LINGUISTICALLY ASSISTED RECOGNITION OF PATTERNS FROM THEIR MULTIPLE TRANSLATIONAL ENCODEMENTS

Thesis for the Degree of Ph. D.
MICHIGAN STATE UNIVERSITY
WILLIAM ARTERBURN BURDETTE
1972

ABSTRACT

LINGUISTICALLY ASSISTED RECOGNITION OF PATTERNS
FROM THEIR MULTIPLE TRANSLATIONAL ENCODEMENTS

By

William Arterburn Burdette

The classical pattern recognition approach to identifying an unknown picture as one of a finite number of prototypes is based on the extraction of features from a single, arbitrary, discretized encodement of that unknown picture. For this approach to be effective, the encodement grid resolution must be fine enough to insure that distinguishing detail is reflected in the encodement of each of the prototype pictures -- regardless of the relative positioning of that picture with respect to the encodement grid. However, in practical situations where such a sufficiently fine resolution may not be feasible, alternative approaches to picture identification are needed.

One such alternative approach investigated in this thesis characterizes a picture of fixed orientation relative to an arbitrary encodement resolution in terms of (1) all the distinct encodement patterns which can result by encoding that picture in all possible relative translational positionings with respect to the encodement grid, and (2) a probability of occurrence for each of these

distinct encodement patterns given a random placement of the encodement grid over the picture. This characterization is called the snapshot signature of the picture of fixed orientation relative to the specified encodement resolution, and constitutes a composite representation of the information contained in all the distinct positional encodements of that picture.

Snapshot signature characterizations are considered for two-dimensional line drawings. A procedure is presented for approximating the snapshot signatures of such pictures whose structural complexity may make the theoretical determination of their snapshot signatures difficult. This procedure is based on a linguistic algorithm which generates the encodement of a picture in an arbitrarily specified position relative to a given encodement grid, using rules resembling the productions of a phrase structure grammar.

The utility of the snapshot signature characterization is demonstrated in picture identification at coarse encodement resolutions where the classical pattern recognition approach fails. Further, it is shown that memory savings can sometimes be realized by using sequential statistical picture identification procedures based on snapshot signatures at coarse resolutions, rather than equally reliable feature extraction techniques at finer resolutions. Hence,

snapshot signature characterizations can provide a useful alternative to increasing encodement resolution for picture identification.

LINGUISTICALLY ASSISTED RECOGNITION OF PATTERNS FROM THEIR MULTIPLE TRANSLATIONAL ENCODEMENTS

By

William Arterburn Burdette

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

1972

TO

MY PARENTS

ACKNOWLEDGMENTS

I wish to express my gratitude to Dr. Carl V. Page for serving as my guidance committee chairman and thesis advisor. He willingly devoted a generous portion of both professional and personal time to discussions concerning this research, for which I am most appreciative. His patience, encouragement, and valuable assistance in directing the course of this research are gratefully acknowledged.

I also wish to thank Dr. Robert O. Barr, Dr. Julian Kateley, Jr., Dr. John B. Kreer, and Dr. Morteza A. Rahimi for serving on my guidance committee and reviewing this thesis, as well as for the many inspiring classes which they provided me as a student.

For the many educational opportunities and generous financial support provided throughout my graduate studies, I am highly grateful to: the National Science Foundation; the National Aeronautics and Space Administration; and the Division of Engineering Research, the Department of Computer Science, and the Graduate Office at Michigan State University.

I wish to thank my typist, Miss Patricia J. Martens, for her conscientious and dedicated preparation of this

manuscript. Also, thanks go to Dr. Floyd E. LeCureux for providing the excellent final drawings of the figures in this thesis. For the personal sacrifices made by both of these individuals to insure the rapid completion of this final draft, I am deeply indebted.

Very special thanks go to my wife Marjorie for her understanding, support, patience, and loyalty throughout this research endeavor. Her love and encouragement provided a special source of inspiration and incentive, which in large measure sustained my efforts throughout my education.

TABLE OF CONTENTS

Chapter Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

Automatic machine processing of pictorial information has become increasingly apparent in recent years. The capability of the modern digital computer to rapidly process large amounts of data has been effectively utilized for the recognition of patterns in a variety of applications. For example, textural patterns occurring in satellite cloud imagery have been successfully analyzed from video data obtained by the TIROS and NIMBUS meteorological satellites [V-1]. Features of the lunar terrain have also been successfully analyzed using video patterns obtained from Air Force Lunar Atlases [V-1]. More common applications of pattern recognition schemes are evident in optical character recognition done by bank check processors, automatic mail sorters (including mail with handwritten addresses), and document readers [P-1]. Medical applications of pattern recognition have also been demonstrated, including automatic sleep state classification of human subjects from their electroencephalograph recordings [V-1].

The overall problem in many pattern recognition situations is that of identifying, or classifying, an

actual picture or scene relative to a finite set of prototypes, where the identification is based on the information included in one or more inexact representations of that actual scene. These inexact representations are usually necessary because the machines involved in data collection or analysis can physically represent or process only a finite amount of information. Hence, when two-dimensional scenes are to be automatically processed, they are often discretely encoded on some finite two-dimensional grid, with individual grid squares being shaded with varying degrees of intensities (gray levels) to reflect the nature of the scene in the region covered by that grid square ([R-4] and [D-2]). In the case of line drawings, this shading is generally done with two gray levels, resulting in binary grid encodements. A grid square is shaded in such an encodement if any part of the scene view lies within the borders of that grid square; otherwise, the grid square is left unshaded. Clearly, the resolution of the grid (i.e., its relative fineness) determines the degree of accuracy with which the binary encodement represents the structural features of the actual scene view.

In any practical picture processing situation, the optimal encodement grid resolution would be the one which is as coarse as possible to eliminate the processing of unnecessary detail, yet fine enough to reflect the significant structure of a scene so that it can be unambiguously

identified or classified. Of course, the degree of encodement accuracy needed for unambiguous identification is highly dependent upon the class of scenes involved, and also upon the method used to make the identification. There are two basic methods which have been used for making this identification in pattern recognition [N-1]: (1) the statistical "decision-theoretic approach," which classifies an unknown scene view based on structural features which can be extracted (measured) from its encodement and then compared statistically to known values of these features for a finite set of prototypes; and (2) the linguistic or "descriptive approach," which analyzes or identifies an unknown scene view in terms of the structural relationships which exist between its component parts or the component parts of known prototypes, based on a "picture language" which formally describes the structural relationships which can exist for the scenes under consideration. Combinations of these two approaches have also been proposed, and fall under the general classification of "syntax-aided analysis" (as opposed to "syntax-directed" or "syntax-controlled" analysis, which is just approach (2)) [N-2]. However, in all of these approaches only a single, arbitrary encodement of the unknown scene view is processed. Further, it has generally been the case that the encodement resolution was heuristically picked, under the assumption that the resolution could be made as fine as necessary for the identification procedure to be used

effectively ([D-2], [H-2], [H-3], [N-2], [V-1]). This last assumption has practical repercussions in cases where grid resolutions may be constrained by the physical aspects of the situation. For instance, an encodement grid might be realized by a remote sensing T-V camera, which has a maximum resolution inherently determined by its design. Or the encodement grid might be realized by photographic film, whose resolution has a practical limit determined by its grain size. So it may not always be feasible to make the grid resolution arbitrarily fine in scene identification situations, and alternative solutions to this problem must be considered.

One classically proposed alternative to increasing the grid resolution is to improve the identification methods used at a given resolution by finding features which have a higher discrimination quality between the prototype classes ([D-2], [F-3], [L-2], [T-1]), or by "sharpening" the image at the current resolution by various image enhancement techniques ([L-2], [R-2], [R-3]). However, such approaches are still based on the examination of a single, arbitrary encodement of the unknown scene view at the encodement resolution, and thus ignore the effect that the relative positioning of the encodement grid may have on the fidelity of an encodement. But since the form of the encodement of a scene can be significantly affected by the relative positioning of the encodement grid (especially at coarse resolutions), it seems reasonable to

expect that additional information about the structure of the unknown scene view could be obtained by examining more than one of its encodements at a given resolution. Hence, this thesis focuses on the investigation of how these multiple encodements of an unknown scene at a given resolution can be utilized in classifying that unknown scene with respect to a finite number of prototype classes. To maintain a reasonable level of complexity, this investigation is limited to examining only those multiple encodements which can result from translations of an encodement grid over a scene having a fixed rotational orientation. However, the methods developed for this restricted case can be generalized to the cases where rotational considerations are important.

The approach taken here in the identification of unknown scenes using their multiple translational encodements at some grid resolution is a "linguistically-aided" approach [N-2] -- i.e., both the linguistic and statistical decision theoretic concepts of scene analysis are combined. The linguistic techniques are used to generate approximate encodement characterizations for each of the prototype scene classes relative to some encodement resolution whenever these encodement characterizations cannot be conveniently determined theoretically (as would be the case for scenes having a complex structure). Such encodement characterizations -- theoretical or approximate -- are based upon the various structural

encodement forms which can arise, and how likely it is that each one will occur, when the encodement grid is translated over the different scenes in the prototype classes. Now, given these encodement characterizations for the prototype classes, statistical decision-theoretic techniques can then be applied to identify an unknown scene view by comparing an approximation of its encodement characterization (based on a random sample of its different translational encodements) with the "known" encodement characterizations for the prototype classes. The effectiveness of such a combined approach to scene identification based on multiple translational encodements is demonstrated in this thesis. In particular, examples are presented to illustrate the capability of this approach to correctly identify unknown scenes at coarse encodement resolutions where other pattern recognition schemes often fail.

1.1 Literature Survey

The "classical" approach to the recognition of patterns by machine divides the process into three basic parts [D-2]: (1) "representation" (encoding or sensing), (2) "feature extraction" (the selection and measurement of significant attributes), and (3) "classification" (the identification or decision process). The gray-level quantization with discretized pictures (already mentioned) is the predominant "representation" scheme used in the

literature for line drawing and textural analyses. Such encodement representations have been shown [R-4] to have significant redundancy (in terms of information content) in many situations, and "picture compression" (data compression) can be done with more "efficient encoding" techniques -- such as "block coding," "predictive coding," or "run coding" -- which more directly reflect the information content in a discretized representation of a scene. A detailed description of such "picture coding" and "picture approximation" techniques can be found in [R-3].

The "feature extraction" portion of the classical pattern recognition approach deals with taking selected measurements -- called "features" -- from an encodement. These "features" should represent significant attributes which can be used to distinguish between the various classes of prototype scenes under consideration. Then, the values of the features extracted from the encodement of an unknown scene view can be used effectively to identify or classify that scene. For instance, significant features in a character recognition situation might be the number of vertical (or nearly vertical) line segments in a character, or the relative degree of straightness along the edge of a character [D-2]. In any event, this selection of significant features to be used for classification is a difficult problem, and various approaches to its solution are discussed in [D-2], [F-3], and [T-1]. The identification of the various "features" of an unknown

scene from its encodement can also be a difficult task, since a discrete encodement can only reflect degraded structural characteristics of the actual scene. Further, in practical situations, the distortion and noise introduced by physical sensing machinery or data transmission channels can further degrade the quality of encodement representations which must be analyzed. Many techniques have been introduced for "sharpening" a picture before features are extracted, and such techniques come under the broad classification of "image enhancement". Two excellent surveys on image enhancement techniques related to future extraction can be found in [L-2] and [R-4], with more detailed presentations given in [A-1] and [R-3].

