lower side. The difference between the two types is not very essential. But, this taxonomy is helpful in making the rules and in determining the weight values.

The <label> specifies the object with which the relation should hold.

The following are typical examples of the property rules and the relation rules.

(a) A property rule in the knowledge block "sky";

knowledge: The sky is blue or grey.

rule : [(GEN (OR (*BLUE *SK)(*GREY *SK)) (1.0 . 0.2)) (*SK)]

(b) A relation rule in the knowledge block "building";

```
knowledge: The boundary between the building and the sky has a
           lot of linear parts, and the building is not on the
           upper side of that boundary.
rule     : [(GEN (AND (*LINEAR-BOUNDARY *BL *SK)
                      (NOT (POSITION UP *BL *SK)) )
                 (1.0 . 0.5) FOR SKY) (*BL *SK)]
```

```
<fuzzy predicate>::=<constant>|<variable>|
                    (<function name><arguments>)
<arguments>::=<empty>|<argument><arguments>
<argument>::=<fuzzy predicate>
<function name>::=<block name>|<subroutine name>
<block name>::=*<letter><letters>
<subroutine name>::=<letter><letters>
<variable>::=<block name>
<constant>::=<letter><letters>|<number><numbers>|
             <number><numbers>.<number><numbers>
<numbers>::=<empty>|<number><numbers>
<letters>::=<empty>|<letter><letters>|<number><letters>
<number>::=0|1|2|...
<letter>::=A|B|C|...
```

Figure 4-6. Syntax of fuzzy predicates.

A <block name> used as a <argument> is a <variable>.
A <letter><letters> used as a <argument> is a <constant>.
A <block name> used as a <function name> corresponds to a <fuzzy predicate> whose definition is described in the model.

IV-3-1-3. Definition of Fuzzy Predicates

We defined various pictorial properties such as "green", "grey", "textural", etc. as fuzzy predicates in the model. Those predicates are defined by using the corresponding pictorial features. For instance, "green" is defined by the feature "hue", "grey" by "saturation", and "textural" by "degree of texture". We use three functions, FUZZY1, FUZZY2, and FUZZY3, to map the pictorial feature values onto the fuzzy truth-values. Figure 4-7 illustrates the mapping schemes. FUZZY1 and FUZZY2 have four arguments. The first one is the pictorial feature. The second and third ones are the thresholds T1 and T2 which are shown in Fig. 4-7. Pictorial features taking values between T1 and T2 are mapped onto fuzzy truth-values between the minimum and maximum truth-values. The fourth argument is used to reflect the uncertainty of the pictorial feature to the fuzzy truth-value. Let us consider the fuzzy predicate *GREY for "grey". When the color of a region is very dark, its saturation is unreliable. In such a case, the fuzzy truth-value must be 0.5, which means "nothing is said about the property 'grey'", regardless of its saturation. Then FUZZY1 and FUZZY2 adjust their maximum and minimum truth-values according to the fuzzy truth-value given in its fourth argument.

maximum truth-value = 0.5+(value of the fourth argument)/2 ,
minimum truth-value = 0.5-(value of the fourth argument)/2 . (4-1)

When the fourth argument is omitted, the maximum and minimum truth-values are defaulted to 1.0 and 0.0, respectively. The function FUZZY3 has two more arguments: the thresholds T3 and T4 (see Fig. 4-7).

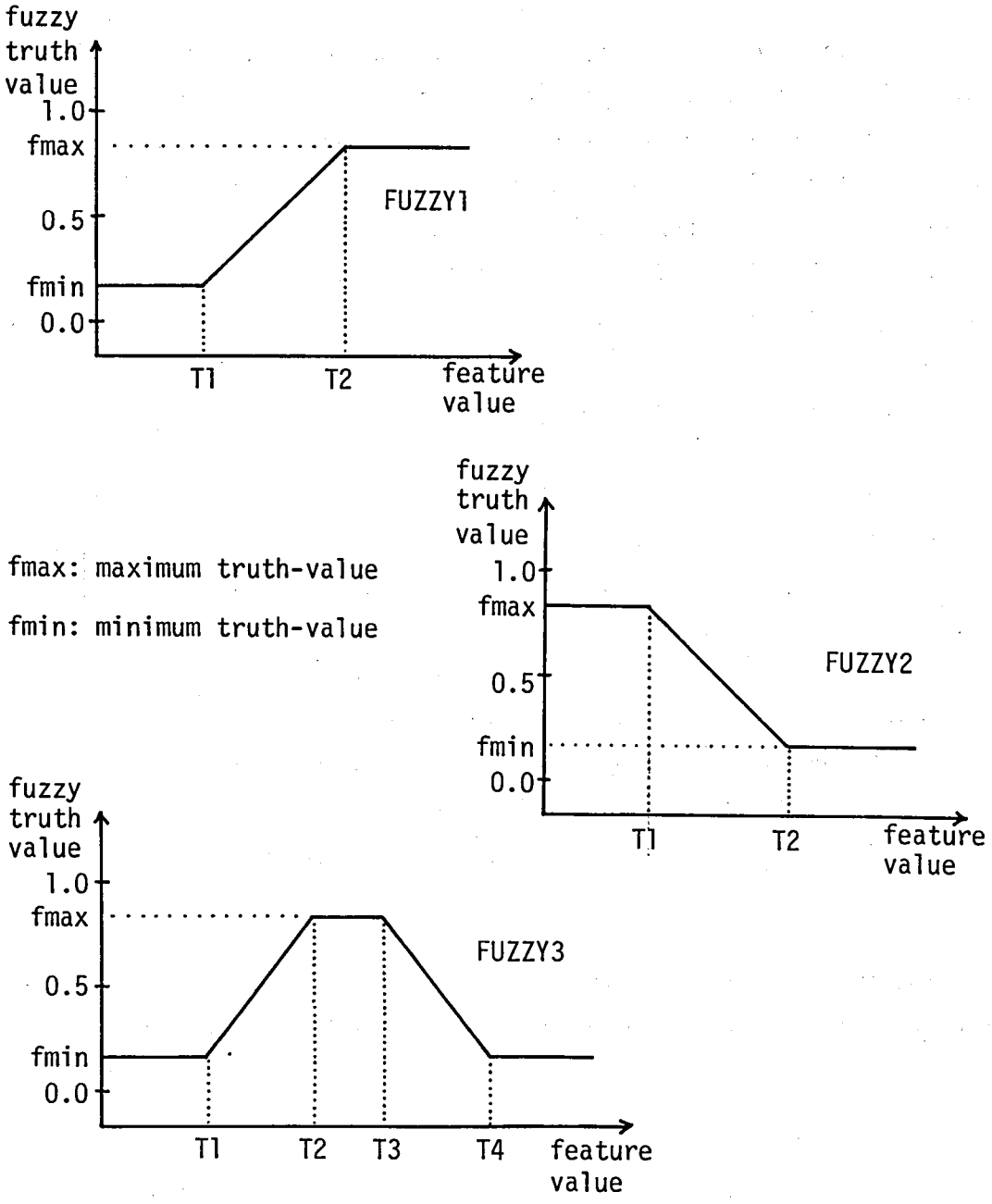


Figure 4-7. Three functions to define truth-value of fuzzy predicates from the feature value.

The followings are explanatory examples of fuzzy-predicate definitions.

```
textural: [(*X)(FUZZY1 (TEXTURE-DEGREE *X) 1.0 4.0 (*BRIGHT *X))]
grey:     [(*X)(FUZZY2 (SATURATION *X) 0.05 0.15 (*BRIGHT *X))]
green:    [(*X)(FUZZY3 (HUE *X) 0.5 1.0 2.5 3.0 (NOT (*GREY *X)))]
```

IV-3-2. Evaluation of Plan

The evaluation of the plan means the evaluation of the degree of correctness of the labels assigned to each region in the plan image. In order to evaluate the plan, the plan manager activates the rules for plan generation to examine the regions in the plan image. Three kinds of figures are used to compute the degree of correctness of a label assigned to a region. They are (1) the a priori probability of the label, (2) the results of the property rules evaluated with the region, and (3) the results of the relation rules evaluated between the region and other regions; these figures are used in this order. The a priori probability given in the model for each object label is used as the base value of the degree of correctness. First, the property rules are used to revise the correctness values. Then, the relation rules are used for further revision. This is because the evaluation of a relation can take place only after the labels of the partner regions are assigned with some confidence.

IV-3-2-1. Calculation of Revision Factor

According to the Bayes rule of conditional probabilities, the following Eq. 4-2 holds. Let P[Xm] be the a priori probability that a

label Xm matches with a region. Let P[Xm|A] be the a posteriori probability that a label Xm matches with a region after a property A is observed with the region. Let P[A] denote the a priori probability that a region satisfies the property A. Let P[A|Xm] be the conditional probability that a region with the label Xm satisfies the property A.

$$P[Xm|A] = P[Xm] \cdot P[A|Xm]/P[A] \cdot \qquad (4\text{-}2)$$

In our region analysis, Equation 4-2 can be interpreted that when a region Qi satisfies a fuzzy-predicate-A in a property rule Rmk

[(<type> fuzzy-predicate-A <weight>)(<var-list>)],

the correctness value of the label Xm for the region Qi can be P[A|Xm]/P[A] times reinforced. Recall that the figures P[A|Xm] and P[A] are given as the <weight> in the rule Rmk.

Let Timk be the fuzzy truth-value of the fuzzy-predicate-A in the rule Rmk evaluated for the region Qi. Timk takes a value of the range [0, 1]; Timk equal to 1 or 0 means that the fuzzy predicate is or is not satisfied totally; Timk between 0 and 1 means that the satisfaction is ambiguous; Timk equal to 0.5 means that nothing is said about the existence of the property. Then, it is necessary to adjust the revision of the correctness value Cim according to the value of Timk. For this purpose, a revision factor Fimk is derived from the P[A|Xm], P[A], and Timk. The Cim is revised by multiplying Fimk in stead of P[A|Xm]/P[A]. Figure 4-8 shows the relation of revision factor vs. fuzzy truth-value.

For the GEN-type rule when the fuzzy truth-value Timk is 1 or 0 (i.e., the property either does or does not exist totally), the correctness value Cim becomes P[A|Xm]/P[A] or $P[\overline{A}|Xm]/P[\overline{A}]$ times reinforced or weakened. When Timk is 0.5 (i.e., nothing is said about the property), the Cim does not change at all (Fimk=1). For the STR-type rule, the correctness value Cim is kept unchanged when the property is not recognized, i.e. Fimk=1 for Timk $\leq$ 0.5.

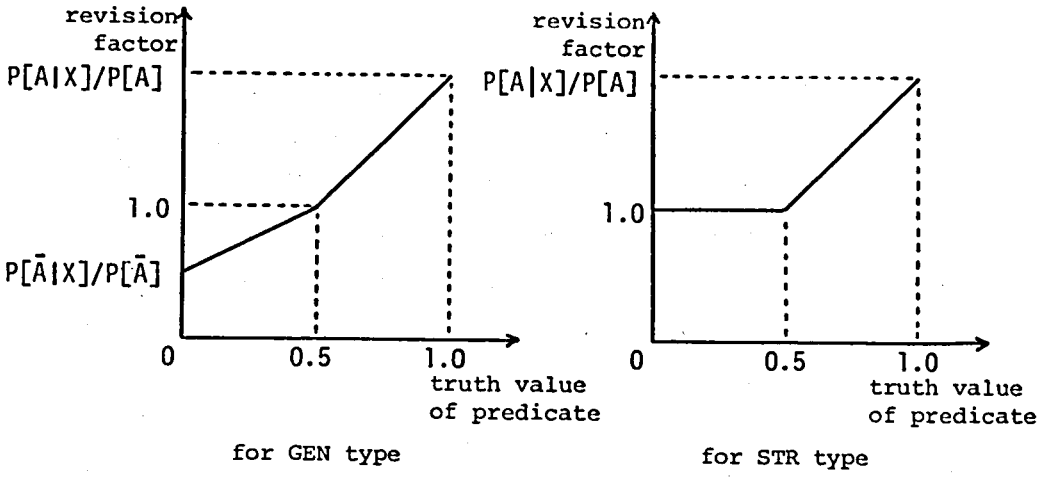


Figure 4-8. Relation of the revision factor vs. the fuzzy truth-value.