Numerous "decision rules" have been proposed for implementing the classification stage of the "classical" approach to pattern recognition. One of the most straightforward is known as "template matching" ([F-2], [F-3]) in which the entire encodement of an unknown scene is compared with known encodements (templates) of a set of prototype scenes. The comparison is based upon a "preselected matching or similarity criterion", and the prototype scene view whose template most closely matches the encodement of the unknown scene view is the one with which the unknown scene is identified. This "template-matching approach can be interpreted as a special case [of the] 'feature-extraction' approach, where the templates are

stored in terms of feature measurements and a special classification criterion (matching) is used for the classifier."[1] Deterministic classification techniques have also been proposed which are based on "discriminant functions" [F-3] defined for each prototype scene class. These discriminant functions operate on a vector of feature measurements taken from the encodement of an unknown scene, and the prototype scene class whose discriminant function has the largest value is the one with which the unknown scene is identified. Other classification techniques which have become increasingly popular in recent years [N-1] are statistically based. Such techniques result from formulating the scene recognition problem as a hypotheses testing problem, and then using standard statistical hypotheses testing procedures for performing scene "classification." These standard statistical hypotheses testing procedures are described in [H-4], [M-1], [M-4], [W-1], and [W-3], and their application specifically to pattern recognition situations is covered in [F-3] and [F-4]. These statistical techniques are discussed in more detail in Chapter 4.

Several extensions to the three part classical approach to pattern recognition have been considered.

---

[1] K. S. Fu, Sequential Methods in Pattern Recognition and Machine Learning (New York: Academic Press, 1968), p. 3.

These relate to the consideration of contexts in an encodement [D-2], the "optimum decision problem" (determining the "best" classification techniques for a given situation), and the "adaptation problem" (determining a classification technique which can learn from, and adjust to, various types of picture recognition environments) [T-1]. The "adaptation" considerations border on the area of artificial intelligence [M-3], and are especially important in remote sensing applications (e.g., unmanned spacecraft). Discussions concerning training and learning in pattern classifiers can be found in [F-3] and [F-4].

Another basic approach to pattern recognition has evolved over the past ten years, due to the need for more powerful and more general methods which are applicable to complex picture recognition situations. This approach is the "linguistic" one, based on the description of a picture in terms of its component parts and the relationships which exist between these components [F-6]. Such a description might either be "generative" (i.e., useful for generating some representative of the described picture class) or "interpretive" (i.e., the result of the analysis of a given scene view) [N-1]. Most of the schemes which have been devised for formal picture descriptions are based on the rewriting systems, or grammars, of formal language theory ([G-2], [H-5]). This is a consequence of the fact that the "heirarchical structure" among the subpatterns which are components of a larger pattern is often

"analogous to the syntactic structure of languages", and formal language theory provides a convenient description for such structure (via the grammar) and a method for analyzing this structure (parsing) [F-6]. Hence, the name "syntactic" picture processing has been adopted.

Since formal language theory has been developed on the basis of one-dimensional strings of characters, some of the first attempts at syntactic picture processing involved the encoding of a 2-dimensional pattern into a one-dimensional string. One well known procedure for doing this is Freeman's "chain encoding" method for line patterns, in which

> "...a rectangular grid is overlaid on the two-dimensional pattern and straight line segments are used to connect the grid points falling closest to the pattern... Each line segment is assigned an octal digit according to its slope. The pattern is then represented by a (possibly multiply connected) chain or chains of octal digits."[2]

Feder [F-1] has used this encoding scheme to form pattern languages based on equations in two variables (lines, circles, etc.) and pattern properties (convexity, etc.). A somewhat more general approach to picture description (and analysis) was presented by Shaw [S-1] in his "Picture Description Language." Shaw uses strings to encode

---

[2] K. S. Fu and P. H. Swain, "On Syntactic Pattern Recognition," in Software Engineering, ed. by Julius T. Tou (New York: Academic Press, 1971), pp. 158-159.

pictures which can be represented by multiply connected graphs, but this method has been criticized because it does not permit any contextual considerations to be considered in either picture generation or analysis [F-6].

One of the major problems with 1-dimensional string representations of higher dimensional patterns is that each pair of characters in the string ("features", or primitives, of the pattern) are related in exactly one way, according to their relative position in the string. Hence, only a single relationship can be expressed between any two primitive features of the pattern. The practical problems which such a restriction can cause are evident in [M-2], where a robot is described which needs to identify multiple relationships between the primitives of a scene in order to identify that scene. A method for linguistically considering multiple contextual relationships in generating scene descriptions is presented in [D-1], in which encodement patterns in the form of simple geometric figures (e.g. a triangle) are generated on a grid. During the generation, symbols are introduced on the grid on the basis of the symbols which already appear in up to all eight of its neighboring grid squares. Unfortunately, there was no discussion regarding how one might use these two-dimensional descriptions in a parsing scheme.

A "natural" generalization of a string language to two-dimensions is represented by the "web grammars" defined by Pfaltz and Rosenfeld [P-4]. The language

defined by a web grammar consists of directed graphs with symbols at their vertices ("webs"), rather than a one-dimensional string of characters. The web generation rules can be used to successively imbed subwebs into a larger web, enabling a complex two-dimensional pattern to be described. Since the embeddings can be done based on the consideration of two or more nodes (vertices) at a time, contextual considerations can be reflected in web grammar descriptions. Hence, if each directed edge between two nodes in a web is considered as a primitive of a pattern, the web grammar can be considered as a generalization of Shaw's "Picture Description Language" which allows contextual considerations.

Several two- (and higher -) dimensional linguistic description schemes which are capable of reflecting multiple (and often arbitrary) relationships between scene primitives have recently been presented. One scheme uses the concept of a "relationship matrix" [R-1] to describe the features, and relationships existing between these features, of arbitrary n-dimensional scenes (particularly line drawings). In the two-dimensional case, the features of the scene (e.g. line segments, points, or even objects) label both the rows and columns of the 2-dimensional relationships matrix, and each entry in the matrix is a vector specifying significant relationships (e.g. angle, distance, connectivity, etc.) which exist between the two features which are the labels of that row and column in the

matrix. The on-diagonal entries in the relationship matrix describe the features of the scene themselves (e.g. their length or dimensions, their orientation, etc.). Grammars based on the generation and manipulation of arbitrary relationship matrices can be used to produce scene descriptions (in relationship matrix form), and such grammars have proven useful in identifying an unknown representative of four prototype classes of handwritten characters through a "parse" of the unknown character's structural relationships [R-1]. While the relationship matrix concept can be used as the data base for syntax-directed generation and analysis of actual scene views, a "picture processing grammar" has been proposed by Chang [C-1] which can be used to generate and analyze arbitrary patterns of shaded grid squares on a two-dimensional encodement grid. Such patterns can be considered as those which might result from a discretized representation of an actual scene view. The grammar uses coordinates assigned to each square of the encodement grid to generate descriptions of these patterns based on certain significant groupings of the shaded grid squares -- e.g., those which form horizontal strings, or "lines", on the encodement grid, which can be described by the special symbol "h-line" followed by parameters which specify both the length of (i.e. the number of shaded grid squares in) the string and the coordinates of the leftmost shaded grid shaded grid square in the string. The construction of picture processing

grammars which can generate a given set of shaded grid square patterns on an encodement grid was considered by Chang, and such a grammar constructed for describing the set of patterns which represent the encodements of hand-written numerals (0 through 9). This grammar was then used to analyze arbitrary samples of encoded handwritten numerals, resulting in the correct identification of approximately 70% of the samples [C-1]. Other methods of syntactically processing scene encodements have recently been developed, and an excellent survey of these methods can be found in [P-2] and [P-3].

One interesting pattern recognition scheme which is neither statistical nor linguistic was proposed by Weiman and Rothstein [W-2]. The scheme is based on using parallel processing cellular automata to recognize straight line patterns on an n x n grid. The encodement of a straight line is given by a string of binary digits which can be constructed by tracing the line from left to right on the encodement grid, and writing a "0" for each grid square crossed through parallel sides and a "1" for each grid square crossed through perpendicular sides, omitting the next grid square crossed in this latter case. Such a code reflects the relative positioning of a line with respect to the grid squares, and cyclic shifts occur in the code as the grid is translated over the line to produce different encodements. The structure of the code and its translational cyclic shifting pattern can be used

to characterize the line on the encodement grid. This characterization is then used in [W-2] as the basis for various procedures to recognize straight lines, topological connectedness, and arbitrary polygons from their representations on an encodement grid.

1.2 Contributions of the Thesis

The overall objective of this thesis is to investigate how the combined information contained in the multiple translational encodements which are possible for a particular scene (under various relative translational positionings of the encodement grid) at a given resolution can be effectively utilized in pattern recognition. Such combined multiple encodement considerations have not been previously dealt with in the literature,[3] so the investigation within this thesis involves a unique approach

---

[3] Multiple translational encodements of straight lines were considered by Weiman and Rothstein [W-2], where the primary concern was with representing a two-dimensional encodement with a 1-dimensional code, and then noting the cyclic changes which occurred in the code as the encodement grid was translated relative to the straight line. Although such a code does reflect the basic structural features of each encodement, only straight lines of rational slope could be fully characterized on a finite grid by the cyclic changes in their codes. Even then, the length of the line was always assumed to be sufficient to span the entire grid in any translational position.

The characterization of scenes in terms of their multiple encodements is considered at a much more general level in this thesis. The two dimensional representation for each of the encodement patterns which can result by translating an encodement grid over an (arbitrary) scene is retained (so that arbitrary relationship information can be maintained). Further, the relative degree

in pattern recognition technology. The utility of this approach is specifically demonstrated relative to the scene classification problem at coarse resolutions, where other pattern recognition schemes often fail.

As a result of this new approach and the linguistically aided scheme with which it is investigated, several other contributions are made by this research. These include:

(1) the definition of some new characterizations of scenes in terms of all their possible encodement forms relative to a given resolution, as well as the formalization of some previously intuitive concepts (e.g., resolution) of picture processing;

---

of persistence (reflected as a probability of occurrence in Chapter 4) with which each distinct encoding pattern appears as the grid is uniformly translated over the scene is also an impotrant part of the charcterization of a scene in terms of its multiple encodements. Such "persistence" (probabilistic) properties were not considered by Weiman and Rothstein, but these properties form a quite powerful part of a positional encodement characterization of a scene relative to some grid resolution. In fact, these persistence properties not only permit the rational slope line characterization constraints to be relaxed, but also permit finite length line segments of any size to be fully characterized on a finite encodement grid. Additionally, many complex scenes can also be characterized (at least relative to other scenes of interest) with respect to almost any encodement resolution. The full detail of this general characterization of a scene in terms of its multiple translational encodements is presented in Chapter 4.

(2) the proposal of a new 2-dimensional linguistic description scheme (denoted "interpretive coordinate grammar") which is used to generate an encodement of a specified scene (line drawing) relative to an arbitrarily chosen resolution. This differs from Chang's "picture processing grammar" [C-1] -- which generates an arbitrarily specified set of encodement forms on a grid -- in that the encodements generated by the "interpretive coordinate grammar" are based on an actual scene, and a specified relative positioning of that scene, with respect to the encodement grid. Although the form of the rules in the "interpretive coordinate grammar" represent an extension of the (coordinate) form of the rules in Chang's "picture processing grammar", the rules of the "interpretive coordinate grammar" -- unlike Chang's rules -- require a semantic evaluation of the coordinates after each step in the generation of a scene description. This is due to the complexity involved in considering encodements generated from a specified positioning of an actual scene view relative to an encodement grid -- a consideration which is ignored by Chang;

and (3) the application of standard sequential statistical decision-theoretic hypothesis testing

procedures to scene identification, based on the multiple translational encodement characterizations of such scenes.

1.3 Organization of the Thesis

Chapter 2 defines the basic terminology which is used in this pattern recognition investigation, and formalizes some of the intuitive concepts of picture processing. Chapter 3 concentrates on a two-dimensional linguistic description scheme for generating an encodement of a scene relative to an arbitrary resolution, based on a standardized representation of that scene in some fixed orientation and relative translational position with respect to the encodement grid. Chapter 4 discusses the theoretical characterization of scenes in terms of their multiple translational encodements, and presents techniques for approximating these characterizations based on the linguistic description scheme of Chapter 3. The utility of these characterizations in scene recognition situations is then demonstrated within the framework of statistical hypothesis testing. Chapter 5 concludes the thesis with a summary of results and suggestions for future work.

Chapter 2

SCENES AND THEIR GRID ENCODEMENTS

2.1 Basic Concepts

The general class of pictures, or "scenes," to be considered throughout this thesis are plane geometric line drawings. A finite, symmetrical square grid can be superimposed over a scene, and the scene then encoded into a two-dimensional grid pattern by shading only those grid squares which are intersected by lines in the scene. The entire scene encodement can be accurately described in the standard terminology of picture processing in [R-4] as a "quantized digital picture" having both the shaded and non-shaded grid squares as its binary valued "picture elements." That portion of the scene encodement consisting only of the pattern of shaded grid squares will be called a "snapshot" of the scene.

The scene is always considered as being "viewed" through an aperture called the "screen." This screen will have fixed size, shape, and orientation characteristics associated with it for each specific picture processing application. This will provide a standard reference for the specification of any grid to be used in encoding the scenes under consideration.

2.2 Scenes

The concept of a "scene" is formalized as follows:

Definition 2.1

A scene is a two-dimensional pattern composed of a finite, but non-zero number of straight line segments of finite, non-zero length, with any interconnections (points of intersection) between the line segments existing only at their endpoints.

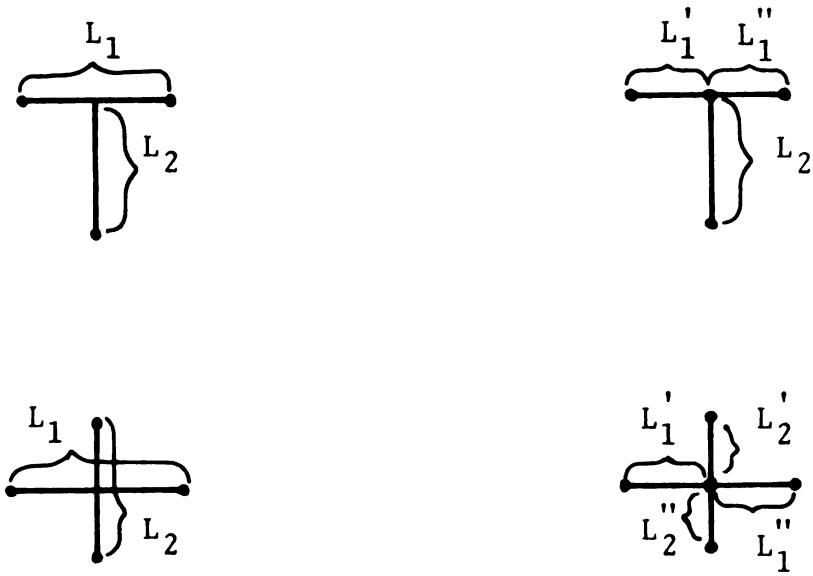
A primitive scene is a scene composed of exactly one straight line segment.

We will denote as the "junction constraint" that part of the preceding definition allowing interconnections between line segments only at their endpoints. The junction constraint is introduced merely for the convenience of standardized scene descriptions (see Chapter 3), and does not restrict the class of scenes under consideration. This is so because a scene containing two straight lines which intersect at a point other than a common endpoint of both lines can alternatively be described as a scene containing this intersection point as the common endpoint of several shorter lines. These shorter lines are formed by dividing both of the intersecting lines at their common point of intersection, with no other alteration of the structure of the scene. With the description of the scene thus

modified, the junction constraint is met without changing the inherent form of the scene in terms of its snapshot generation characteristics. And since we will be examining scenes only through their snapshot representations, we have effectively left the class of scenes under consideration unchanged.

The modification of a scene description to meet the junction constraint is illustrated in Figure 2.1, where two alternative scene descriptions are provided for both the letter "T" and addition operator "+". Both descriptions given for each scene are valid, but only one conforms with the junction constraint in each case.

The straight line segments which comprise a scene will be considered "ideal", i.e., arbitrarily thin. No additional restrictions -- other than the non-zero but finite length, and junction constraints of the definition -- are placed on these line segments at this point, although restricting their slopes to being rational can be useful when scenes will be recognized only in fixed orientations (see [W-2]). In addition to the lines themselves, an important part of scene specification is incorporated in the relationships existing between lines in the pattern -- e.g., whether two lines are connected, perpendicular, parallel, of different lengths, etc. However, it is important to note that only the lines themselves and their relative relationships are considered as part of the scene structure; any areas or regions that might be enclosed or



(a) Descriptions of the letter "T" and addition operator "+" by two intersecting straight lines, L_1 and L_2, which are connected at a point other than a common endpoint of both lines.

(b) Alternate descriptions of the letter "T" by 3 lines intersecting at a common endpoint, and the addition operator "+" by 4 lines intersecting at a common endpoint. The respective descriptions in (a) have been modified so that all interconnections between lines exist only at their endpoints.

Figure 2.1. Examples of scene description modification to meet the "junction constraint."

otherwise implicitly defined by the line segments are not considered part of the scene. In other words, we are considering line drawings, not solid objects or areas.

If we consider a scene as a graph, with the vertices being the collection of all line segment endpoints and the edges being the line segments themselves, then we can make the following definitions for scenes using the terminology of graph theory in [H-1]:

Definition 2.2

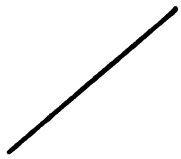
An object is any connected component (maximally connected "subgraph") of a scene.

A simple scene is a scene composed of exactly one object.

A composite scene is a scene containing two or more objects.

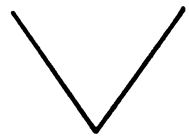
Note that a composite scene is disconnected by definition in a graph theoretic sense.

Some examples of typical scenes are shown in Figure 2.2. Even though the scenes are illustrated in a particular orientation, it is important to bear in mind that scenes are structures defined irrespective of any specific orientation in which they may appear. Note also that only straight line approximations of any nonlinear geometric contours are permitted by the definition of scenes.

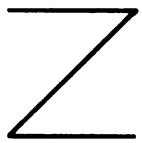


(a) Primitive Scene (also a simple scene, or object)

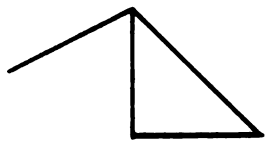
(b) Primitive Scene (also a simple scene, or object)



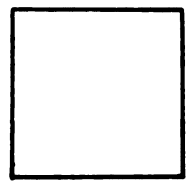
(c) Simple Scene (object)



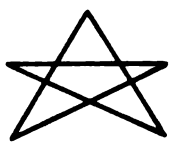
(d) Simple Scene (object)



(e) Simple Scene (object)

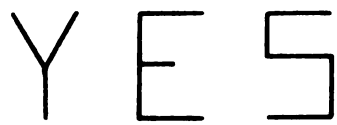


(f) Simple Scene (object)

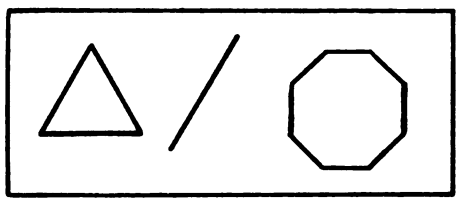


(g) Simple Scene (object)

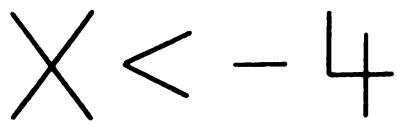
(h) Composite Scene (2 objects)



(i) Composite Scene (3 objects)



(j) Composite Scene (4 objects)



(k) Composite Scene (4 objects)

Figure 2.2. Examples of typical scenes.

2.3 Scene Encodements and Snapshots

In order to encode scenes for processing by a digital computer, or to provide scene encodements which allow the possibility of grammatical description, it is convenient to "discretize" the scene in terms of a finite set of finitely valued scene primitives. The discretizing method chosen here encodes the scene into a finite number of shaded and non-shaded grid squares, with each grid square being a binary valued "primitive" of the scene encodement. The shaded portion of an encodement is called a "snapshot," which is formally defined next along with the construction rules for forming a scene encodement.

Definition 2.3

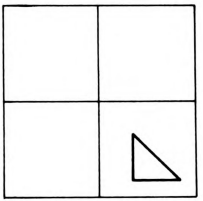
A snapshot of a scene is the entire pattern of zero or more shaded grid squares which results when a finite square grid is superimposed over the scene, and the scene then encoded on the grid according to the following grid square shading scheme:

(1) any grid square having at least one of its sides touched (intersected) by one or more of the line segments in the scene is shaded;

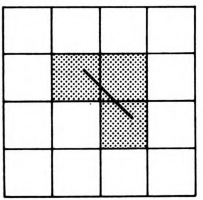
and (2) no other grid squares except those specified in (1) are shaded.

Notice that this definition implies that it is only the structural form of the shaded grid square pattern which defines the snapshot, independent of the relative translational location of the pattern on the grid. Additionally, notice that the empty snapshot, defined as the snapshot resulting from a scene encodement having no shaded grid squares, is a valid entity.

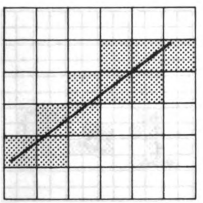
Examples of snapshots for each of six different scenes are shown in Figures 2.3 and 2.4. Two encoding peculiarities arise in these examples which may warrant further explanation. The first is that a line segment of a scene which lies exactly along a vertical or horizontal grid line of a superimposed grid will, when encoded, shade grid squares along its length on both sides of the grid line. The second is that a line segment of a scene which passes through an intersection point of the horizontal and vertical grid lines in a superimposed grid will, when encoded, shade all four grid squares having this grid intersection point as a common corner. Careful examination of the encoding scheme will reveal that the grid shading is being done in accordance with the specified rules, with only grid squares having one or more of their sides touched (intersected) by a line in the scene being shaded in the encodement. This is clear for the first case, and is clarified for the second case by noting that a corner point of a grid square is actually part of two



(a) The empty snapshot of a simple scene, or object (a right, isosceles triangle with leg length of 1/2 grid unit), from a 2 x 2 grid encodement.

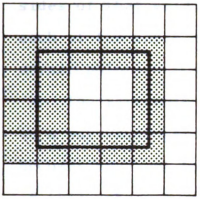


(b) A snapshot of a primitive scene (a straight line segment with length of $\sqrt{2}$ grid units) from a 4 x 4 grid encodement.

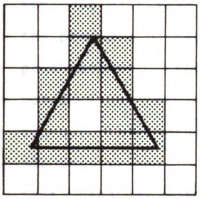


(c) A snapshot of a primitive scene (a straight line segment with length of 6-1/4 grid units) from a 6 x 6 grid encodement.

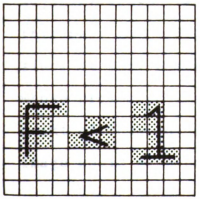
Figure 2.3. Snapshot examples.



(a) A snapshot of a simple scene, or object (a rectangle of dimensions 3 x 3-1/2 grid units), from a 6 x 6 grid encodement.



(b) A snapshot of a simple scene, or object (an equilateral triangle with side length of 4 grid units), from a 6 x 6 grid encodement.



(c) A snapshot of a three object composite scene from a 12 x 12 grid encodement.

Figure 2.4. More snapshot examples.

sides of that grid square (and also part of two sides of each of the three other grid squares sharing the same corner point).