IV-3-2-2. Evaluation of Property Rules

In general, there are plural property rules describing the properties which must be satisfied by a region with label Xm. Let Km be the number of the property rules for the label Xm. The computation of a correctness value is performed sequentially; it is revised as the rules are evaluated one by one. Let Fimk (k=1...Km) be the revision factor obtained by evaluating the k-th property rule Rmk for a region Qi. Let $C_{im}^{(0)}$ be the correctness value of the label Xm for the region Qi after evaluating all the property rules (the relation rules have not been used yet). Then $C_{im}^{(0)}$ is calculated by

$$C_{im}^{(0)} = (P[Xm] \cdot \prod_{k=1}^{Km} Fimk) / \sum_{m=1}^{M} (P[Xm] \cdot \prod_{k=1}^{Km} Fimk) , \qquad (4-3)$$

where, P[Xm] is the a priori probability of the label Xm.

Notice that Eq. 4-3 includes the normalization of $C_{im}^{(0)}$ such that $\sum_{m=1}^{M} C_{im}^{(0)} = 1$, where M is the number of labels (objects) in the model.

IV-3-2-3. Evaluation of Relation Rules

The correctness value $C_{im}^{(0)}$ is now revised by evaluating the relation rules Rmnk (k=1...Kmn) which describe the relations between a region with label Xm and another region with label Xn. Let Timjnk be the fuzzy truth-value obtained by evaluating the k-th rule Rmnk for a region Qi with the label Xm and a region Qj with the label Xn. Then the correctness value Cim of the label Xm for the region Qi can be revised by multiplying the revision factor Fimjnk which is calculated from Timjnk and the weight of the rule Rmnk. But the situation is a little different from the case of property rules. The correctness value Cjn of the label Xn in the region Qj must be considered. Actually, Cjn takes a 0-to-1 value. When Cjn is near 0, the evaluation of the rule Rmnk between the regions Qi and Qj makes no sense. Then, the revision factor Fimjnk must be adjusted according to the value of Cjn, as well as Timjnk, when revising the correctness value Cim. Equation 4-4 defines the computation of Cim.

$$Cim = C_{im}^{(0)} \cdot \prod_{j=1}^{J} \prod_{n=1}^{N} \prod_{k=1}^{Kmn} \{(Fimjnk - 1) \cdot Cjn + 1\} , \qquad (4-4)$$

where, J is the number of regions in the plan image,
N is the number of labels in the model,
Kmn is the number of relation rules described between the labels Xm and Xn.

The revision is performed for all pairs of the region Qi and other regions Qj in the plan image, for all labels Xn in Qj, and for all relation rules between the labels Xm and Xn. The Cjn in the right side of Eq. 4-4 is also revised by evaluating the relation rules. Then we would need to solve J x N equations to obtain the exact values of Cim (i=1...J and m=1...N). Here, we employed the relaxation method to solve the equation approximately. Let $C_{im}^{(h)}$ is the value of Cim after the h-th iteration. $C_{im}^{(h)}$'s are computed by the

successive use of Eq. 4-5.

$$C_{im}^{(h)} = C_{im}^{(0)} \cdot \prod_{j=1}^{J} \prod_{n=1}^{N} \prod_{k=1}^{Kmn} \{(Fimjnk - 1) \cdot C_{jn}^{(h-1)} + 1\} ,$$

$$(i=1...J \text{ and } m=1...N) . \quad (4-5)$$

The initial values for $C_{jn}^{(h-1)}$ are $C_{jn}^{(0)}$ which are the correctness values obtained by using the property rules. When $C_{im}^{(0)}$ or $C_{jn}^{(h-1)}$ is small, the values of Fimjnk are not sufficiently reflected to the revision of the Cim. This means the evaluation of relation rules to obtain Fimjnk's in such a situation does not pay much from the computational point of view. In our system, Fimjnk is evaluated only when $C_{im}^{(0)} \geq 0.1$ and $C_{jn}^{(h-1)} \geq 0.5$. Consequently, the number of relation rules which need to be actually evaluated is rather small, and the computation of Eq. 4-5 becomes feasible. Two or three iterations of Eq. 4-5 are enough to produce approximate solutions of Cim.

IV-4. A Production System for Region Growing

IV-4-1. Representing Knowledge by Production Rules

When we import the production system architecture to image analysis, it is an important problem to determine the "size" of knowledge which can be represented by one production rule. An example of "large" knowledge size is a system in which each production rule corresponds to one object to be extracted and has whole knowledge about the object. This enables the performance of skillful analysis

according to the characteristics of each object. On the other hand, the rules become large and complex, which makes it difficult to manage them.

In our system the size of knowledge represented by a single rule is fairly small. Each rule describes a combination of basic operations in region growing: selecting a patch which has not been interpreted yet from the segmented image, assigning a label to it, and assembling it into the scene description. This scheme has the following merits: Each rule is simple and easy to write and modify; the interaction among rules can be performed in a clear way because the access method to the database is uniform.

On the other hand, the patch-by-patch analysis in our scheme gives rise to a certain difficulty in dealing with global constraints such as object shapes or relations among objects. We took three steps to resolve this difficulty. First, we generate a plan as a rough interpretation of the input scene. The plan is put into the database and each production rule can freely see it. This enables the production rules to catch information about global structures of the scene. Secondly, a set of patches can be dealt with at one time, as well as individual patches, and it becomes possible to extract an object which is defined as a combination of mutually constrained patches such as windows of buildings or a car on roads. Lastly, we have devised special rules which extract information from the segmented image without sticking to the patch-by-patch analysis. Typically, this kind of rule is used to extract the shape of a building.

IV-4-2. Control Structure

IV-4-2-1. Scene Phase and Object Phase

Conceptually, each production rule in the production system architectures concurrently checks the status of the database, and executes the associated action when its condition is satisfied. Actually, however, the control program examines one by one every pair of production rules and patches which have not been interpreted yet. The pair which seems to give the most reliable interpretation is to be selected for execution. An agenda is used to schedule the executable pairs at any moment in the analysis. The agenda must be updated whenever the database is changed. Roughly speaking, the number of tests to be done each time is estimated as

$$\text{number of tests} = (\text{the number of un-interpreted patches}) \times (\text{the number of production rules}), \quad (4\text{-}6)$$

which become several thousand. Testing them all is computationally unfeasible.

It is necessary to reduce the number of the patches and production rules which must be actually examined at a time. For this, the structure of scenes must be considered.

A scene usually has two different properties from the view point of image analysis: "Globality" and "Locality". Results of analysis such as the determination of scene horizon or the detection of objects can have significant influence on the analysis of the overall structure of the scene. This property is called "Globality". On the other hand, results of analysis in a small part of an object scarcely have influence on the analysis of other parts in the scene. This property is called "Locality".

These two properties are effectively utilized in the control structure of our scene analyzer. In accordance with the two

properties, the control program works in two phases: scene phase and object phase. The scene phase is for analyzing the overall structure of the input scene without sticking to details. Since, it is almost meaningless to examine small patches in this scene phase, only the keypatches are examined. Whenever a keypatch is labeled, the agenda activates the scene phase and all keypatches that have not been labeled yet are re-examined. In the object phase, the analysis of detailed structures proceeds under the context of the results in the scene phase. When a patch which belongs to an object is labeled, the agenda activates the object phase corresponding to that object, and those patches touching the patch just labeled are examined or re-examined.

The set of production rules can be divided into subsets to be used in the scene phase and the object phase. The production rules for the object phase are further divided into subsets corresponding to each object. In each phase, only the production rules in appropriate subsets are activated by the agenda to examine the patches which have not been interpreted yet.

Consequently, the number of tests to be done at a time is reduced to several 10's.

IV-4-2-2. Control Structure

The control structure of our production system is simple. Basically, the analysis iterates the following three steps.

1. An executable action registered on the agenda is selected and executed. A patch or a set of patches is interpreted, and the database is modified.

2. If a keypatch is interpreted in step 1, the control program enters into the scene phase. The production rules for the scene phase, which are described in the knowledge block SCENE, are activated to (re-)examine the keypatches not yet interpreted. The agenda is updated.
3. The control program enters into the object phase corresponding to the object as which the patch(es) has been just interpreted in step 1. The production rules in the knowledge block of the object are activated to (re-)examine the patches touching the patch(es) just interpreted. The agenda is updated.
   GO TO 1.

This simple control scheme is an important outcome of the production system architecture; each production rule independently checks the database and modifies it whenever the condition is satisfied. However, two problems need to be considered to make this mechanism actually work: scheduling or focussing of attention (a method to direct the analysis to a goal) and conflict resolution (a method to resolve conflict among executable actions which are inconsistent with each other).

IV-4-2-3. Conflict Resolution

In a production system architecture, it is an important and difficult problem to resolve the conflicts between the modules (rules) which work concurrently and independently to modify the database. In our system, each production rule individually tries to assign a label to a patch whenever the patch satisfies the condition

attached to the rule. Then it is usually the case that a patch simultaneously satisfies the conditions of several production rules whose associated actions try to assign different labels to the patch.

Since the basic operation which changes the database in our system is assignment of an object label to a patch, the detection of the inconsistent actions is quite straightforward. Our solution for the problem of conflict resolution is as follows.

(1) Actions which are determined to be executable are registered on the agenda. The agenda schedules the execution of the registered actions by the scheduling method which will be explained next.
(2) Whenever a patch is interpreted by executing an action, every action which is going to give a different interpretation to the patch is decided to be inconsistent and deleted from the agenda.

IV-4-2-4. Scheduling

Every executable action in our system is registered on the agenda with its score, and the action which has the highest score is executed to actually change the database. Thus, the score plays an important role in directing the analysis toward the goal. The score given to an executable action is calculated as the sum of a base value and a premium value.

The base value is a constant given to each production rule. It plays a role in specifying the order of analysis. In the outdoor scene analyzer, the base value of each rule is determined so that the analysis basically proceeds according to the following order: (1) detection of objects in the scene, (2) determination of exact

boundaries between objects and analysis of detailed structures of the objects, (3) analysis of occlusion.

The premium value depends on the degree of satisfaction of the condition part of the production rule when the action part was determined to be executable. It plays a role in guiding the analysis toward the correct interpretation. The fuzzy truth-value of the fuzzy predicate in the condition part and the correctness value of the label on the plan is often used as a premium value.

To sum up, the analysis proceeds toward the goal, guided by the premium values following the strategy specified by the base values.

IV-4-3. Description of Production Rules

IV-4-3-1. Knowledge Blocks and Production Rules

The model in our system is described as a network of knowledge blocks which define the objects, materials, and concepts in the world given as a task. As well as the rules for the plan generation, the production rules are arranged and stored in the network of knowledge blocks. The production rules are divided into subsets according to the role they play in the analysis process. Each subset is stored in a particular knowledge block corresponding to its role; for instance, the subset for the scene phase analysis is stored in the knowledge block SCENE, the subset to analyze the "sky" in the object phase is in the knowledge block SKY, and so on. This enables the agenda to pick up and activate only the appropriate and effective production rules according to the phases in the analysis process.

IV-4-3-2. Format of Production Rules

The production rules in our system have the following format.

[(ACT <fuzzy-predicate> (THEN <action-list>)) <var-list>]

The ACT indicates that the rule is a production rule. The <fuzzy-predicate> is the condition part of the production rule. It examines the database and produces a 0-to-1 fuzzy truth-value. Its syntax is the same as that of fuzzy predicates in the GEN- and STR-type rules used for the plan generation.

The (THEN <action-list>) is the action part of the production rule. The <action-list> is a set of actions to manipulate the database to build the scene description. Each action is described as a form in Lisp, i.e., a list of a function name and its arguments. The <action-list> of to-do rules includes a function to calculate the score to be associated to the action.

The <var-list> is the list of external variables to be used in the fuzzy predicate and the actions. Before evaluating the fuzzy predicate, the control program binds those external variables to regions or patches to be examined by the rule. The number of variables in the <var-list> is 0, 1, or 2. The <var-list> in the to-do rules for the scene phase analysis has exactly one variable, and a keypatch is assigned to it. The <var-list> in the to-do rules for the object phase has two variables; the first variable is bound to a patch which has not been interpreted yet, and the second one to a region belonging to the object corresponding to that phase. In the case of if-done rules, the variables, if any, are bound to the same patch or region that is examined by the to-do rule which triggered the if-done rule.

IV-4-3-3. Description of Actions

The action part of a production rule includes a list of actions which manipulate the scene description. The manipulation is a combination of several simple operations.

(1) Patch-level operation
A patch is assigned with a label --- (P-LABEL <label>).

(2) Region-level operation
The patch is merged with a region --- (R-MERGE <region>).
If the description of the region has not been created yet, a new region is created --- (R-CREATE).

(3) Object-level operation
When a new region is created at the region-level operation, the region is associated with an object. A pointer representing a relation between the region and other regions belonging to that object is set --- (O-MERGE WITH <relation> <region>).
If the description of the object has not been created yet, a new object is created and the region is associated with it --- (O-CREATE).

(4) Scene-level operation
When a new object is created at the object-level operation, the object is registered to the scene --- (S-MERGE).

When a patch is interpreted, one of the following three combinations of actions actually takes place.

(a) P-LABEL & R-MERGE ,
(b) P-LABEL & [R-CREATE] & O-MERGE ,
(c) P-LABEL & [R-CREATE] & [O-CREATE] & [S-MERGE] .

The actions enclosed by [] are defaulted and they are not explicitly described in the model. In the action-list of a production rule, those actions are described as the arguments of the functions which register them on the agenda. We use two functions for this purpose: CONCLUDE and MUST-BE. The function CONCLUDE registers the action specified in its arguments on the agenda with a score which is calculated by the SCORE-IS function in the action-list. The function MUST-BE registers an action which must be executed when the action registered by CONCLUDE in the same action-list is executed. The MUST-BE function allows defining an operation which manipulates a set of patches at a time (see the next section for an example).

IV-4-3-4. Examples

Figure 4-9 shows a to-do rule to be used for the scene phase analysis. It has responsibily to detect a keypatch corresponding to "sky", and to assemble it into the scene description. The fuzzy predicate in the condition part is a fuzzy logical-product of two predicates PROBABLY and NOTFOUND. The predicate PROBABLY refers the plan and examines the correctness value of the label SKY for the keypatch which is assigned to the external variable *PCH. The

```
[(ACT (AND (PROBABLY SKY *PCH)
           (NOTFOUND SKY))
      (THEN (CONCLUDE P-LABEL SKY)
            (CONCLUDE O-CREATE)
            (SCORE-IS (ADD 4.0 (CONFIDENCE-VALUE *PCH)))))
                                                    (*PCH)]
```

Figure 4-9. A to-do rule for "sky" detection.

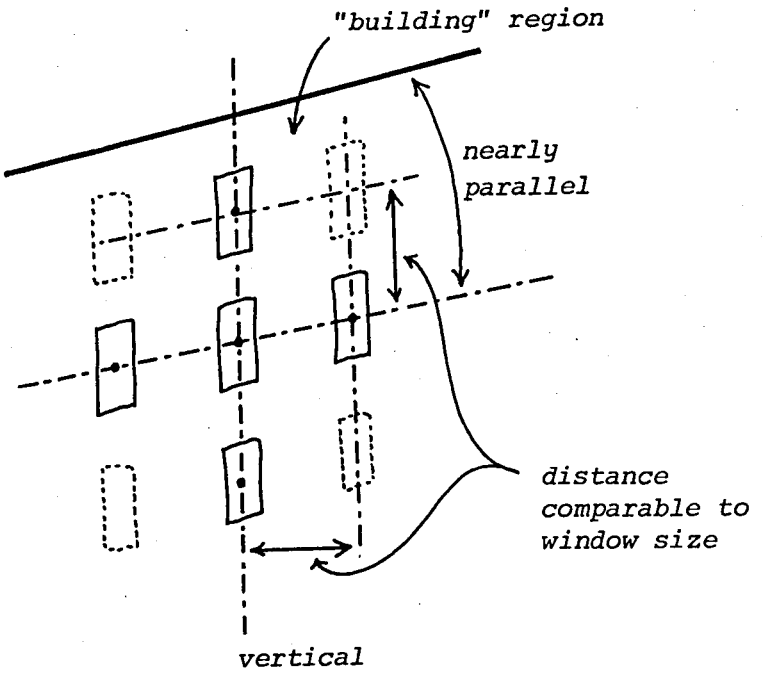
predicate NOTFOUND produces 1.0 (i.e. true), if the description of the object SKY has not been included in the scene description yet.

The action-list of the rule contains two actions: labeling the patch, and creating the description of the object SKY. The function SCORE-IS calculates a score attached to the action to be registered on the agenda. The base value is 4.0 and the premium value is the degree of confidence of the label SKY for the keypatch *PCH.

Figure 4-10 illustrates another example. It is a to-do rule to be evaluated in the object phase to analyze an object "building". It has the responsibility of interpreting "windows", a substructure of the "building". In order to interprete the "windows", it is necessary to assign the label "window" to the all patches that are of rectangle shape and are arranged in a particular way within the area which has been interpreted as "building" (see Fig. 4-10-a). This operation is described by the rule in Fig. 4-10-b. The functions, GET-SET, ALL-FETCH, and THERE-IS, employed in the rule actively fetch a region (patch) or a set of regions (patches) from the database.

The control program binds a patch not yet interpreted to the external variable *PCH, and a region interpreted as a "building" to the external variable *MRGN. The fuzzy predicate in the condition part of the rule takes two roles: (1) It examines the status of the patch *PCH, and (2) it fetches a set of patches which match the condition specified in the predicate. The meaning of each part of the predicate is as follows:

(1) The predicate IS-PLAN checks whether the patch *PCH is included in the plan region of *MRGN. The predicate *VERTICALLY-LONG examines the crude shape of the patch *PCH.
(2) The function GET-SET fetches all the patches that are merged into the plan region corresponding to the region *MRGN. As the result, a set of patches which are included in the area around *MRGN is assigned to the variable *PLSET.
(3) The function ALL-FETCH fetches all the patches that have not



(a) "windows" and "building"

```
[(ACT (IF (AND (IS-PLAN *PCH *MRGN)                          ..... (1)
               (*VERTICALLY-LONG *PCH))
          (THEN (GET-SET *PLSET (PLAN *MRGN) PATCHES)        ..... (2)
                (AND (ALL-FETCH *WLIKE *PLSET                ..... (3)
                     (AND (IS (LABEL *WLIKE) NIL)
                          (*VERTICALLY-LONG *WLIKE)))
                     (ALL-FETCH *WIND *WLIKE                 ..... (4)
                      (THERE-IS *WK *WLIKE
                       (*W-RELATION *WIND *WK))))))
      (THEN (CONCLUDE P-LABEL B-WINDOW)
            (FOR-EACH *WIND (AND (MUST-BE *WIND P-LABEL B-WINDOW)
                                 (DONE-FOR *WIND)))
            (SCORE-IS (ADD 2.1 (DIV (NUMBER-OF *WIND) 100.0)))))
                                                    (*PCH *MRGN)]
```

(b) listing of the to-do rule for "window" detection

Figure 4-10. A to-do rule for "window" detection.

been interpreted yet and whose shape satisfies the predicate *VERTICALLY-LONG. The patches are fetched out of the set *PLSET, and the selected patches are assigned to *WLIKE.

(4) The function ALL-FETCH fetches all the patches that have at least one partner patch within the set *WLIKE with which the relation *W-RELATION holds. The predicate *W-RELATION examines the placement relation which is illustrated in Fig. 4-10-a. A set of patches are fetched out of the patches in the set *WLIKE and assigned to the variable *WIND.

The action part includes the following actions. The function CONCLUDE registers P-LABEL action on the agenda to assign the label B-WINDOW to the patch *PCH. The function MUST-BE also registers on the agenda the labeling operation to the patch in the set *WIND. Further operations to assemble them into the scene description are described by if-done rules in the knowledge-block B-WINDOW. The function SCORE-IS calculates the score for the action. The premium value is the number of patches which are extracted as the "window".

IV-5. Experiments

IV-5-1. Implementation

The analysis scheme described in the preceding sections is applied to obtain meaningful segmentations of outdoor scenes. Each scene is a shot in the campus of Kyoto University. Ordinary 35mm color transparency films are used to take the photographs. A color

flying-spot-scanner is used to digitize the photographs. The digitization is performed by 256 x 256 size with 5-bit or 6-bit density resolution for each of the red, green, and blue components of color.

The systems for the preliminary segmentation and the rule-based analysis are implemented in FORTRAN which is augmented with specially designed functions for image manipulation and list processing.

The facilities for image manipulation enable the users of the FORTRAN system to deal with the image arrays on the disk files as if they were on the 2-D arrays defined in FORTRAN programs.

The facilities for list processing enable the users to write programs in the same fashion as the "program feature" in Lisp. Pointers are manipulated as integer variables in FORTRAN. Elementary Lisp functions such as CONS, CAR, CDR, RPLACA, RPLACD, etc. and list I/O functions are implemented as FORTRAN functions. Moreover, our FORTRAN system allows the recursion in function calling. Thus we can easily implement a system which needs to handle list structures such as the model and the scene description in the framework of FORTRAN.

The objects "sky", "building", "tree", and "road" are defined in the model. The "window" of buildings is also defined as the

Table 4-1. Number of rules in each knowledge block.

	SCENE	SKY	TREE	BUILDING	ROAD	material*	total
property	-	5	2	4	3	6	20
relation	-	2	0	2	3	0	7
production (to-do)	8	3	2	4	4	-	21
(if-done)	2	1	1	5	0	-	9
total	10	11	5	15	10	6	57

** material: CONCRETE, TILE, etc. (cf. Fig. 4-5 in p.101)*

substructures of the "building". The "car" and its "shadow" are defined as the substructures of the "road". The "concrete", "brick", "tile", "asphalt", and "leaves" are defined as the materials of the "building", "road", and "tree". The number of rules described in each knowledge block is shown in Table 4-1. Table 4-2 presents the fuzzy predicates defined and used to describe the rules. The number of predicates for properties is 19, and that for relations is 15. Predicates which examine the plan and the scene description are also used. Table 4-3 shows the functions to derive pictorial features from the image represented on the Patchery Data Structure. The table also includes the functions which are defined to retrieve information from the plan and the scene description. A complete listing of the model is included in the appendix.

IV-5-2. Results

Figure 4-11-a is a digitized input scene. Figure 4-11-b shows the result of preliminary segmentation. The patches with area greater than 300 are selected as keypatches. Figure 4-11-c is the plan image. Figure 4-12 illustrates the plans generated during the analysis of the scene. The brightness of each region indicates the degree of correctness of the labels, SKY, TREE, BUILDING, and ROAD, for that region in the plan image. Figure 4-12-a is the plan evaluated by using only the property rules. Notice that the region corresponding to this side of the building is assigned accidentally a high correctness value for SKY because its color is grey and very bright. Figure 4-12-b shows the plan revised by using the relation rules. The same region now obtains a high correctness value for BUILDING by means of the rules which represent the relation between the

Table 4-2. Fuzzy predicates being used to describe the model.