The rationale behind defining the encodement scheme in a way which permits these peculiarities to arise becomes apparent by considering a practical implementation of this scheme. The grid might be physically approximated by a finite, discrete array of contiguous photoelectric sensors, each with sufficient sensitivity for detecting the presence of any portion of an image on any part of its photocathode.[1] Now suppose that the image of a scene composed of non-ideal line segments (i.e., ones with a finite, non-zero thickness) is projected on this array of photoelectric sensors. It is easy to imagine that a line segment whose projected image would theoretically (if it were arbitrarily "thin") lie on the common border between adjacent sensors would realistically have sufficient thickness to be detected by the sensors lying on both sides of that border. Similarly, a "thick" line segment whose projected image passes through the common border point of four adjacent sensors -- and whose ideally thin image would also pass through this same point -- would in

---

[1] The photoelectric sensors are assumed to have square-bordered areas for their photocathodes. Additionally, the physical realization discussed here is referred to as an approximation because the empty snapshot cannot be realized.

reality be detected by all four of those sensors. Note that both of these situations occur in practical applications solely because of the thickness of the line segments, which permits one location along the length of a single line segment to physically overlap the photocathodes of two or more sensors simultaneously. Thus, with grid squares corresponding to sensors, we see that the apparent peculiarities in our theoretical encodement scheme for ideally thin line segments anticipate the extension of the scheme to practical applications involving scenes composed of finitely thick line segments.

2.4 Screens and Grids

It should be evident from the discussion and examples of the previous section that whenever a grid is superimposed over a scene, the resulting encodement is dependent upon each of the following factors: (1) the relative translational position between grid and scene; (2) the relative rotational orientation between grid and scene; and (3) the relative size of the grid squares with respect to the size of the scene. The first factor will be discussed in detail in Chapter 4, where it will become apparent that most scenes produce more than one snapshot when encodements are made with a grid of fixed square size and orientation that is superimposed in different relative locations with respect to the scene. The last two factors affecting scene encodements will be considered in the

remainder of this section, with the result being the formalization of the concepts of grids and their specifications only intuitively referred to up to this point.

In many practical picture processing situations it is realistic to assume that a finite bound exists on the size of all scenes under consideration. This means that some single (but not unique) finitely bounded area can be found whose borders -- in all orientations -- could entirely enclose any one of the scenes to be analyzed. In this case, it will be no loss of generality to further assume that such an area has a square border of fixed, finite size. Such a square bordered area, once specified and fixed in an application, will be called the "screen" for that application.

Definition 2.4

A screen is some fixed, finite, square-bordered two-dimensional area which is specified for a particular picture processing application, and whose border -- in every rotational orientation -- could be translationally positioned to entirely enclose (without contact) each individual scene under consideration.

The single screen thus chosen for a particular application will serve as the fixed size reference for all scenes and grids under consideration in that application. The length of a side of the screen border will be assigned an arbitrary reference value of "1 unit."

The screen can be envisioned as a movable aperture of fixed size, through which any scene to be encoded and processed will be viewed. The screen positioning motions would generally be permitted three degrees of freedom -- two-dimensional translation, and rotation -- but, as noted in Chapter 1, we are restricting this investigation by allowing only translational motions when viewing and encoding a scene. Thus, we will hereafter restrict our attention in any application to a single screen having one fixed, but arbitrarily specified, rotational orientation. The effect of this restriction is that each scene under consideration will be viewed in only one specific rotational orientation at a time. But since scenes are structures having no definite orientation, the specific rotational orientation of any scene viewed through a screen of fixed rotational orientation will be arbitrary. Therefore, if we characterize a "view" of a scene by specifying a fixed, but arbitrary, rotational orientation of the scene relative to the screen through which it is viewed, the particular rotational orientation of that screen is immaterial. Hence, for notational convenience,

we will always assume that any view of a scene is defined relative to a rotationally fixed screen having its border lines oriented horizontally and vertically.

Motivated by the preceding discussion, we now make the following definitions:

Definition 2.5

A standard screen is a screen with a fixed rotational orientation which makes its border lines horizontal and vertical.

Definition 2.6

A view of a scene consists of the scene in some single, arbitrary, rotational orientation defined relative to a standard screen.

The single standard screen thus chosen in an application serves as the size and orientation reference for any view of a scene. Note that each view possible for a scene can be considered by simply specifying the appropriate fixed rotational orientation of that scene defined relative to the standard screen.

With a single standard screen as the size and orientation reference, we can now discuss standardized grid descriptions. The grid structure is formed by using evenly spaced horizontal and vertical lines to

symmetrically divide the entire area enclosed by the standard screen into identically-sized square cells. The square root of the total number of these square cells then defines the "resolution" of the grid. Any grid thus formed can be considered as a "retina" inscribed on the standard screen, with scene views being imposed and encoded on this retina. The ability of a retina to reflect the detail in an image imposed upon it depends on the size, number, and spacing of its sensory elements. Analogously, with grid squares considered as the "sensory elements," the ability of a grid to reflect detail in the encodement of a scene view is characterized by the relative fineness, or "resolution," of that grid.

The concepts of grids and grid specifications are now made more precise.

Definition 2.7

A grid is the symmetrical geometric lattice, defined by the border lines of the standard screen and zero or more additional intersecting horizontal and vertical line segments of length 1 which lie between those border lines such that the standard screen is divided into a number of smaller, identically-sized square cells.

Definition 2.8

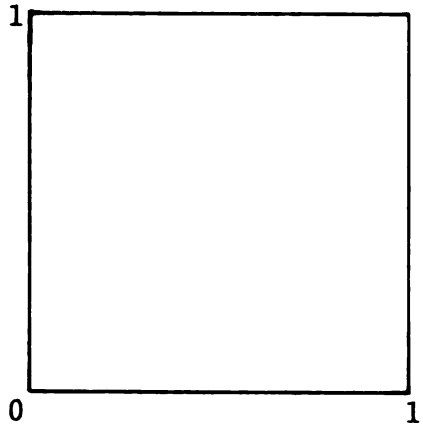
The resolution of a grid is that positive, non-zero integer whose algebraic square represents the total number of identical square cells which the grid defines on the standard screen.

The dimensions of a grid of resolution "r" are denoted by the expression "r x r".

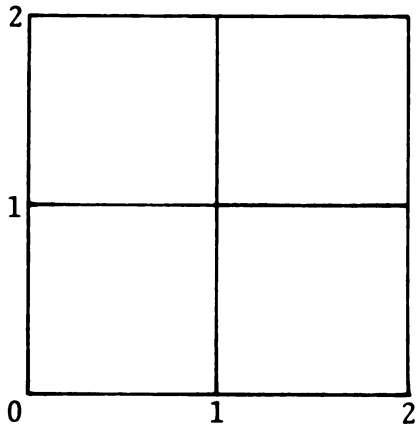
A finite grid is a grid having a finite resolution.

We note from the above definitions that the standard screen is considered as the finite grid of resolution "1".

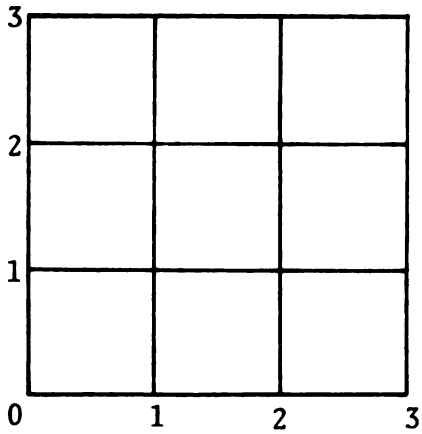
Some examples of finite grids with different resolutions defined relative to the same standard screen are shown in Figure 2.5. Note that all of the grids are of the same absolute size, although each one has a relative Cartesian coordinate system uniquely associated with it in a natural way according to its resolution. The lower horizontal border line of the grid is taken as the positive x-axis, and the left vertical border line is taken as the positive y-axis, with the common intersection point of these axes at the lower left-hand corner of the grid serving as the origin of the natural coordinate system. These relative coordinate systems for grids will be useful in examining the effects of resolution changes on scene descriptions, to be considered in Chapter 3.



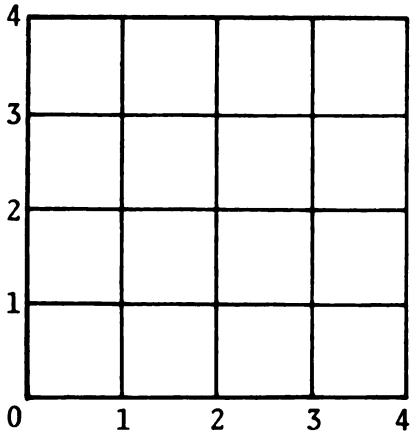
(a) Grid of resolution "1", dimensions "1x1". (The Standard Screen)



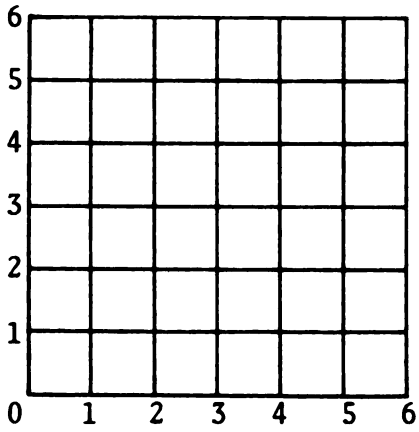
(b) Grid of resolution "2", dimensions "2x2".



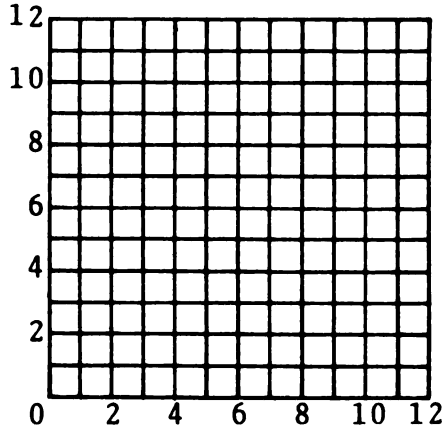
(c) Grid of resolution "3", dimensions "3x3".



(d) Grid of resolution "4", dimensions "4x4".



(e) Grid of resolution "6", dimensions "6x6".



(f) Grid of resolution "12", dimensions "12x12".

Figure 2.5. Examples of finite grids.

Chapter 3

"LINGUISTIC" SNAPSHOT GENERATION

3.1 Description of Translationally-Fixed Scene Views

Since scenes will be examined via their encodements, it is desirable to develop an algorithmic procedure for producing encodements from scene descriptions. This algorithm can be divided into two parts: (1) the generation of views of a scene from a single general scene description, and (2) the generation of encodements for any view of a scene. The first part relates to the derivation of specific descriptions of a scene for arbitrary rotational orientations; the second part deals with the effects of the relative translational positioning between a view of a scene and the grid being used for encoding snapshots. This two-part division thus separates the rotational and translational aspects of the process, permitting the independent study of either one. In accordance with our original objectives, we will confine this investigation to translational considerations by assuming that scene views are our starting point, and then focusing our attention on the view encodement generation process in

part (2). The first concern here is the development of a uniform standard for view descriptions.

The definition of "scene" in Chapter 2 implicitly designates finite length straight line segments as scene "primitives." Note that a scene can be uniquely characterized by specifying its structure in terms of these "primitives" (including their lengths), along with the interconnections (at endpoints), the relative translational locations, and the relative rotational orientations existing between them. Suppose such a scene description is constrained by fixing both the relative rotational orientations and the relative translational locations of the scene primitives with respect to the designated standard screen, without alteration of the scene structure. In this case, the primitives themselves and the relative relationships -- i.e., interconnections, orientations, and translational positions -- existing between them remain unchanged, and the resulting description specifies a translationally-fixed view of the scene.