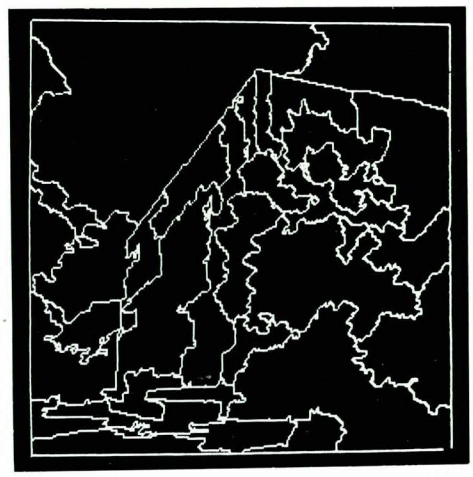
property	color	*DARK, *BRIGHT, *SHINING, *GREY, *VIVID, *RED, *BLUE, *GREEN, *YELLOW	9
	position	*UPPER, *MIDDLE, *LOWER	3
	shape	*HORIZONTALLY-LONG, *VERTICALLY-LONG *MANYHOLE, *MANYLINE, *HOLELINE	5
	texture	*TEXTURAL, *HEAVY-TEXTURE	2
relation	color	*SAME-COLOR, *LOW-CONTRAST, *DARKER	3
	position	*WITH-IN, *CONTACT, TOUCHING, FACING *(HORIZONTALLY, VERTICALLY)*, POSITION *(UP, DOWN)*, ABOVE, BELOW, BETWEEN, *W-RELATION, SAME-ZONE, DIFFERENT-ZONE	11
	shape	*LINEAR-BOUNDARY	1
description		PROBABLY, MAY-BE, NOTFOUND, IS, IS-PLAN	5

Table 4-3. Functions being used to describe the model.

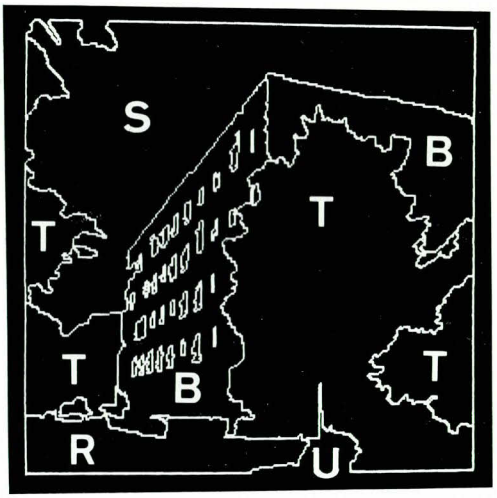
property	color	INTENSITY, SATURATION, HUE, RED-VALUE, GREEN-VALUE, BLUE-VALUE, CONTOUR-CONTRAST	7
	position	MASS-CENTER-X, MASS-CENTER-Y, V-ZONE, MBR-UP-SIDE, MBR-LOW-SIDE, MBR-LEFT-SIDE, MBR-RIGHT-SIDE	7
	shape	AREA, CONTOUR-LENGTH, VH-RATIO, COMPACTNESS, WIDTH-X, WIDTH-Y, HOLE-NUMBER, LINE-DEGREE, HOLE-LINE-DEGREE	9
	texture	TEXTURE-DEGREE	1
relation	color	R-G-B-DIFFERENCE, CROMATIC-DIFFERENCE, BOUNDARY-CONTRAST	3
	position	DISTANCE, ANGLE-DIFFERENCE, O-RATIO	3
	shape	BOUNDARY-LENGTH, BOUNDARY-LINE-DEGREE, T-RATIO	3
description		ALL-FETCH, THERE-IS, T-FETCH, GET-SET, OF, LABEL, REGION, OBJECT, PLAN, MASTER, ASK-VALUE, CONFIDENCE-VALUE	12

(a) digitized input scene

(b) result of preliminary segmentation



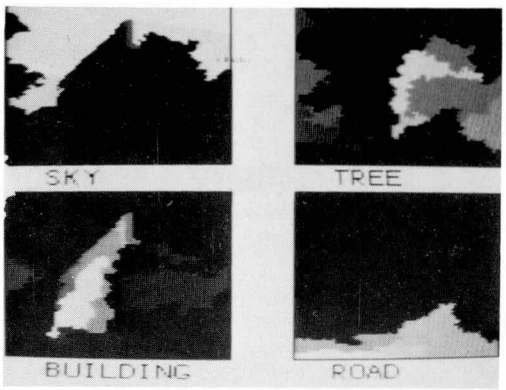
(c) plan image



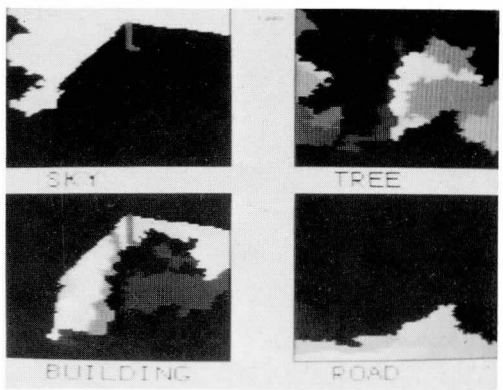
(d) result of meaningful segmentation

S: sky, T: tree, R: road, B: building, U: unknown.

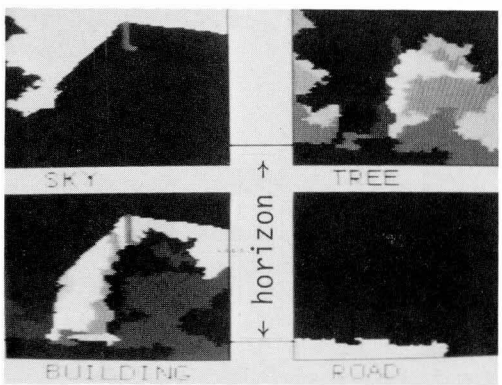
Figure 4-11. Result of the rule-based analysis: Example 1.



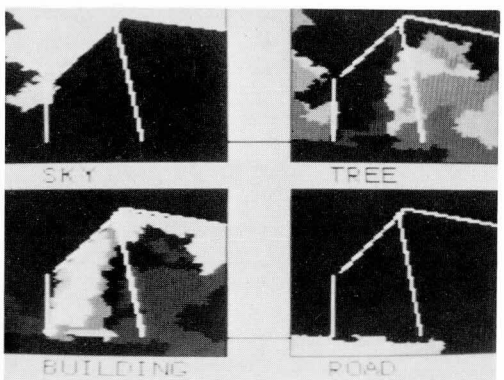
(a) first plan obtained by using only the property rules



(b) after using the relation rules



(c) after extracting the horizon by the top-down analysis

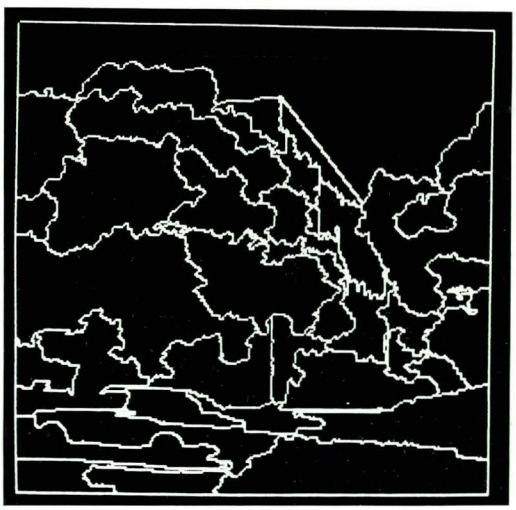


(d) outlines of the building extracted by the top-down analysis

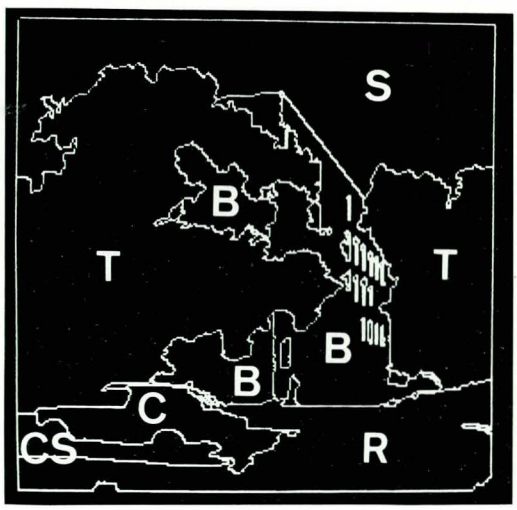
Figure 4-12. Plans generated for the scene of Fig. 4-11.

(a) digitized input scene

(b) result of preliminary segmentation



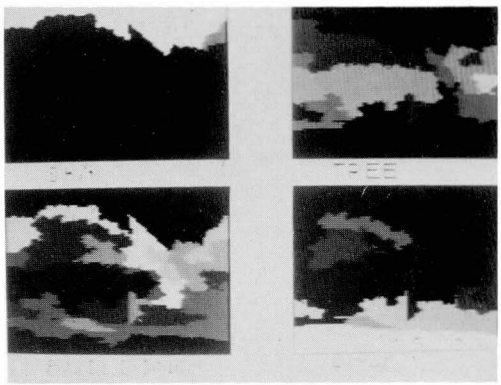
(c) plan image



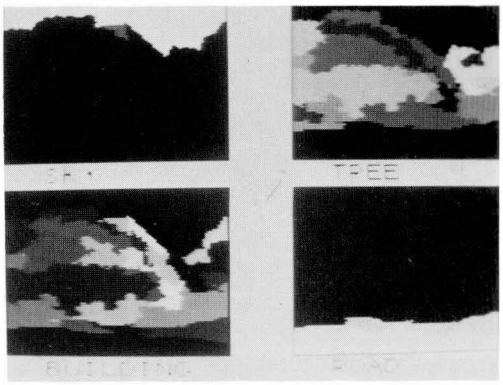
(d) result of meaningful segmentation

S: sky, T: tree, B: building
R: road, C: car,
CS: car shadow.

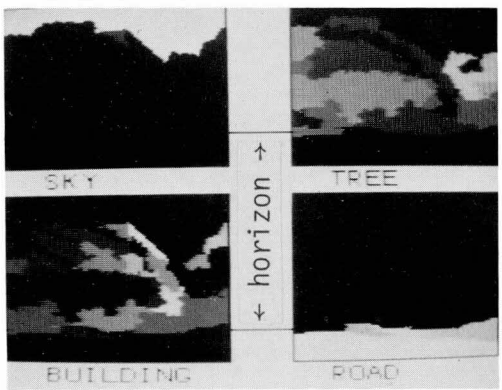
Figure 4-13. Result of the rule-based analysis: Example 2.



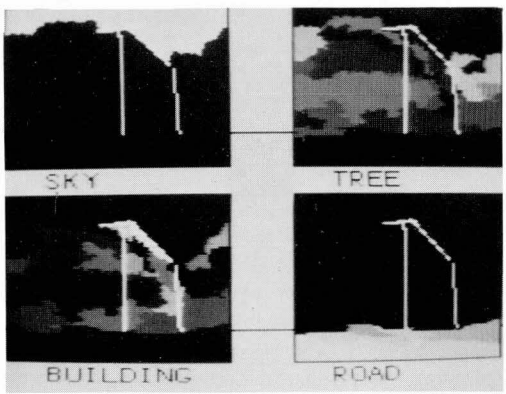
(a) first plan obtained by using only the property rules



(b) after using the relation rules



(c) after extracting the horizon by the top-down analysis

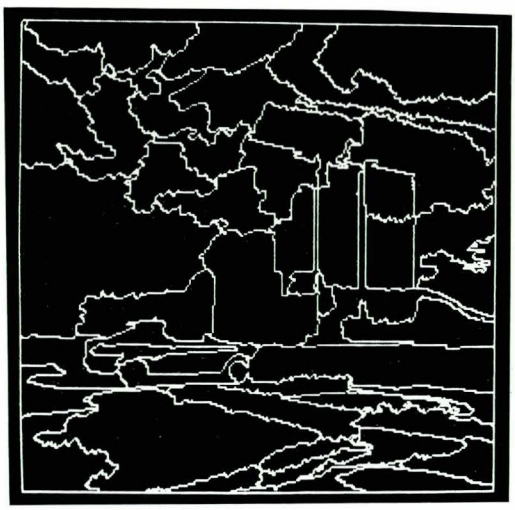


(d) outlines of the building extracted by the top-down analysis.