Recall the elementary fact that a straight line segment having both its rotational orientation and translational position fixed can be completely characterized by the coordinates of its two endpoints specified relative to a fixed two-dimensional coordinate system. Thus it becomes apparent that a translationally-fixed [hereafter denoted as "T-F"] view of a scene can be completely characterized by representing each of its "primitives"

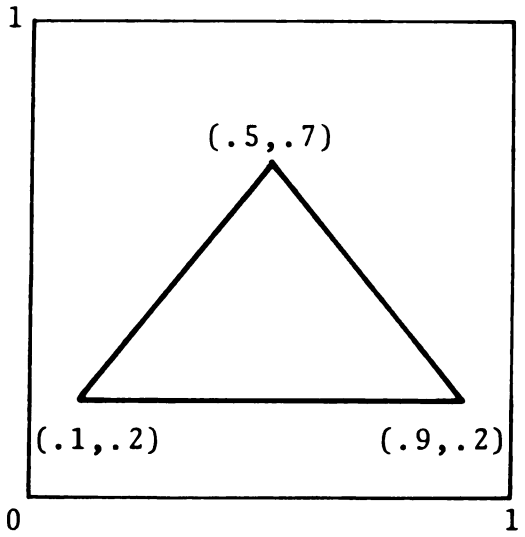
by their endpoint coordinates specified relative to the standard screen. The notational representation for the T-F scene view will be a finite, non-empty set whose elements are sets of two ordered pairs -- each ordered pair in a set representing the Cartesian coordinates of a point on the standard screen, and each set of two ordered pairs representing the two endpoints of one of the primitives of the T-F view. An example of a T-F view of a scene is shown in Figure 3.1 - (a), with its corresponding notational representation in terms of a set of coordinate pairs given in Figure 3.1 - (b).

Notice that any T-F view of a scene has a unique representation using this notation, due to the uniqueness of the description of the scene's structure guaranteed by the junction constraint in Definition 2.1. Furthermore, we observe that any finite set consisting of one or more sets of two ordered pairs describes some unique T-F scene view if and only if such a set has the following properties:

(a) both coordinates in each ordered pair have a value between 0 and 1, inclusive;

(b) the two ordered pairs in any one set are not identical;

and (c) the line segments, defined by interpreting each set of ordered pairs as the two endpoints of the straight line segment



(a) A T-F view of a scene illustrated on the standard screen.

{{(.1,.2),(.5,.7)},{(.5,.7),(.9,.2)},{(.9,.2),(.1,.2)}}

(b) The set notational representation of the T-F scene view in (a).

Figure 3.1. A translationally-fixed scene view and its set notational representation.

connecting them, would intersect each other only at common endpoints if at all.

This is clarified by noting that condition (a) insures that the T-F scene view appears on the standard screen, condition (b) guarantees that each primitive represented by its endpoints has non-zero length (the finite length requirement is guaranteed by condition (a)), and condition (c) insures that the junction constraint is met by the scene whose T-F view is being represented.

The uniqueness properties noted in the previous paragraph confirm the utility of the set notational representation for T-F scene views. Unless designated otherwise, we will hereafter assume that any T-F view of a scene is expressed relative to the coordinate system of the standard screen using this set notational representation.

3.2 Resolution Effects on Descriptions

In order to consider encodements with grids of various resolutions, it becomes necessary to investigate the conversion of a T-F scene view description from the coordinate system of the standard screen to the coordinate system of a grid of resolution "r". We begin by re-examining the coordinate systems of grids in greater detail.

Recall that the natural coordinate system of a grid has the lower horizontal grid border as the x-axis, the

left vertical grid border as the y-axis, and the lower left corner of the grid as the origin. Additionally, for a grid of resolution "r", the lower right corner of the grid defines the maximum x-coordinate value as "r" while the upper left corner of the grid defines the maximum y-coordinate value as "r". Further, each axis is divided into r equal divisions by r + 1 points (including both endpoints of the axis) which are consecutively labeled from the origin with the integers "0" through "r". Each integer valued point along the x-axis represents the lower endpoint of one of the r + 1 identical length vertical line segments of the grid (which include the two vertical border line segments of the standard screen), while each integer valued point along the y-axis represents the left endpoint of one of the r + 1 identical length horizontal line segments of the grid (which include the two horizontal border line segments of the standard screen). The horizontal and vertical line segments intersect at $(r + 1)^2$ points, and divide the standard screen into r^2 identical squares. Of course, any point lying along a vertical grid line has an integer valued x-coordinate, while any point lying along a horizontal grid line has an integer valued y-coordinate. Thus, the intersection points of the horizontal and vertical grid lines are characterized by having integer valued x and y coordinates relative to the natural coordinate system of the grid.

Definition 3.1

A lattice point (or intersection point) of a grid is any point defined by the intersections of the horizontal and vertical line segments (including the border lines of the standard screen) defining that grid.

It should be obvious from the preceding discussion that a grid of resolution r has $(r + 1)^2$ lattice points, each one having integer valued x and y coordinates relative to the grid.

It is important to bear in mind here that an increase in grid resolution is obtained not merely by increasing the number of grid squares which are present, but also by shrinking the size of individual squares in the grid. This is due to the fact that all grids under consideration at any one time are defined relative to the same standard screen. Thus, every grid coordinate system has the same origin and the same x and y axes. The only distinction between the coordinate systems of grids of different resolution is the relative coordinate labeling along the x and y axes. So, given a fixed point on the standard screen being used, each grid imposed on the standard screen would merely relabel the coordinates of the point relative to the natural coordinate system of that grid. In particular, the coordinates of such a fixed point relative to the grid of resolution "1" (which is the

standard screen) can be multiplied by the factor r' to obtain the coordinates of that same fixed point relative to the grid of resolution r'. This simple coordinate transformation is known as a "dilation", or "stretching", by r' relative to coordinate system of the standard screen [L-1], and forms the basis for the following Lemma:

Lemma 3.1

A fixed point on the standard screen having coordinates (a,b) relative to the grid of resolution "1" will have coordinates (r'a,r'b) relative to the grid of resolution r'.

Corollary 3.1

A fixed point on the standard screen having coordinates (x,y) relative to the grid of resolution r will have coordinates $(\frac{r'}{r}x, \frac{r'}{r}y)$ relative to the grid of resolution r'.

Proof:

Let the fixed point on the standard screen having coordinates (x,y) relative to the grid of resolution r have coordinates (a,b) relative to the grid of resolution "1".

Then using Lemma 3.1,

$$(x,y) = (ra,rb) = r(a,b),$$

which implies that

$$(a,b) = \frac{1}{r}\,(x,y) = (\frac{x}{r}\ ,\ \frac{y}{r}).$$

Also, the fixed point having coordinates (a,b) relative to the grid of resolution "1" will have coordinates (x',y') relative to the grid of resolution r', determined using Lemma 3.1 as

$$(x',y') = (r'a,r'b) = r'(a,b),$$

which implies that

$$(a,b) = \frac{1}{r'},\ (x',y') = (\frac{x'}{r'}\ ,\ \frac{y'}{r'}).$$

So, we have

$$(a,b) = \frac{1}{r'},\ (x',y') = \frac{1}{r}\,(x,y)$$

which gives

$$(x',y') = \frac{r'}{r}\,(x,y) = (\frac{r'}{r}x\ ,\ \frac{r'}{r}y).$$

Q.E.D.

Recall from Section 3.1 that a T-F scene view can be completely characterized in terms of the coordinates of the endpoints of each scene primitive specified relative to the standard screen. By Lemma 3.1 we can multiply each of these endpoint coordinates by the same factor r to obtain their corresponding specification relative to the grid of resolution r, with the aggregate result providing a description of the same T-F scene view relative to the grid of resolution r. Because this T-F scene view description change is accomplished using a direct similarity

transformation[1] which also preserves the slope of any line[2], it follows that a change in grid resolution when viewing a scene does not affect the orientation of the primitives, the relative positioning among the primitives, or the relative interconnections which exist between the primitives. Only the lengths of the primitives and relative distance measures within the scene view are changed, all being multiplied by the same factor which determines the coordinate relabeling in the grid resolution conversion. We thus have the following theorem:

Theorem 3.2

Given a T-F view of a scene having the following notational representation in terms of a set of coordinate pairs specified relative to the standard screen:

---

[1] Intuitively, a similarity transformation is a mapping of the plane into itself which preserves angle size. A direct similarity transformation is a similarity transformation which preserves the sense of every angle. For a more formal definition and detailed discussion of similarity transformations, including their structure preserving properties, see [G-1].

[2] If the endpoint coordinates of a straight line segment on the standard screen are (a,b) and (c,d), the line segment has slope $\frac{d-b}{c-a}$. By Lemma 3.1, the same line segment on the grid of resolution r will have endpoint coordinates (ra,rb) and (rc,rd) respectively, with its slope given by $\frac{rd-rb}{rc-ra} = \frac{d-b}{c-a}$. Since the resolution r was arbitrary, this shows that a change in grid resolution does not affect the slope of straight lines.

$$S = \left\{ \{(a_1,b_1),(a_2,b_2)\} \;\middle|\; \begin{array}{l}(a_1,b_1) \text{ and } (a_2,b_2) \text{ are the standard screen coordinates of the two endpoints of a scene primitive}\end{array} \right\}.$$

Then the set

$$rS = \left\{ \{(ra_1,rb_1),(ra_2,rb_2)\} \;\middle|\; \begin{array}{l}\{(a_1,b_1),(a_2,b_2)\} \in S \\ \text{and r an integer} \geq 1\end{array} \right\}$$

describes the same T-F scene view at grid resolution "r".

An immediate consequence of this theorem, in light of Corollary 3.1, is the following:

Corollary 3.2

Given a T-F scene view having the following notational representation in terms of a set of coordinate pairs specified relative to the grid of resolution "r":

$$S = \left\{ \{(x_1,y_1),(x_2,y_2)\} \;\middle|\; \begin{array}{l}(x_1,y_1) \text{ and } (x_2,y_2) \text{ are the coordinates on the grid of resolution r of the two endpoints of a scene primitive}\end{array} \right\}.$$

Then the set

$$\frac{r'}{r} S = \left\{ \{(\frac{r'}{r}x_1,\frac{r'}{r}y_1),(\frac{r'}{r}x_2,\frac{r'}{r}y_2)\} \;\middle|\; \begin{array}{l}\{(x_1,y_1),(x_2,y_2)\} \\ \in S \text{ and } r,r' \text{ are} \\ \text{integers} \geq 1\end{array} \right\}$$

describes the same T-F scene view at grid resolution "r'".

In summary then, shapes and orientations in T-F scene views are preserved under grid resolution conversions; only effective sizes and distances -- and hence the detail reflected in any encodement -- are increased or decreased by a corresponding change in the resolution of the grid being used for viewing and encoding a scene.

3.3 Algorithmic Encodement of T-F Scene Views

Before presenting the formal algorithm which can be used to generate grid encodements for T-F scene views, the notation for describing individual grid squares will be discussed. Each grid square will be labeled g(x,y), where x and y are the abscissa and ordinate, respectively, of the geometric center of that grid square. These coordinates are specified relative to the natural coordinate system of whatever resolution grid the grid square is a part. Note that there are r^2 grid squares in a grid of resolution r, each one specified by a unique label of the form g(p-.5, q-.5) where p and q are both integers between 1 and r inclusive.

The encodement algorithm to be presented has the structural form of a formal grammar, and is applied in a manner analogous to the derivation of terminal strings using the productions of a grammar. Making this grammatical comparison more concrete, the algorithm can be considered as having the set of non-terminal symbols $\{S,L,V,H,G_v,G_h\}$, the set of terminal symbols $\{g\}$, the starting type S, and a set of productions given by groups (I) through (VI) of the algorithm specification (see Subsection 3.3.1). Every terminal or non-terminal symbol is followed by a set of parentheses enclosing one or more coordinates separated by commas, each of which -- with one exception -- represents a real value, either explicitly or through some algebraic expression which may involve variables. The one exception to the form of these coordinates is the first coordinate of the starting type S, which is a finite set of pairs of two-dimensional real-valued coordinates representing a portion of the notational description of the scene in terms of the endpoints of each of its primitives. Because of the analogy which exists between the algorithm and formal grammars, and also because of the coordinates associated with each terminal and non-terminal symbol, the algorithm is termed a "coordinate grammar".

One restriction must be observed when the algorithm is applied: as each step of the algorithm is executed (i.e., as each "production" is applied), the explicit

numerical value of each coordinate must be calculated before proceeding to the next step. In other words, after each application of a syntactic "production" of the algorithm, it is necessary to consider the semantics of each coordinate expression which results before proceeding further. The application of the algorithm thus bears a strong resemblance to interpretive processing, and so we make a further refinement of our terminology by referring to the algorithm as an "interpretive coordinate grammar".

The rules, or "productions", of the algorithm will be presented next in Subsection 3.3.1, followed by some comments regarding notation in Subsection 3.3.2. Subsection 3.3.3 contains a brief discussion on the method of the algorithm, while an example in Subsection 3.3.4 illustrates the manner in which the algorithm is applied. A detailed explanation of how the algorithm works, in which we conclude that the algorithm does indeed generate the proper encodements of T-F scene views and nothing more, is the subject of Subsection 3.3.5. This is followed with the consideration in Subsection 3.3.6 of two cases which require special semantic interpretations for the coordinates generated by the algorithm. Subsection 3.3.7 concludes this section by introducing the concept of extended grids -- larger than those previously defined -- which are needed for the proper interpretation of some encodements generated by the algorithm.

3.3.1 The "Linguistic" Algorithm

(0) The completely specified starting type:

$$S\left(\left\{\begin{matrix}\{(a_1,b_1),(c_1,d_1)\},\{(a_2,b_2),(c_2,d_2)\},\\ \ldots,\{(a_p,b_p),(c_p,d_p)\}\end{matrix}\right\},r\right)$$

(I) Rule for changing the specification of each primitive to the encodement resolution "r", and a termination rule:

(a) $S(\{L_1,L_2,\ldots,L_k\},r) \rightarrow S(\{L_1,L_2,\ldots,L_{k-1}\},r) \cup L(r\cdot a_k, r\cdot b_k, r\cdot c_k, r\cdot d_k)$

(b) $S(\emptyset,r) \rightarrow \emptyset$

(II) Rule defining which vertical and horizontal grid lines can be crossed by a primitive:

$$L(a,b,c,d) \rightarrow V(\lceil \min(a,c) \rceil, \lfloor \max(a,c) \rfloor, 0, a, b, \frac{d-b}{c-a}) \cup H(\lceil \min(b,d) \rceil, \lfloor \max(b,d) \rfloor, 0, a, b, \frac{d-b}{c-a})$$

(III) Rules for generating the coordinates of all intersections of a primitive with vertical grid lines:

(a) $V(x_1,x_2,n,x,y,m) \rightarrow V(x_1-1,x_2-1,n+1,x-1,y-1,m)$

(b) $V(0,x_2,n,x,y,m) \rightarrow V(0,x_2-1,n+1,x-1,y-1,m) \cup G_v(n,y-m\cdot x+n)$

(c) $V(0,0,n,x,y,m) \rightarrow G_v(n,y-m.x+n)$

(d) $V(1,0,n,x,y,m) \rightarrow \emptyset$

(IV) Rules for generating the coordinates of all intersections of a primitive with horizontal grid lines:

(a) $H(y_1,y_2,n,x,y,m) \rightarrow H(y_1-1,y_2-1,n+1,x-1,y-1,m)$

(b) $H(0,y_2,n,x,y,m) \rightarrow H(0,y_2-1,n+1,x-1,y-1,m) \cup G_h(x-\frac{1}{m}.y + n,n)$

(c) $H(0,0,n,x,y,m) \rightarrow G_h(x-\frac{1}{m}.y + n,n)$

(d) $H(1,0,n,x,y,m) \rightarrow \emptyset$

(V) Rule for generating the grid squares which are shaded due to a point of intersection of a primitive with a vertical grid line:

$$G_v(n,y) \rightarrow \left\{\begin{matrix} g(n-.5,\lceil y \rceil -.5), g(n-.5,\lfloor y \rfloor +.5), \\ g(n+.5,\lceil y \rceil -.5), g(n+.5,\lfloor y \rfloor +.5) \end{matrix}\right\}$$

(VI) Rule for generating the grid squares which are shaded due to a point of intersection of a primitive with a horizontal grid line:

$$G_h(x,n) \rightarrow \left\{\begin{matrix} g(\lceil x \rceil -.5,n-.5), g(\lfloor x \rfloor +.5,n-.5) \\ g(\lceil x \rceil -.5,n+.5), g(\lfloor x \rfloor +.5,n+.5) \end{matrix}\right\}$$

3.3.2 Remarks on Notation

There are a number of symbols appearing in the algorithm which were not included in the terminal or non-terminal alphabets of the previous grammatical analogy discussion. While a precise formal description of the algorithm as an interpretive coordinate grammar would require that all these symbols be included in the specification of its alphabet, the more intuitive grammatical analogy notions already introduced will be sufficient for understanding the structure and application of the algorithm. Thus, in the interest of keeping notational complexity to a minimum, the algorithm will not be formalized further in terms of its grammatical properties. Instead, the symbols in the algorithm not specified as part of the alphabet in the grammatical analogy will be described informally as to their interpretation rather than as abstract symbols.

First, several delineators are apparent in the algorithm: left-right bracket pairs, "{ }", used to enclose the elements of a set; left-right parentheses pairs, "( )", used to enclose coordinates; and commas, ",", used to separate coordinates and the elements of a set. Additionally, the empty set "$\emptyset$" appears as the null symbol of the algorithm, performing a function comparable to that of the empty string in conventional grammars.

Next, the symbols "p" and "k" are integer variables representing positive, non-zero, integer values defining

subscript limits. We then have the symbols "a_i", "b_i", "c_i", and "d_i" -- for i = 1, 2,...,p -- as algebraic variables acting as placeholders for specific numerical values. The symbols "L_i" -- for i = 1, 2,...,p -- represent the set $\{(a_i,b_i),(c_i,d_i)\}$.[3] Other algebraic variables acting as placeholders for specific numerical values are "a", "b", "c", "d", "r", "m", "n", "x_1", "x_2", "x", "y", "y_1", and "y_2". The symbols "0", ".5", and "1" are themselves interpreted as specific numerical values.

Further, there are symbols in the algorithm which represent ordinary algebraic operations: "+", "-", ".", and "—" represent the conventional notations for addition, subtraction, multiplication, and division, respectively. Several algebraic functions are also involved: the symbol "min" represents the function which has a value equal to the least numerical value of its arguments; the symbol "max" represents the function which has a value equal to the greatest numerical value of its arguments; the notation "$\lfloor\ \rfloor$" denotes the "entier" function, whose value is the greatest integer less than or equal to the numerical value of the enclosed argument; and the notation "$\lceil\ \rceil$" denotes the least integer greater than or equal to the numerical value of the enclosed argument.

---

[3] All possible symbols "L_i" are covered with the value of "i" ranging from 1 to p instead of 1 to k, because the value of k will never exceed the value of p in this algorithm.

Finally, the set theoretic operation of set union is denoted by the conventional symbol "$\cup$" in the algorithm.

It should be emphasized here that the rewriting symbol "$\rightarrow$" is part of the describing language for the algorithm interpreted as a grammar, and thus does not comprise any part of the strings produced by the algorithm. Rather, the symbol "$\rightarrow$" gives a structure to the rules of the algorithm analogous to the structure of the productions of a grammar, and thus suggests a grammatical interpretation of the algorithm as a set of rewriting rules to be used for deriving strings of terminal symbols. The rewriting symbol "$\rightarrow$" thus has its conventional grammatical interpretation within the rules of the algorithm, which is: a string of symbols having the form specified to the left of the arrow can be replaced by (or "rewritten" as) the string of the form which appears to the right of the arrow.

3.3.3 Method of the Algorithm: A Synopsis

The algorithm is designed to generate binary valued (shaded or non-shaded) grid encodements of T-F views of scenes by starting with both a specification of a T-F scene view relative to the standard screen, and a specification of the grid resolution r at which the encodement is to be produced. With such specifications as the coordinates of the starting type S, the rewriting rules in groups (I) through (VI) of the algorithm are

applied like the productions of a formal grammar -- with the coordinate values numerically interpreted at each step -- to produce a set of terminal symbols representing exactly those grid squares which are shaded in the encodement at grid resolution r. Thus, the result of a "derivation" using this algorithm is a set

$$\{g(x_1,y_1),g(x_2,y_2),\ldots,g(x_q,y_q)\}$$

which designates q grid squares $g(x_i,y_i)$, for i=1,2,...,q, as those which are shaded in the encodement at grid resolution r. All other grid squares are not shaded in the encodement. Note that the number "q" of shaded grid squares in the encodement will be zero in the case of the empty snapshot represented by the empty set of terminals, "$\emptyset$".

The algorithm is based on the fact that a grid encodement of a T-F scene view is uniquely determined by the intersection points of all the primitives with the horizontal and vertical grid lines. This follows directly from the fact that all intersection points of primitives with grid lines reflect all touching points of the scene primitives with the sides of grid squares, and that the touching of the side of a grid square by one or more scene primitives is the only encodement criterion for shading that grid square. We thus have the following interpretation of the encodement scheme in terms of the intersection points of scene primitives with grid lines:

(1) if an intersection point is a grid lattice point, all four grid squares sharing the intersection point at a common corner are shaded;

(2) if an intersection point lies on a vertical grid line other than at a lattice point, the two grid squares -- one to the left, and one to the right, of the vertical grid line -- which share the intersection point on a common vertical side are shaded;

and (3) if an intersection point lies on a horizontal grid line other than at a lattice point, the two grid squares -- one above, and one below, the horizontal grid line -- which share the intersection point on a common horizontal side are shaded.

This interpretation of the encodement scheme forms the procedural basis for generating shaded grid squares in the algorithm.

In generating the intersection points between primitives and grid lines, upon which the encodement of a T-F view of a scene is based, the algorithm determines the intersection points separately for each primitive. The algorithm further subdivides this generation by determining each primitive's intersections with horizontal grid lines (if any) separately from it's intersections with vertical grid lines (if any). In determining the

vertical grid line intersection points for a primitive, the algorithm effectively substitutes the x-value along each vertical grid line crossed by the primitive into the straight line equation of that primitive to determine the corresponding y-coordinate of the intersection point. Similarly, in determining the horizontal grid line intersection points for a primitive, the algorithm effectively substitutes the y-value of each horizontal grid line crossed by the primitive into the straight line equation of that primitive to determine the corresponding x-coordinate of the intersection point. The term "effectively" is used in both cases to denote that the intersection point coordinates are determined relative to successively translated coordinate systems. These translations convert the vertical grid line intersections of a primitive into y-axis intercepts, and convert the horizontal grid line intersections of a primitive into x-axis intercepts. Additional detail will be provided on these intersection determinations using coordinate translations when we examine the interpretation of each rule in the algorithm in Subsection 3.3.5.