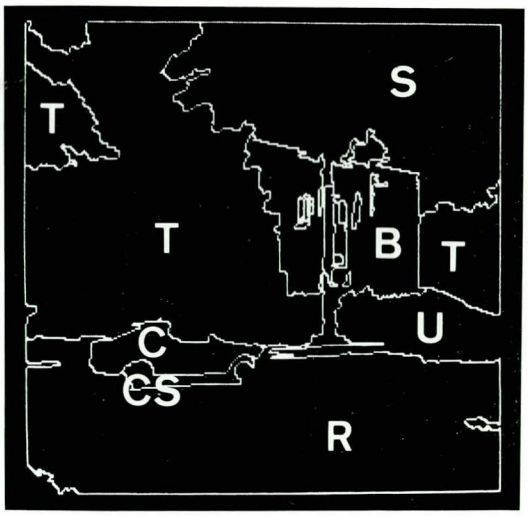
Figure 4-14. Plans generated for the scene of Fig. 4-13.

(a) digitized input scene

(b) result of preliminary segmentation



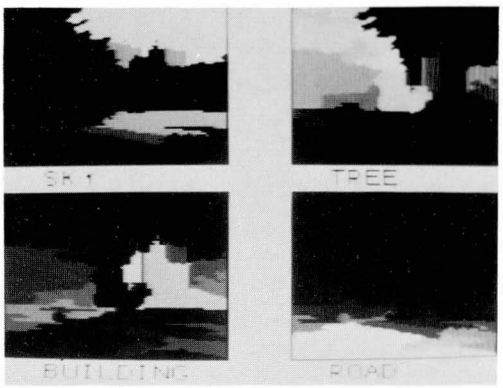
(c) plan image



(d) result of meaningful segmentation

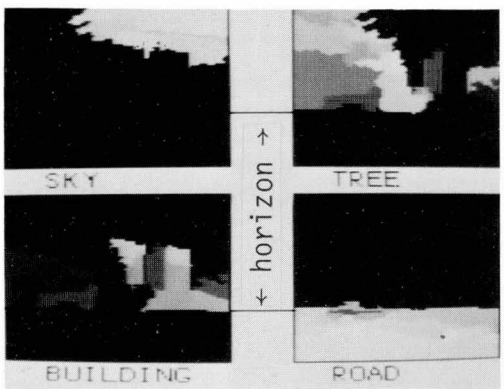
S: sky, T: tree, B: building, R: road, C: car, U: unknown, CS: car shadow.

Figure 4-15. Result of the rule-based analysis: Example 3.

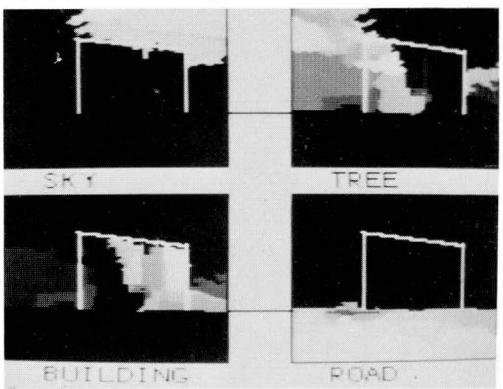


(a) first plan obtained by using only the property rules

(b) after using the relation rules



(c) after extracting the horizon by the top-down analysis



(d) outlines of the building extracted by the top-down analysis

Figure 4-16. Plans generated for the scene of Fig. 4-15.

"building" and the "sky". Figure 4-12-c is the plan revised after the scene horizon was detected by a production rule. The position of the extracted horizon is indicated in the figure. Figure 4-12-d illustrates the outlines of the "building" extracted by an if-done rule which was activated when the object "building" is created. Figure 4-11-d is the final result of segmentation. It plots the contours of the regions and sub-regions in the scene description created through the analysis. The "windows" of the building are successfully analyzed. Note that the images in Figs. 4-11-b, c, d, and 4-12 are generated only for visualization. In our region analyzer all data included in those images are recorded symbolically in the database.

Figure 4-13 shows another example. The scene in Fig. 4-13-a includes a car on the road. Figure 4-13-b is the result of preliminary segmentation. Figure 4-13-c shows the plan image and Fig. 4-13-d shows the segmentation finally obtained. The car and its shadow were successfully analyzed. Figure 4-14 illustrates the plans generated for the scene in Fig. 4-13.

Figures 4-15 and 4-16 show the result for another scene which also includes a car on the road.

IV-6. Conclusion

In this chapter, we have described a rule-based region analyzer which can deal with rather complex scenes including objects with substructures. Outdoor scenes in our university campus were analyzed by the system.

We have achieved the following results.

(1) The top-down control scheme was applied to the region growing method. This enables the region analysis to deal with the detailed structures in the scene, such as the windows of a building, which has been difficult for the so-called semantic region analysis.

(2) The plan of the input scene is generated by the bottom-up control scheme. It is effectively combined with the top-down analysis implemented by using the production system architecture. As the result, a data-driven control mechanism can be realized to improve reliability of the analysis; the portions in the scene which will be analyzed with high confidence are analyzed before other portions with low confidence.

(3) We have developed a scheme for approximate reasoning to handle the four kinds of uncertainty existing in (a) pictorial features, (b) property definitions, (c) object descriptions, and (d) interpretation results. Fuzzy predicates are used for the uncertainty in (a) and (b). The fuzzy truth-values and the uncertainty in (c) and (d) are all incorporated into the computation of the revision factors used for the plan evaluation.

(4) The region growing method has a clear control structure: selecting a region which has not been interpreted yet, assigning an object label to it, and assembling it into the scene description. By applying this structure together with the basic control scheme of the production system architecture, we have achieved a simple and clear knowledge representation scheme for image analysis.

(5) We have utilized the production system architecture in image analysis. Several problems were addressed and solved concerning the modeling, computation, scheduling, etc.

Especially, in order to reduce the computation, two phases (scene phase and object phase) have been established in the control structure based on the two opposite properties (globality and locality) of the scene.

Chapter V

CONCLUSION

We have studied a region analysis system for color scenes. In the system described here, the input image is once converted into a structured data network, and the knowledge to be used in the higher-level analysis is coded as a set of rules which work on this network.

A wide range of issues have been dealt with in this thesis from signal domains to semantic ones.

Color Information

In chapter II, we have discussed about color information in region segmentation. Systematic experiments have been performed to examine the role of color information. A segmentation scheme, called the "dynamic K. L. transformation", was developed for this purpose. We have found a set of color features that can approximate all the color features used in segmenting various color images by the dynamic K. L. transformation. The effectiveness of the color features was verified by a comparative study with various sets of color features from the view point of segmentation and computation. An experiment showing that the color in natural scenes is physically almost two-dimensional was also presented in this chapter.

Signal-level Segmentation

In chapter III, we have presented a system for preliminary segmentation. A powerful segmentation program was developed based on the algorithm which uses multihistograms to find the cutoff values for partitioning. Schemes which are developed to improve the quality of segmentation include a scheme for avioding the fragmentation of textural parts and a scheme for extracting the detailed structures veiled by dominant ones.

Symbolic Data Structure for Segmented Images

The result of segmentation is organized into a well-structured symbolic data network with powerful retrieving facilities. It includes the properties and relations of regions and supports the merging operation of regions. Only the "primary" features are described in the network. The "secondary" features can be derived easily from the primary ones when they become to be neccessary. The system has been applied to various color images.

Rule-based Region Analysis for Interpretation

Chapter IV describes a rule-based region analyzer. Bottom-up control and top-down control were effectively combined in the framework of region growing. The system generates a plan by the bottom-up control as a representation of the rough structures in input scenes. A scheme of approximate reasoning was developed to handle the uncertainty contained in the knowledge and the pictorial data.

A symbolic description of the input scene is made in the top-down analysis. The top-down process was made by using a production system architecture. Employing the region growing as the basic control structure of the production system, we have achieved a

simple and clear knowledge representation scheme for image analysis. In order to reduce the computation needed to manage the production system, two phases (scene phase and object phase) have been successfully established in the control structure utilizing the two opposite properties (globality and locality) of scenes.

Successful analysis of outdoor scenes including sky, trees, buildings, and roads have been demonstrated.

REFERENCES

[1] Bajcsy,R. and Lieberman,L.I. (1974),
"Computer Description of Real Outdoor Scenes",
Proc. IJCPR-II, pp.174-179.

[2] Barrow,H.G. and Popplestone,R.J. (1971),
"Relational Descriptions in Picture Processing",
Machine Intelligence, Vol.6, (Meltzer,B. and Michie,D., eds.),
American Elsevier, New York, pp.377-396.

[3] Brice,C. and Fennema,C. (1970),
"Scene Analysis Using Regions",
Artificial Intelligence, Vol.1, pp.205-226.

[4] Davis,R. and King,J. (1975),
"An Overview of Production System",
AIM-271, Stanford University.

[5] Duda,R.O. and Hart,P.E. (1973),
"Pattern Classification and Scene Analysis",
p.338, John Wiley and Sons.

[6] Freuder,E.C. (1977),
"A Computer System for Visual Recognition Using Active Knowledge",
Proc. IJCAI-V, pp.671-677.

[7] Hewitt,C. (1968),
"PLANNER",
A.I.Memo 168, A.I. Lab. MIT.

[8] Horowitz,S. and Pavlidis,T. (1974),
"Picture Segmentation by a Directed Split-and-Merge Procedure",
Proc. IJCPR-II, pp.424-433.

[9] Kanade,T. (1977),
"Computer Recognition of Human Faces",
ISR-47, Birkhauser.

[10] Kanade,T. (1978),
"Region Segmentation: Signal vs. Semantics",
Proc. IJCPR-IV, pp.95-105.

[11] Kender,J.R. (1976),
"Saturation, Hue, and Normalized Color; Calculation,
Digitization Effects, and Use",
Technical Report, Department of Computer Science, Carnegie-
Mellon University.

[12] Kunii,T. Weyl,S. and Tenenbaum,J. (1974),
"A Relational Data Base Schema for Describing Complex Pictures
with Color and Texture",
Information Processing '74, pp.310-316.

[13] Lee,C.T. and Chang,C.L. (1971),
"Some Properties of Fuzzy Logic",
Information and Control, Vol.19, No.5, pp.417-431.

[14] Marr,D. (1975),
"Early Processing of Visual Information",
A.I.Memo 340, A.I. Lab. MIT.

[15] Nevatia (1976),
"A Color Edge Detector",
Proc. IJCPR-III, pp.829-832.

[16] Ohlander,R. (1975),
"Analysis of Natural Scenes",
Ph.D. Thesis, Department of Computer Science, Carnegie-Mellon
University.

[17] Pavlidis,T. (1979),
"Segmentation by Texture Using a Co-Occurrence Matrix and a Split-and-Merge Algorithm",
Computer Graphics and Image Processing, Vol.10, No.2, pp.172-182.

[18] Platt,W.K. (1978),
"Digital Image Processing",
p.78, Wiley-Interscience.

[19] Preparata,F.P. and Ray,S.R. (1972),
"An Approach to Artificial Non-Symbolic Cognition",
Information Science, Vol.4, pp.65-86.

[20] Riseman,E. et al. (1977),
"I. Segmentation Processes in the VISIONS System",
"II. Model-Building in the VISIONS System",
"III. Between Regions and Objects -- Surfaces and Volumes",
Proc. IJCAI-V, pp.642-647.