The general method upon which the algorithm is based is summarized in the flow charts of Figures 3.2 and 3.3. Notice that the flow chart in Figure 3.3 details the logic of block (E) of the flow chart in Figure 3.2. The other letter labels which appear with the various blocks

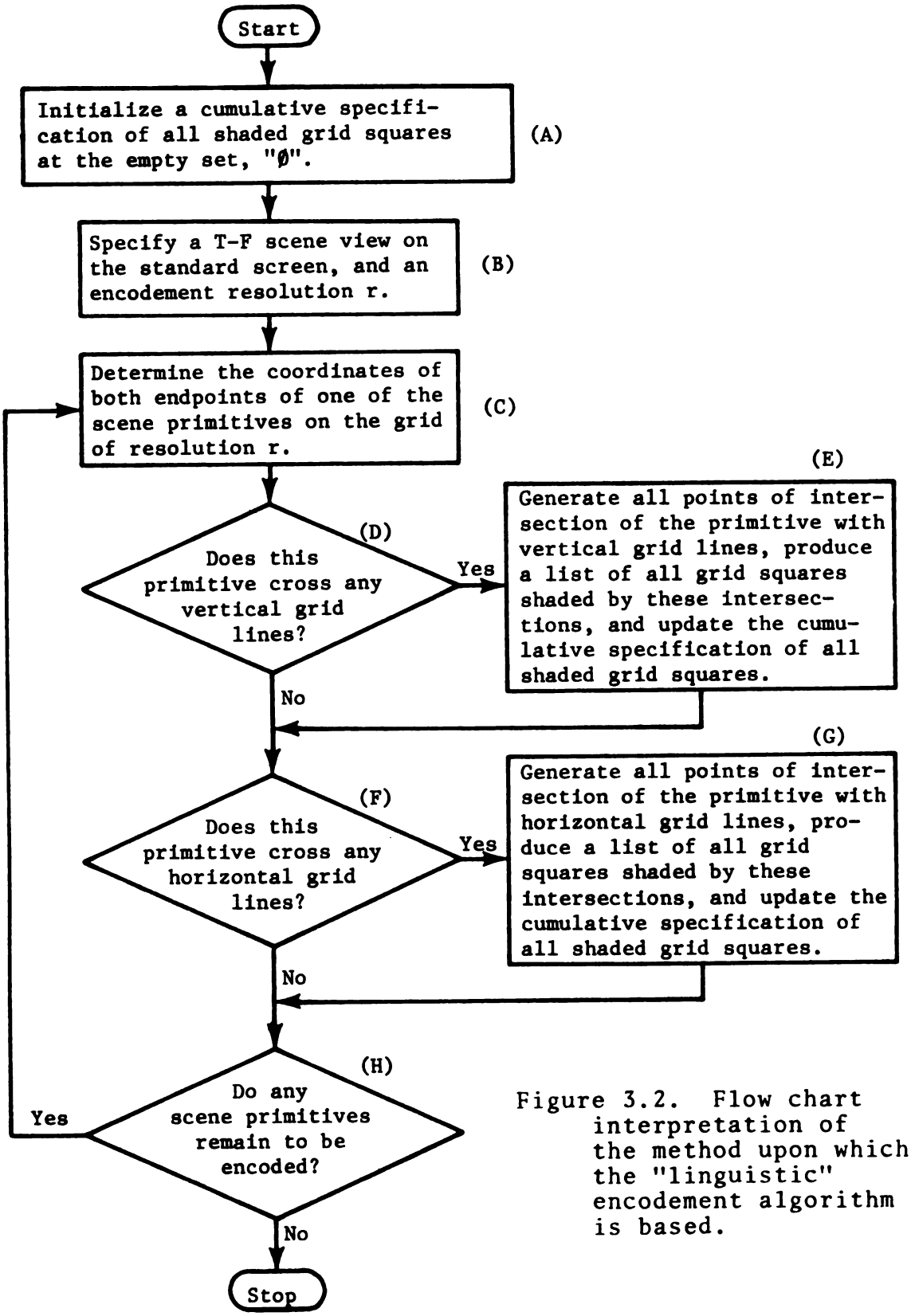


Figure 3.2. Flow chart interpretation of the method upon which the "linguistic" encodement algorithm is based.

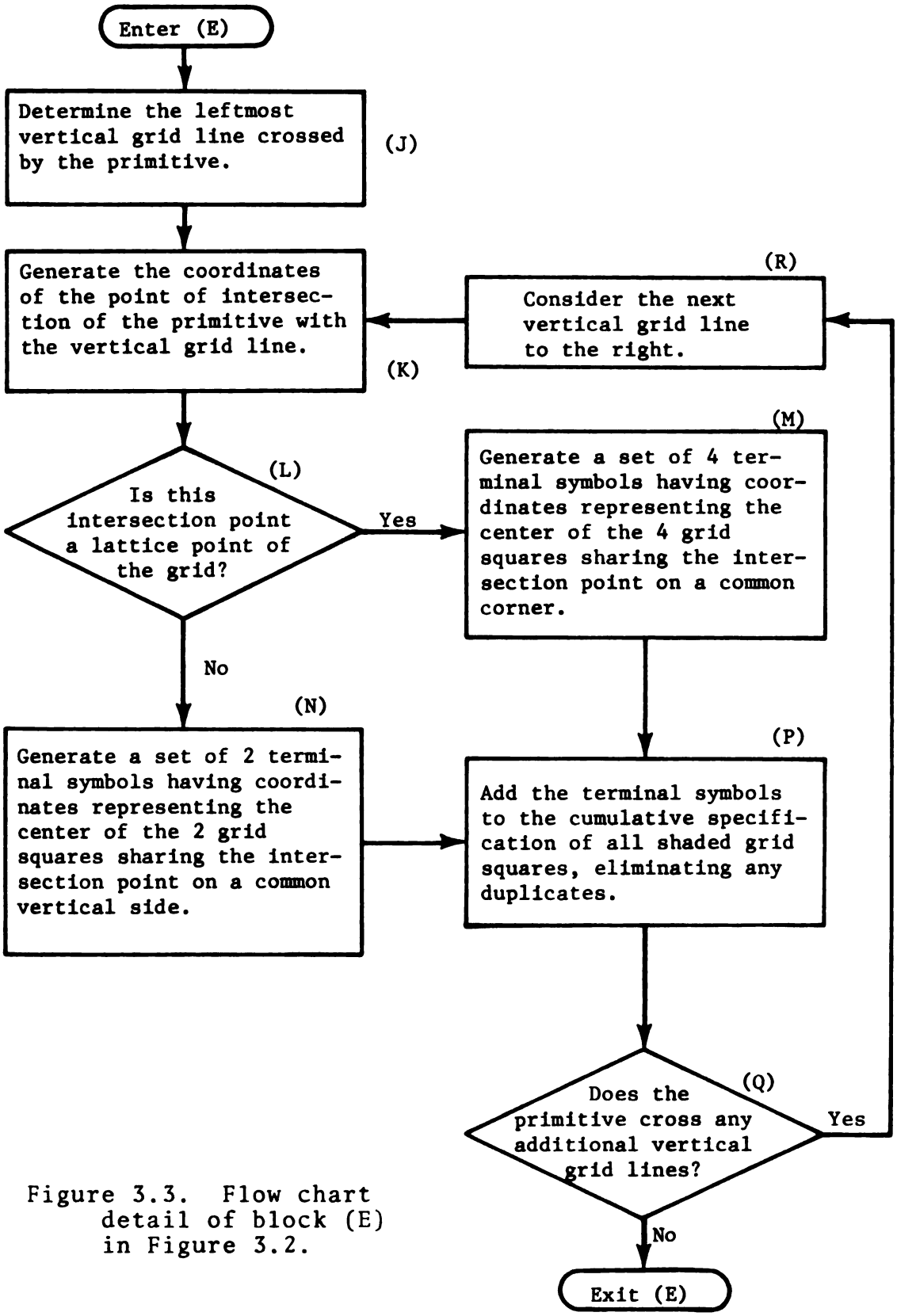


Figure 3.3. Flow chart detail of block (E) in Figure 3.2.

will be used for reference in Subsection 3.3.5, and are of no consequence here.

3.3.4 An example

To illustrate the application of the algorithm, the T-F scene view in Figure 3.4 will be encoded on the grid of resolution "4". The completely specified starting type (0) for the algorithm is

$$S(\{L_1,L_2,L_3,L_4\},4),$$

where L_1, L_2, L_3, and L_4 are as given in Figure 3.4 - (b). The rules of the algorithm can now be applied to this starting type to ultimately produce the proper encodement of the T-F scene view illustrated in Figure 3.5. We proceed as follows:

The completely specified starting type is:

$$S(\{L_1,L_2,L_3,L_4\},4).$$

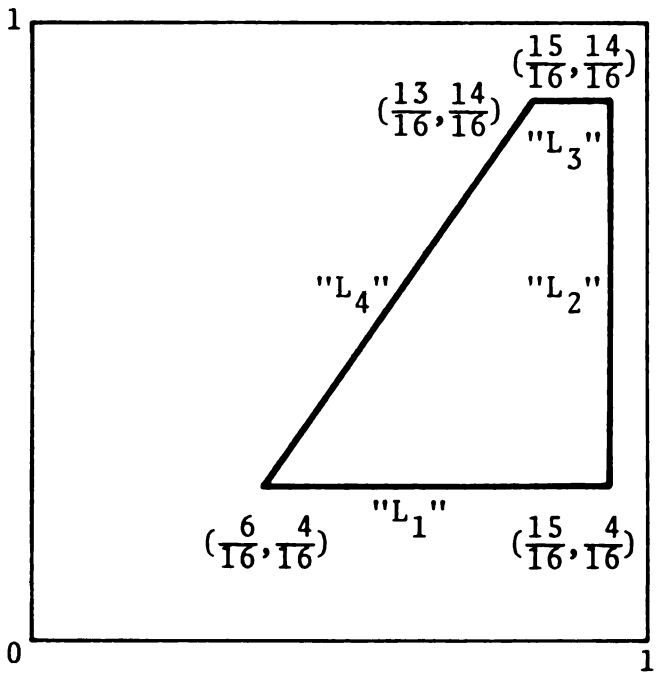
To encode the primitive represented by "L_4" (the "slanted" line segment) on the grid of resolution 4, rules (I) - (a) and (II) must be applied.

Applying rule (I) - (a) yields:

$$S(\{L_1,L_2,L_3\},4) \cup L(\tfrac{13}{16}\cdot 4,\tfrac{14}{16}\cdot 4,\tfrac{6}{16}\cdot 4,\tfrac{4}{16}\cdot 4).$$

Interpreting the coordinates yields:

$$S(\{L_1,L_2,L_3\},4) \cup L(3\tfrac{1}{4},3\tfrac{1}{2},1\tfrac{1}{2},1).$$



(a) A T-F scene view on the standard screen.

$\{L_1, L_2, L_3, L_4\}$

where $L_1 = \{(\frac{6}{16},\frac{4}{16}),(\frac{15}{16},\frac{4}{16})\}$

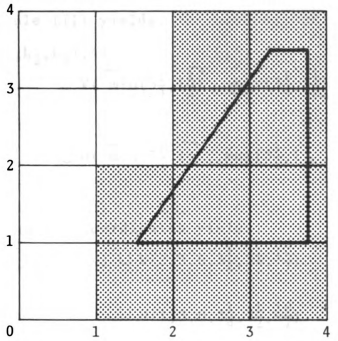
$L_2 = \{(\frac{15}{16},\frac{4}{16}),(\frac{15}{16},\frac{14}{16})\}$

$L_3 = \{(\frac{15}{16},\frac{14}{16}),(\frac{13}{16},\frac{14}{16})\}$

$L_4 = \{(\frac{13}{16},\frac{14}{16}),(\frac{6}{16},\frac{4}{16})\}$

(b) The set notational representation of the T-F scene view in (a).

Figure 3.4. T-F scene view for algorithmic encodement example.



(a) Encodement of the T-F scene view of Figure 3.4 on the grid of resolution 4.

$$\left\{\begin{matrix} g(1.5,.5),g(2.5,.5),g(3.5,.5),g(1.5,1.5),g(2.5,1.5), \\ g(3.5,1.5),g(2.5,2.5),g(3.5,2.5),g(2.5,3.5),g(3.5,3.5) \end{matrix}\right\}$$

(b) Set of terminals denoting the grid encodement in (a).

Figure 3.5. Encoded result of T-F scene view of Figure 3.4 on the grid of resolution 4.

Applying rule (II) yields:

$$S(\{L_1,L_2,L_3\},4) \cup V(\lceil \min(3\tfrac{1}{4},1\tfrac{1}{2}) \rceil, \lfloor \max(3\tfrac{1}{4},1\tfrac{1}{2}) \rfloor, 0, 3\tfrac{1}{4}, 3\tfrac{1}{2}, \frac{1-3\frac{1}{2}}{1\frac{1}{2}-3\frac{1}{4}}) \cup H(\lceil \min(3\tfrac{1}{2},1) \rceil, \lfloor \max(3\tfrac{1}{2},1) \rfloor, 0, 3\tfrac{1}{4}, 3\tfrac{1}{2}, \frac{1-3\frac{1}{2}}{1\frac{1}{2}-3\frac{1}{4}}).$$

Interpreting the coordinates yields:

$$S(\{L_1,L_2,L_3\},4) \cup V(2,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}) \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

To determine all grid squares shaded by primitive "L_4" due to its intersections with vertical grid lines, all occurrences of the symbol "V" are to be removed by applications of rules in group (III), with all symbols "G_v" which may result also to be removed by applications of rule (V).