[21] Rubin,S.M. and Reddy,R. (1977),
"The Locus Model of Search and its Use in Image Interpretation",
Proc. IJCAI-V, pp.590-595.

[22] Sakai,T. Nagao,M. and Kanade,T. (1972),
"Computer Analysis and Classification of Photographs of Human Faces",
Proc. First USA-JAPAN Computer Conference, pp.55-62.

[23] Sloan,K.R. (1977),
"World Model Driven Recognition of Natural Scenes",
Ph.D. Thesis, The Moore School of Electrical Engineering, University of Pennsylvania.

[24] Tanimoto,S. and Pavlidis,T.(1975),
"A Hierarchical Data Structure for Picture Processing",
Computer Graphics and Image Processing, Vol.4, No.2, pp.104-119.

[25] Tenenbaum,J.M. and Barrow,H.G. (1976),
"Experiments in Interpretation-Guided Segmentation",
Artificial Intelligence, Vol.8, pp.241-274.

[26] Tomita,F. Yachida,M. and Tsuji,S. (1973),
"Detection of Homogeneous Regions by Structural Analysis",
Proc. IJCAI-III, pp.564-571.

[27] Yakimovsky,Y. and Feldman,J.A. (1973),
"A Semantics-Based Decision Theory Region Analyzer",
Proc. IJCAI-III, pp.580-588.

List of Publications and Technical Reports by the Author

Publications in English

1. "Picture Processing System Using a Computer Complex",
T.Sakai, T.Kanade, M.Nagao, and Y.Ohta,
Computer Graphics and Image Processing, Vol.2, No.3/4, pp.207-215, 1973.
2. "A Conversational Picture Processing System by Computer",
T.Sakai, T.Kanade, Y.Ohta, et al.
Information Processing in Japan, Vol.15, pp.70-74, 1975.
3. "KUIPNET: Inhouse Computer Network for Information Processing and Some Applications",
T.Sakai, K.Tabata, T.Kanade, T.Hayashi, S.Kitazawa, H.Ohnishi, and Y.Ohta,
Information Chemistry, S.Fujiwara and H.B.Mark(ed.),
University of Tokyo Press, pp.309-336, 1975.
4. "Color Information for Region Segmentation",
Y.Ohta, T.Kanade, and T.Sakai,
Computer Graphics and Image Processing, 1980, (to appear).

Publications in Japanese

5. "A Conversational Picture Processing System by Computer",
T.Sakai, T.Kanade, Y.Ohta, et al.
Journal of IPSJ, Vol.15, No.12, pp.940-947, Dec. 1974,
(in Japanese).

6. "Structured Description of Color Image Data by Region Splitting",
Y.Ohta, T.Kanade, and T.Sakai,
Journal of the IPSJ, Vol.19, No.12, pp.1130-1136, Dec. 1978,
(in Japanese).
7. "A Region Analyzer Using Bottom-up and Top-down Control",
Y.Ohta, T.Kanade, and T.Sakai,
Trans. of the IPSJ, Vol.21, No.2, pp.116-124, Mar. 1980,
(in Japanese).

Oral Presentations in English

8. "Picture Processing Laboratory and Its Applications",
T.Kanade and Y.Ohta,
Proc. of the IFIP Congress 74, pp.738-742, Aug. 1974.
9. "Model-based Interpretation of Outdoor Scene",
T.Sakai, T.Kanade, and Y.Ohta,
Proc. of the 3rd Int. Joint Conf. on Pattern Recognition,
pp.581-585, Nov. 1976.
10. "An Analysis System for Scenes Containing Objects with
Substructures",
Y.Ohta, T.Kanade, and T.Sakai,
Proc. of the 4th Int. Joint Conf. on Pattern Recognition,
pp.752-754, Nov. 1978.
11. "A Production System for Region Analysis",
Y.Ohta, T.Kanade, and T.Sakai,
Proc. of the 6th Int. Joint Conf. on Artificial Intelligence,
pp.684-686, Aug. 1979.

Oral Presentations in Japanese

12. "Picture Processing System and Its Application",

T.Sakai, T.Kanade, and Y.Ohta,
Proc. of the 4th Joint Conf. on Image Engineering, pp.21-24,
Oct. 1973, (in Japanese).

13. "Conversational Processing of Remotely Sensed Multiband Images",
T.Sakai, T.Kanade, and Y.Ohta,
Proc. of the 15th Joint Convention of the IPSJ, pp.295-296,
Dec. 1974, (in Japanese).

14. "Computer Recognition of Outdoor Scene",
T.Sakai, T.Kanade, and Y.Ohta,
Proc. of the 7th Joint Conf. of Image Engineering, pp.17-20,
Nov. 1976, (in Japanese).

15. "Knowledge Representation and Control Structure for Region Analysis",
T.Sakai, T.Kanade, and Y.Ohta,
Proc. of the 17th Joint Convention of the IPSJ, pp.167-168,
Nov. 1976, (in Japanese).

16. "Computer Recognition of Outdoor Scene -- Image Segmentation and Analysis Using a Descriptive Model -- ",
Y.Ohta, T.Kanade, and T.Sakai,
Technical Report of the Professional Group on Image Processing of the IPSJ, 10-3, Jan. 1977, (in Japanese).

17. "Color Information in Region Segmentation",
Y.Ohta, T.Kanade, and T.Sakai,
Proc. of the 18th Joint Convention of the IPSJ, pp.405-406,
Oct. 1977, (in Japanese).

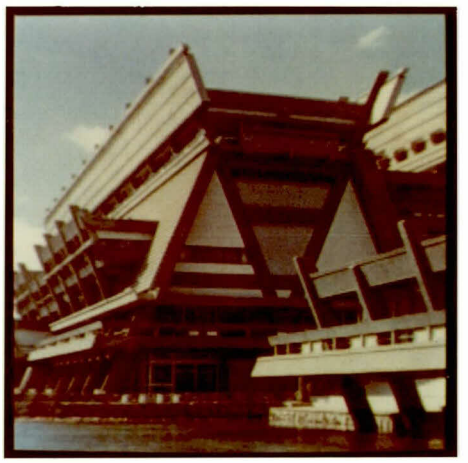
18. "Analysis of Scenes Containing Objects with Substructures",
Y.Ohta, T.Kanade, and T.Sakai,
Proc. of the 19th Joint Convention of the IPSJ, pp.535-536,
Aug. 1978, (in Japanese).

19. "Reports of the Vision Sessions at the 6th IJCAI and a Survey of the Papers",
Y.Ohta,
Technical Report of the Professional Group on Computer Vision of

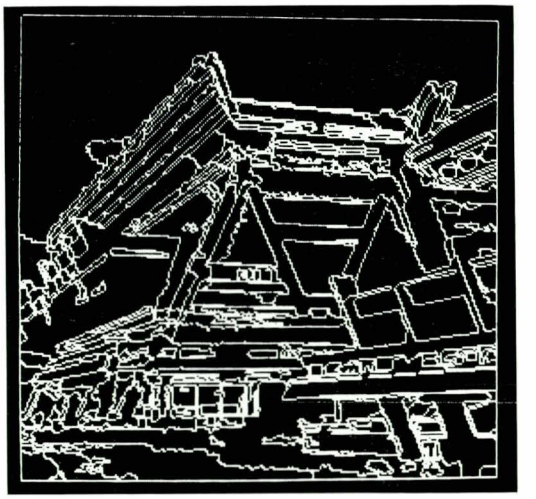
the IPSJ, 2-4, Sep. 1979, (in Japanese).

APPENDIX-A

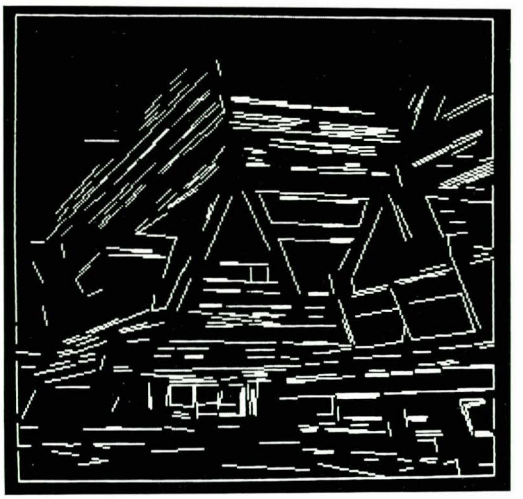
SUPPLEMENTARY RESULTS OF PRELIMINARY SEGMENTATION



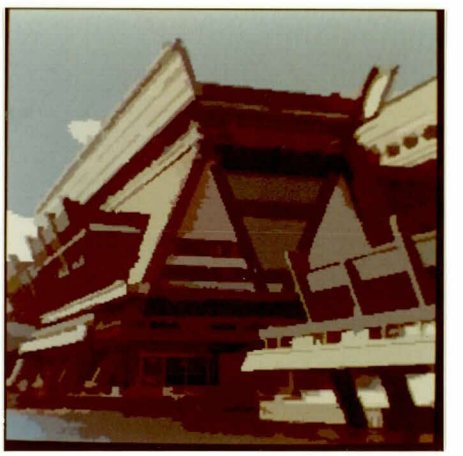
(a) digitized color image



(b) result of segmentation



(c) straight line segments



(d) reconstructed color image

417 regions, 1156 boundary segments, 763 vertices,
477 line segments, 25 holes.

Figure A-1.

(a) digitized color image

(b) result of segmentation

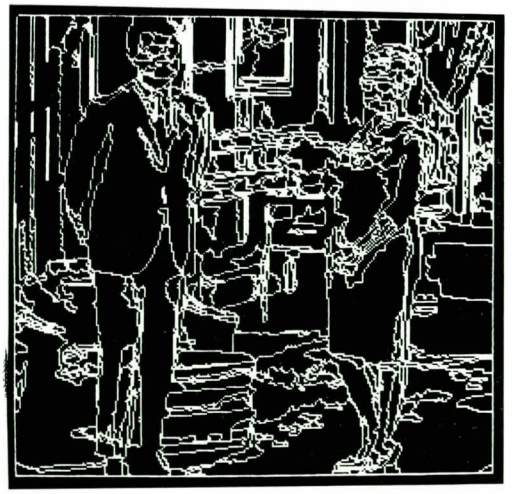
(c) straight line segments

(d) reconstructed color image

415 regions, 1174 boundary segments, 773 vertices, 328 line segments, 15 holes.

Figure A-2.

(a) digitized color image



(b) result of segmentation

(c) straight line segments

(d) reconstructed color image

487 regions, 1374 boundary segments, 908 vertices, 504 line segments, 21 holes.

Figure A-3.

APPENDIX-B

COMPLETE LISTING OF THE MODEL

```
*SCENE  knowledge-block-of-scene

(OBJECTS (*SKY *TREE *BUILDING *ROAD *UNKNOWN)
SUB-OBJECTS (*B-WINDOW *CAR *C-SHADOW)
KEY-PATCH-IS [(GREATERP (AREA *PCH) 300)(*PCH)]

PLAN-IMAGE-GENERATION [(DIV (BOUNDARY-LENGTH *PCH *KPCH)
                            (MULT (R-G-B-DIFFERENCE *PCH *KPCH)
                                  (BOUNDARY-CONTRAST *PCH *KPCH)))
                                                      (*PCH *KPCH)]

IF-PLAN-IS-MODIFIED (IF-DONE (

rule-for-horizon-detection
[(ACT (IF (IS (OF HORIZON (SCENE)) NIL)
          (ALL-FETCH *HRGN *PLAN-REGIONS
           (IF (AND (NOT (PROBABLY ROAD *HRGN))
                    (NOT (TOUCHING *HRGN LOW-SIDE))
                    (ALL-FETCH *WRGN *PLAN-REGIONS
                     (IF (AND (MAY-BE ROAD *WRGN)
                              (ABOVE *HRGN *WRGN)
                              (NOT (*SAME-COLOR *HRGN *WRGN))
                              (FACING HORIZONTALLY *HRGN *WRGN))
                         (MULT (SUB (FACING HORIZONTALLY *HRGN *WRGN)
                                    0.5)
                               (SUB (MIN (ASK-VALUE ROAD *WRGN) 0.6)
                                    (ASK-VALUE ROAD *HRGN) )))))
               (VALUE *WRGN *WRGN) )))
      (THEN (MEMO (SCENE) ROAD-ZONE
                          (WITH (MBR-LOW-SIDE *HRGN) 256 1 256))
            (MEMO (SCENE) HORIZON (MBR-LOW-SIDE *HRGN))
            (EXECUTE PLAN-EVALUATION) )) ] ))

P-SELECT (TO-DO (

rule-for-initial-start
[(ACT (AND (PROBABLY BUILDING *PCH)(NOTFOUND BUILDING))
      (THEN (CONCLUDE P-LABEL BUINDING)
            (SCORE-IS (ADD 4.0 (CONFIDENCE-VALUE *PCH)))))(*PCH)]
[(ACT (AND (PROBABLY ROAD *PCH)(NOTFOUND ROAD))
      (THEN (CONCLUDE P-LABEL ROAD)
            (SCORE-IS (ADD 4.0 (CONFIDENCE-VALUE *PCH)))))(*PCH)]
[(ACT (AND (PROBABLY SKY *PCH)(NOTFOUND SKY))
      (THEN (CONCLUDE P-LABEL SKY)
            (SCORE-IS (ADD 4.0 (CONFIDENCE-VALUE *PCH)))))(*PCH)]
```

```
[(ACT (AND (PROBABLY TREE *PCH)
           (NOT (THERE-IS *TR *REGIONS
                 (AND (IS (LABEL *TR) TREE)
                      (OR (TOUCHING (PLAN *PCH)(PLAN *TR))
                          (*WITH-IN2 (PLAN *PCH)
                                     (V-ZONE2 30 (PLAN *TR))))))))
      (THEN (CONCLUDE P-LABEL TREE)
            (SCORE-IS (ADD 4.0 (CONFIDENCE-VALUE *PCH)))))(*PCH)]

rule-for-adjacent-wall-of-building
[(ACT (AND (MAY-BE BUILDING *PCH)
           (THERE-IS *BL *REGIONS
            (AND (IS (LABEL *BL) BUILDING)
                 (NOT (IS (OF SHAPE VIEW (OBJECT *BL)) 1))
                 (IS (OF ADJACENT (OBJECT *BL)) NIL)
                 (DIFFERNT-ZONE *PCH *BL))))
      (THEN (CONCLUDE P-LABEL BUILDING)
            (CONCLUDE O-MERGE (WITH ADJACENT *BL))
            (SCORE-IS (ADD 5.0 (ASK-VALUE BUILDING *PCH)))))(*PCH)]

rule-for-building-occlusion
[(ACT (AND (MAY-BE BUILDING *PCH)
           (THERE-IS *BL *REGIONS
            (AND (IS (LABEL *BL) BUILDING)
                 (SAME-ZONE *PCH *BL)
                 (*SAME-COLOR *PCH *BL)
                 (THERE-IS *TR *KEYPATCHES
                  (AND (BETWEEN *TR *PCH *BL)
                       (OR (IS (LABEL *TR) TREE)
                           (AND (IS (LABEL *TR) BUILDING)
                                (NOT (IS (OBJECT *BL)
                                         (OBJECT *TR))))))))))
      (THEN (CONCLUDE P-LABEL BUILDING)
            (CONCLUDE O-MERGE (WITH OCCLUDE *BL (REGION *TR)))
            (SCORE-IS (ADD 1.0 (ASK-VALUE BUILDING *PCH)))))(*PCH)]

rule-for-tree-occlusion
[(ACT (AND (*DARK *PCH)(*UPPER *PCH)
           (OR (TOUCHING *PCH UP-SIDE)(TOUCHING *PCH SIDE))
           (THERE-IS *TR *REGIONS
            (AND (IS (LABEL *TR) TREE)
                 (ABOVE *PCH *TR)
                 (TOUCHING *TR SIDE)
                 (*WITH-IN2 *PCH (V-ZONE *TR)))))
      (THEN (CONCLUDE P-LABEL TREE)
            (CONCLUDE O-MERGE (WITH OCCLUDE *TR FRAME))
            (SCORE-IS 1.0)))(*PCH)]

rule-for-tree-garbage
[(ACT (PROBABLY TREE *PCH)
      (THEN (CONCLUDE P-LABEL TREE)
            (SCORE-IS (ASK-VALUE TREE *PCH))))(*PCH)]  ))

P-LABEL (IF-DONE (

if-done-rule-to-be-activated-when-keypatch-is-labeled
[(ACT (NOT (IS (OF PLAN *PCH) NIL))
      (THEN (EXECUTE PLAN-EVALUATION)))(*PCH)]  ))  )
```

```
*SKY  knowledge-block-of-sky

(PROPERTY-RULES (
 [(GEN (NOT (*LOWER *RGN))(1.0 . 0.6))(*RGN)]
 [(GEN (*SHINING *RGN)(1.0 . 0.2))(*RGN)]
 [(GEN (OR (*BLUE *RGN)(*GREY *RGN))(1.0 . 0.2))(*RGN)]
 [(GEN (NOT (*TEXTURAL *RGN))(1.0 . 0.7))(*RGN)]
 [(STR (TOUCHING *RGN UP-SIDE)(0.7 . 0.2))(*RGN)] )

 RELATION-RULES (
 [(STR (AND (*LINEAR-BOUNDARY *RGN *RGN2)
            (IF *LINEAR-BOUNDARY (POSITION DOWN *RGN *RGN2)))
                          (0.0 . 0.5) FOR SKY)(*RGN *RGN2)]
 [(STR (IF (NOT (IS (OF BUILDING-ZONE (SCENE)) NIL))
           (FUZZY1 (O-RATIO *RGN (OF BUILDING-ZONE (SCENE))) 0.3 0.9))
                                        (0.0 . 0.5) FOR SCENE)(*RGN)])

 P-SELECT (

 TO-DO (
 [(ACT (MAY-BE SKY *PCH)
       (THEN (SCORE-IS (ADD 2.0 (ASK-VALUE SKY *PCH)))))(*PCH)]
 [(ACT (AND (IS-PLAN *PCH *MRGN)(*BRIGHT *PCH))
       (THEN (SCORE-IS 3.0)))(*PCH *MRGN)]
 [(ACT (*BRIGHT *PCH)(THEN (SCORE-IS 0.05)))(*PCH)] )

 IF-DONE (
 [(ACT *T* (THEN (CONCLUDE P-LABEL SKY)
                 (CONCLUDE R-MERGE (MASTER *PCH))))(*PCH)] ))

 APRIORI-VALUE-IS 0.1)


*TREE  knowledge-block-of-tree

(MADE-OF (*LEAVES)

 PROPERTY-RULES (
 [(GEN (*MIDDLE *RGN)(0.6 . 0.3))(*RGN)]
 [(STR (*HEAVY-TEXTURE *RGN)(0.8 . 0.2))(*RGN)] )

 P-SELECT (

 TO-DO (
 [(ACT (MAY-BE TREE *PCH)
       (THEN (SCORE-IS (ADD 2.0 (ASK-VALUE TREE *PCH)))))(*PCH)]
 [(ACT (AND (IS-PLAN *PCH *MRGN)(NOT (*SHINING *PCH)))
       (THEN (SCORE-IS 3.0)))(*PCH *MRGN)]

 IF-DONE (
 [(ACT *T* (THEN (CONCLUDE P-LABEL TREE)
                 (CONCLUDE R-MERGE (MASTER *PCH))))(*PCH)] ))

 APRIORI-VALUE-IS 0.2)


*BUILDING  knowledge-block-of-building

(MADE-OF (OR *CONCRETE *TILE *BRICK)
 SUB-OBJECTS (*B-WINDOW)
```

```
PROPERTY-RULES (
[(GEN (*MIDDLE *RGN)(0.6 . 0.3))(*RGN)]
[(STR (*MANYHOLE *RGN)(0.8 . 0.2))(*RGN)]
[(STR (*MANYLINE *RGN)(0.4 . 0.2))(*RGN)]
[(GEN (*HOLELINE *RGN)(0.9 . 0.5))(*RGN)] )

RELATION-RULES (
[(GEN (AND (*LINEAR-BOUNDARY *RGN *RGN2)
           (IF *LINEAR-BOUNDARY (NOT (POSITION UP *RGN *RGN2))))
                                    (0.8 . 0.4) FOR SKY)(*RGN *RGN2)]
[(STR (IF (NOT (IS (OF BUILDING-ZONE (SCENE)) NIL))
          (AND (O-RATIO *RGN (OF BUILDING-ZONE (SCENE)))
               (*MANYLINE *RGN)))
            (0.9 . 0.3) FOR SCENE)(*RGN)] )

P-SELECT (

TO-DO (
[(ACT (AND (MAY-BE BUILDING *PCH)(SAME-ZONE *PCH *MRGN))
      (THEN (CONCLUDE P-LABEL BUILDING)
            (CONCLUDE R-MERGE *MRGN)
            (SCORE-IS (ADD 2.0 (ASK-VALUE BUILDING *PCH)))))
                                                    (*PCH *MRGN)]
[(ACT (AND (NOT (IS-PLAN *PCH *MRGN))(SAME-ZONE *PCH *MRGN)
           (MAY-BE BUILDING (PLAN *PCH)))
      (THEN (CONCLUDE P-LABEL BUILDING)
            (CONCLUDE R-MERGE *MRGN)
            (SCORE-IS (ADD 1.95 (ASK-VALUE BUILDING (PLAN *PCH))))))
                                                    (*PCH *MRGN)]

rule-for-window-extraction
[(ACT (IF (AND (IS-PLAN *PCH *MRGN)(SAME-ZONE *PCH *MRGN)
               (*VERTICALLY-LONG *PCH)(*CONTACT *PCH (PLAN *MRGN)))
          (THEN (GET-SET *PLSET (PLAN *MRGN) PATCHES)
                (AND (ALL-FETCH *WLIKE *PLSET
                       (AND (IS (LABEL *WLIKE) NIL)
                            (SAME-ZONE *WLIKE *MRGN)
                            (*VERTICALLY-LONG *WLIKE)
                            (*CONTACT *WLIKE (PLAN *MRGN))))
                     (THERE-IS *WK *WLIKE (*W-RELATION *PCH *WK))
                     (ALL-FETCH *WIND *WLIKE
                      (THERE-IS *WK *WLIKE
                       (*W-RELATION *WIND *WK)))))
      (THEN (CONCLUDE P-LABEL B-WINDOW)
            (FOR-EACH *WIND (AND (MUST-BE *WIND P-LABEL B-WINDOW)
                                 (DONE-FOR *WIND)))
            (SCORE-IS (ADD 2.1 (DIV (NUMBER-OF *WIND) 100.0)))))
                                                    (*PCH *MRGN)]
[(ACT (AND (IS-PLAN *PCH *MRGN)(SAME-ZONE *PCH *MRGN))
      (THEN (CONCLUDE P-LABEL BUILDING)
            (CONCLUDE R-MERGE *MRGN)
            (SCORE-IS 2.0)))(*PCH *MRGN)]

O-MERGE (IF-DONE (
[(ACT *T* (DESCRIBE-BUILDING (REGION *PCH)))(*PCH)] ))

O-CREATE (IF-DONE (
[(ACT *T* (THEN (EXTRACT-BUILDING-SHAPE (REGION *PCH))
                (DESCRIBE-BUILDING (REGION *PCH))
                (EXECUTE PLAN-EVALUATION)))(*PCH)] ))

APRIORI-VALUE-IS 0.2)
```

```
*ROAD  knowledge-block-of-road

(MADE-OF (OR *ASPHALT *CONCRETE)
 SUB-OBJECTS (*CAR *C-SHADOW)

 PROPERTY-RULES (
 [(GEN (*LOWER *RGN)(0.8 . 0.4))(*RGN)]
 [(GEN (*HORIZONTALLY-LONG *RGN)(0.7 . 0.2))(*RGN)]
 [(STR (TOUCHING *RGN LOWER-SIDE)(0.9 . 0.2))(*RGN)] )

 RELATION-RULES (
 [(STR (AND (*SAME-COLOR *RGN *RGN2)(TOUCHING *RGN *RGN2))
                                  (0.9 . 0.2) FOR ROAD)(*RGN *RGN2)]
 [(GEN (IF (NOT (IS (OF HORIZON (SCENE)) NIL))
           (O-RATIO *RGN (OF ROAD-ZONE (SCENE))))
                                  (1.0 . 0.3) FOR SCENE)(*RGN)]
 [(STR (IF (NOT (IS (OF HORIZON (SCENE)) NIL))
           (NOT (GREATERP (ADD (MBR-UP-SIDE *RGN) 10)
                          (OF HORIZON (SCENE)))))
                                  (0.0 . 0.5) FOR SCENE)(*RGN)])

 P-SELECT (

 TO-DO (
 [(ACT (IF (PROBABLY ROAD *PCH)
           (THEN (AND (*LOW-CONTRAST (PLAN *PCH)(PLAN *MRGN))
                      (NOT (OR (ABOVE *PCH *MRGN)
                               (BELOW *PCH *MRGN))))))
       (THEN (CONCLUDE P-LABEL ROAD)
             (CONCLUDE R-MERGE *MRGN)
             (SCORE-IS (ADD *PREDICATE-VALUE 2.9))))(*PCH *MRGN)]

 rule-for-car-extraction
 [(ACT (IF (MAY-BE ROAD *PCH)
           (THEN (AND (*HORIZONTALLY-LONG *PCH)
                      (*DARK *PCH)
                      (*DARKER *PCH *MRGN)
                      (POSITION UP (PLAN *PCH)(PLAN *MRGN))
                      (THERE-IS *CLIKE *KEYPATCHES
                       (AND (IS (LABEL *CLIKE) NIL)
                            (NOT (IS *CLIKE *PCH))
                            (*HORIZONTALLY-LONG (PLAN *CLIKE))
                            (*WITH-IN (PLAN *CLIKE)
                                      (V-ZONE (PLAN *PCH)))
                            (POSITION UP (PLAN *CLIKE)
                                         (PLAN *PCH)))) )))
       (THEN (ALL-FETCH *WK *KEYPATCHES
              (AND (IS (LABEL *WK) NIL)
                   (NOT (IS *WK *PCH))
                   (*HORIZONTALLY-LONG (PLAN *WK))
                   (*CONTACT2 (PLAN *WK)(PLAN *CLIKE))
                   (*LINEAR-BOUNDARY (PLAN *WK)(PLAN *CLIKE))
                   (*SAME-COLOR *WK *CLIKE)
                   (*WITH-IN (PLAN *WK)(V-ZONE (PLAN *PCH)))))
             (CONCLUDE P-LABEL C-SHADOW)
             (GET-SET *PLSET (PLAN *PCH) PATCHES)
             (FOR-EACH *PLSET
              (IF (NOT (IS *PLSET *PCH))
                  (THEN (MUST-BE *PLSET P-LABEL C-SHADOW)
                        (MUST-BE *PLSET R-MERGE *PCH)
                        (DONE-FOR *PLSET))))
             (GET-SET *PLSET (PLAN *CLIKE) PATCHES)
             (FOR-EACH *PLSET
              (AND (MUST-BE *PLSET P-LABEL CAR)
                   (MUST-BE *PLSET R-MERGE *CLIKE)
                   (DONE-FOR *PLSET)))
```

```
              (FOR-EACH *WK
               (AND (GET-SET *PLSET (PLAN *WK) PATCHES)
                    (FOR-EACH *PLSET
                     (AND (MUST-BE *PLSET P-LABEL CAR)
                          (MUST-BE *PLSET R-MERGE *CLIKE)
                          (DONE-FOR *PLSET)))))
              (SCORE-IS 2.95)))(*PCH *MRGN)]
 [(ACT (MAY-BE ROAD *PCH)
       (THEN (CONCLUDE P-LABEL ROAD)
             (CONCLUDE R-MERGE *MRGN)
             (SCORE-IS (ADD 2.0 (ASK-VALUE ROAD *PCH)))))(*PCH *MRGN)]
 [(ACT (AND (IS-PLAN *PCH *MRGN)
            (OR (*DARK *PCH)(*GREY *PCH)))
       (THEN (CONCLUDE P-LABEL ROAD)
             (CONCLUDE R-MERGE *MRGN)
             (SCORE-IS 3.0)))(*PCH *MRGN)] ))

 APRIORI-VALUE-IS 0.1)


*UNKNOWN          knowledge-block-for-unknown-object

(APRIORI-VALUE-IS 0.1)


*B-WINDOW         knowledge-block-of-windows-of-building

(P-LABEL (IF-DONE (
 [(ACT (AND (T-FETCH *WK *PCH)
            (THERE-IS *WIND *WK (IS (LABEL *WIND) B-WINDOW)))
       (THEN (CONCLUDE R-MERGE *WIND)))(*PCH)] )))


*CONCRETE
(PROPERTY-RULES (
 [(GEN (AND (*BRIGHT *RGN)(*GREY *RGN))(0.6 . 0.2))(*RGN)] ))


*ASPHALT
(PROPERTY-RULES (
 [(GEN (AND (*DARK *RGN)(*GREY *RGN))(0.6 . 0.2))(*RGN)] ))


*TILE
(PROPERTY-RULES (
 [(GEN (AND (*YELLOW *RGN)(NOT (*VIVID *RGN)))(0.6 . 0.2))(*RGN)] ))


*BRICK
(PROPERTY-RULES (
 [(GEN (*RED *RGN)(0.6 . 0.1))(*RGN)] ))


*LEAVES
(PROPERTY-RULES (
 [(GEN (OR (*GREEN *RGN)(*YELLOW *RGN))(0.9 . 0.4))(*RGN)]
 [(GEN (*TEXTURAL *RGN)(0.7 . 0.4))(*RGN)] ))


*UPPER
(PROPERTY-DEFINITION ((*X)(FUZZY2 (MASS-CENTER-X *X) 50 150)))
```

```
*MIDDLE
(PROPERTY-DEFINITION ((*X)(FUZZY3 (MASS-CENTER-X *X) 0 100 150 250)))

*LOWER
(PROPERTY-DEFINITION ((*X)(FUZZY1 (MASS-CENTER-X *X) 150 220)))

*DARK
(PROPERTY-DEFINITION ((*X)(FUZZY2 (INTENSITY *X) 30 50)))

*BRIGHT
(PROPERTY-DEFINITION ((*X)(FUZZY1 (INTENSITY *X) 25 45)))

*SHINING
(PROPERTY-DEFINITION ((*X)(FUZZY1 (BULE-VALUE *X) 50 60)))

*TEXTURAL
(PROPERTY-DEFINITION ((*X)(FUZZY1 (TEXTURE-DEGREE *X)
                                  1.0 4.0 (*BRIGHT *X))))
*HEAVY-TEXTURE
(PROPERTY-DEFINITION ((*X)(FUZZY1 (TEXTURE-DEGREE *X)
                                  5.0 7.0 (*BRIGHT *X))))
*GREY
(PROPERTY-DEFINITION ((*X)(FUZZY2 (SATURATION *X)
                                  0.05 0.15 (*BRIGHT *X))))
*VIVID
(PROPERTY-DEFINITION ((*X)(FUZZY1 (SATURATION *X)
                                  0.1 0.2 (*BRIGHT *X))))
*RED
(PROPERTY-DEFINITION ((*X)(FUZZY3H (HUE *X)
                                  5.0 5.5 0.1 0.6 (NOT (*GREY *X)))))
*BLUE
(PROPERTY-DEFINITION ((*X)(FUZZY3 (HUE *X)
                                  2.5 3.0 4.5 5.0 (NOT (*GREY *X)))))
*GREEN
(PROPERTY-DEFINITION ((*X)(FUZZY3 (HUE *X)
                                  0.5 1.0 2.5 3.0 (NOT (*GREY *X)))))
*YELLOW
(PROPERTY-DEFINITION ((*X)(FUZZY3H (HUE *X)
                                  5.8 0.0 1.0 1.5 (NOT (*GREY *X)))))
*HORIZONTALLY-LONG
(PROPERTY-DEFINITION ((*X)(FUZZY2 (VH-RATIO *X) -1.5 1.5)))

*VERTICALLY-LONG
(PROPERTY-DEFINITION ((*X)(FUZZY1 (VH-RATIO *X) -1.5 1.5)))

*MANYHOLE
(PROPERTY-DEFINITION ((*X)(FUZZY1 (HOLE-NUMBER *X) 0 10)))

*MANYLINE
(PROPERTY-DEFINITION ((*X)(FUZZY1 (LINE-DEGREE *X) 0.2 0.5)))

*HOLELINE
(PROPERTY-DEFINITION ((*X)(IF (GREATERP (HOLE-NUMBER *X) 2)
                              (FUZZY1 (HOLE-LINE-DEGREE *X) 0.2 0.5))))
*SAME-COLOR
(PROPERTY-DEFINITION
    ((*X *Y)(FUZZY2 (CROMATIC-DIFFERENCE *X *Y) 0.015 0.055)))

*WITH-IN
(PROPERTY-DEFINITION ((*X *Y)(FUZZY1 (O-RATIO *X *Y) 0.5 0.7)))

*WITH-IN2
(PROPERTY-DEFINITION ((*X *Y)(FUZZY1 (O-RATIO *X *Y) 0.0 0.1)))
```

```
*LINEAR-BOUNDARY
(PROPERTY-DEFINITION
    ((*X *Y)(IF (TOUCHING *X *Y)
            (FUZZY1 (BOUNDARY-LINE-DEGREE *X *Y) 0.2 0.5
                    (FUZZY1 (BOUNDARY-LENGTH *X *Y) 10 30)))))

*CONTACT
(PROPERTY-DEFINITION ((*X *Y)(T-RATIO *X *Y)))

*CONTACT2
(PROPERTY-DEFINITION ((*X *Y)(FUZZY1 (T-RATIO *X *Y) 0.1 0.3)))

*LOW-CONTRAST
(PROPERTY-DEFINITION
    ((*X *Y)(FUZZY2 (BOUNDARY-CONTRAST *X *Y) 1.0 2.0)))

*DARKER
(PROPERTY-DEFINITION
 ((*X *Y)(FUZZY1 (SUB (INTENSITY *Y)(INTENSITY *X)) -10 10)))

*W-RELATION
(PROPERTY-DEFINITION
    ((*X *Y)(IF (AND (NOT (IS *X *Y))(NOT (TOUCHING *X *Y)))
                (OR (AND (FUZZY3 (ANGLE-DIFFERENCE *X *Y 90)
                                                  -30 -10 10 30)
                         (FUZZY3 (DISTANCE *X *Y)
                                 0 (MULT (WIDTH-X *X) 2.0)
                                 (MULT (WIDTH-X *X) 4.0)
                                 (MULT (WIDTH-X *X) 6.0)))
                    (AND (FUZZY3 (ANGLE-DIFFERENCE *X *Y
                                   (OF SHAPE HL THETA *MRGN))
                                 -45 -15 15 45)
                         (FUZZY3 (DISTANCE *X *Y)
                                 0 (MULT (WIDTH-Y *X) 2.0)
                                 (MULT (WIDTH-Y *X) 4.0)
                                 (MULT (WIDTH-Y *X) 6.0)))))))


NIL
```