Applying rule (III) - (a) and interpreting coordinates yields:

$$S(\{L_1,L_2,L_3\},4) \cup V(1,2,1,2\tfrac{1}{4},2\tfrac{1}{2},\tfrac{10}{7}) \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Again applying rule (III) - (a) and interpreting coordinates yields:

$$S(\{L_1,L_2,L_3\},4) \cup V(0,1,2,1\tfrac{1}{4},1\tfrac{1}{2},\tfrac{10}{7}) \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Applying rule (III) - (b) and interpreting coordinates:

$$S(\{L_1,L_2,L_3\},4) \cup V(0,0,3,\tfrac{1}{4},\tfrac{1}{2},\tfrac{10}{7}) \cup G_v(2,1\tfrac{5}{7}) \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Applying rule (V) and interpreting coordinates:

$$S(\{L_1,L_2,L_3\},4) \cup V(0,0,3,\tfrac{1}{4},\tfrac{1}{2},\tfrac{10}{7}) \cup \begin{Bmatrix} g(1.5,1.5), g(1.5,1.5), \\ g(2.5,1.5), g(2.5,1.5) \end{Bmatrix} \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Eliminating duplicate set elements:

$$S(\{L_1,L_2,L_3\},4) \cup V(0,0,3,\tfrac{1}{4},\tfrac{1}{2},\tfrac{10}{7}) \cup \{g(1.5,1.5),g(2.5,1.5)\} \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Applying rule (III) - (c) and interpreting coordinates:

$$S(\{L_1,L_2,L_3\},4) \cup G_v(3,3\tfrac{1}{7}) \cup \{g(1.5,1.5),g(2.5,1.5)\} \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Applying rule (V), interpreting coordinates, and eliminating duplicate set elements:

$$S(\{L_1,L_2,L_3\},4) \cup \{g(2.5,3.5),g(3.5,3.5)\} \cup \{g(1.5,1.5),g(2.5,1.5)\} \cup H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Combining sets of terminals with the set union operator:

$$S(\{L_1,L_2,L_3\},4) \cup \begin{Bmatrix} g(1.5,1.5), g(2.5,1.5), \\ g(2.5,3.5), g(3.5,3.5) \end{Bmatrix}$$

$$\cup \; H(1,3,0,3\tfrac{1}{4},3\tfrac{1}{2},\tfrac{10}{7}).$$

Next we must determine all grid squares shaded by primitive "L_4" due to its intersections with horizontal grid lines. This is done by removing all occurrences of the symbol "H" by applications of the rules in group (IV), and also by removing any resulting symbols "G_h" by applications of rule (VI).

Applying rule (IV) - (a) and interpreting coordinates:

$$S(\{L_1,L_2,L_3\},4) \cup \begin{Bmatrix} g(1.5,1.5), g(2.5,1.5), \\ g(2.5,3.5), g(3.5,3.5) \end{Bmatrix}$$

$$\cup \; H(0,2,1,2\tfrac{1}{4},2\tfrac{1}{2},\tfrac{10}{7}).$$

Applying rule (IV) - (b) twice and then applying rule (IV) - (c), interpreting coordinates after each step, yields:

$$S(\{L_1,L_2,L_3\},4) \cup \begin{Bmatrix} g(1.5,1.5), g(2.5,1.5), \\ g(2.5,3.5), g(3.5,3.5) \end{Bmatrix}$$

$$\cup \; G_h(1\tfrac{1}{2},1) \cup G_h(2\tfrac{1}{5},2) \cup G_h(2\tfrac{9}{10},3).$$

Applying rule (VI) three times, interpreting coordinates, and eliminating duplicate elements within each set, yields:

$$S(\{L_1,L_2,L_3\},4) \cup \left\{\begin{matrix} g(1.5,1.5), g(2.5,1.5), \\ g(2.5,3.5), g(3.5,3.5) \end{matrix}\right\}$$
$$\cup \{g(1.5,.5), g(1.5,1.5)\}$$
$$\cup \{g(2.5,1.5), g(2.5,2.5)\}$$
$$\cup \{g(2.5,2.5), g(2.5,3.5)\}.$$

Combining sets of terminals with the set union operators:

$$S(\{L_1,L_2,L_3\},4) \cup \left\{\begin{matrix} g(1.5,.5), g(1.5,1.5), g(2.5,1.5), \\ g(2.5,2.5), g(2.5,3.5), g(3.5,3.5) \end{matrix}\right\}.$$

At this point, the algorithm has produced the complete encodement of the "slanted" line segment primitive "L_4". The encodement of each of the remaining three line segment primitives -- represented by "L_1","L_2", and "L_3" -- can be produced in a similar manner.

The primitive represented by "L_3" (the short horizontal line segment at the top) is encoded by the sequential application of rule (I) - (a), rule (II), rule (III) - (a) three times, rule (III) - (d), rule (IV) - (a) three times, and then rule (IV) - (d), in that respective order, to the string produced thus far. Coordinates are interpreted at each step, and the set union operator is applied to unite the two sets (both empty) of terminals generated for the encodement of "L_3". The result is:

$$S(\{L_1,L_2\},4) \cup \emptyset$$
$$\cup \left\{\begin{matrix} g(1.5,.5), g(1.5,1.5), g(2.5,1.5), \\ g(2.5,2.5), g(2.5,3.5), g(3.5,3.5) \end{matrix}\right\}.$$

Note that the encodement of primitive "L_3" is the empty set, since "L_3" does not cross any grid lines at resolution "4".

The primitive represented by "L_2" (the vertical line segment on the right) is encoded by the sequential application of rule (I) - (a), rule (II), rule (III) - (a) three times, rule (III) - (d), rule (IV) - (a), rule (IV) - (b) twice, rule (IV) - (c), and then rule (VI) three times, in that respective order, to the string produced thus far. Coordinates are interpreted at each step, and the set union operators are applied to unite the four sets (one of which is the empty set) of terminals generated for the encodement of "L_2". The result is:

$$S(\{L_1\},4) \cup \left\{\begin{matrix} g(3.5,.5), g(3.5,1.5), \\ g(3.5,2.5), g(3.5,3.5) \end{matrix}\right\}$$

$$\cup \emptyset$$

$$\cup \left\{\begin{matrix} g(1.5,.5), g(1.5,1.5), g(2.5,1.5), \\ g(2.5,2.5), g(2.5,3.5), g(3.5,3.5) \end{matrix}\right\} .$$

Note that the entire encodement of primitive "L_2" resulted from its intersections with horizontal grid lines, as it lies entirely between two consecutive vertical grid lines at resolution "4".

The primitive represented by "L_1" (the horizontal line segment on the bottom) is encoded by the sequential

application of rule (I) - (a), rule (II), rule (III) - (a) twice, rule (III) - (b), rule (III) - (c), rule (V) twice, rule (IV) - (a), rule (IV) - (c)[4], and then rule (VI), in that respective order, to the string produced thus far. Coordinates are interpreted at each step, and the set union operators are applied to unite the three sets of terminals generated for the encodement of "L_1". The result is:

$$S(\emptyset,4) \cup \left\{\begin{matrix} g(1.5,.5), g(1.5,1.5), g(2.5,.5), \\ g(2.5,1.5), g(3.5,.5), g(3.5,1.5) \end{matrix}\right\}$$

$$\cup \left\{\begin{matrix} g(3.5,.5), g(3.5,1.5), \\ g(3.5,2.5), g(3.5,3.5) \end{matrix}\right\} \cup \emptyset$$

$$\cup \left\{\begin{matrix} g(1.5,.5), g(1.5,1.5), g(2.5,1.5), \\ g(2.5,2.5), g(2.5,3.5), g(3.5,3.5) \end{matrix}\right\} .$$

At this point, the encodement for each of the four primitives in the T-F scene view is complete. The first coordinate of "S" has become the empty set, signifying that no more primitives remain to be encoded. Applying rule

---

[4] The x-coordinate of the "G_h" symbol generated by the application of rule (IV) - (c) includes the expression "$\frac{1}{m}.y$", which in this case takes the mathematically undefined form "$\frac{1}{0}.0$". This expression is to be interpreted as having the value "zero", in order to insure that proper encodements are produced. This is explained in detail in Subsection 3.3.6. (Also see Footnote 6 of this Chapter).

(I) - (b) to the string generated thus far, and combining all the sets of terminals with the set union operators, terminates the algorithm with the following result:

$$\{g(1.5,.5),g(1.5,1.5),g(2.5,.5),g(2.5,1.5),g(2.5,2.5),$$
$$g(2.5,3.5),g(3.5,.5),g(3.5,1.5),g(3.5,2.5),g(3.5,3.5)\}.$$

This set of terminals represents the complete encodement -- as shown in Figure 3.5 -- of the T-F scene view of Figure 3.4 on the grid of resolution "4".

This example has demonstrated the application of the algorithm in producing the encodement of a specific T-F scene view. While this has illustrated the manner in which the rules can be applied, it remains to be shown that the algorithm works in general. Hence, the next topic to be considered will be the verification of the validity of the algorithm in the general case.

3.3.5 Method of the Algorithm: A Detailed Explanation and Verification

Through a discussion emphasizing the semantics associated with the rules of the algorithm, this subsection details the structure, application, and interpretation of the rules of the algorithm. In addition to explaining the method and logic of the algorithm, the goal is to justify that the structure of the algorithm is such that the proper encodement of T-F scene views is assured.

Recall the grammatical structure of the algorithm with rules which are applied like the productions of a grammar in deriving a terminal string. As is the case with productions in a grammar, the order of application of the rules of the algorithm in deriving sets of terminals is not necessarily unique. However, a unique order of application for the rules of the algorithm will be used in this discussion in order to make the logic of the algorithm more understandable. This unique order appears in the flow charts of Figures 3.2 and 3.3, which represent the method used by the algorithm in generating encodements. Bear in mind that the flow chart in Figure 3.2 details the logic of block (E) of the flow chart in Figure 3.1. Both of these flow charts will be referenced frequently as we associate this logical structure with the rules of the algorithm.

The entity labeled "(0)" in the algorithm represents the complete specification of the starting type for the algorithm. More specifically, the starting symbol "S", chosen to be suggestive of the word "scene", is followed by two coordinates. The first coordinate of "S" specifies a T-F view of a scene in terms of its set notational description. The scene has "p" primitives ($p \geq 1$), with each primitive represented by its two endpoint coordinates specified relative to the standard screen. The second coordinate of "S" is a finite, non-zero, positive integer "r" which specifies the grid resolution at which

the T-F scene view described by the first coordinate is to be encoded. Once the starting type has been completely specified with its two coordinates -- corresponding to block (B) of Figure 3.2 -- the rewriting rules in groups (I) through (VI) of the algorithm can be applied to produce the complete encodement specification.

A single application of rule (I) - (a) corresponds to block (C) of the flow chart in Figure 3.2. One of the "p" primitives "L_i" = $\{(a_i,b_i),(c_i,d_i)\}$ is removed from the first coordinate of the starting type "S" and converted to its corresponding representation on the grid of resolution "r" as "$L(r\cdot a_i,r.b_i,r\cdot c_i,r\cdot d_i)$". The symbol "L" is chosen to be representative of the term "line segment primitive", with its first two coordinates representing the abscissa and ordinate, respectively, of the other endpoint. One application of rule (I) - (a) thus corresponds to a partial application of Theorem 3.2, using a modified notation for the result of the coordinate transformation.

Rule (II) of the algorithm is preparatory to asking the questions posed in blocks (D) and (F) of Figure 3.2. This rule converts the specification of line segment primitive "L" into the specification of symbol "V", which will be used in generating intersection points of the primitive with vertical grid lines, and the specification of symbol "H", which will be used in generating intersection points of the primitive with horizontal grid lines.

Recalling that the coordinates of "L" specify the endpoints of the primitive on the grid of resolution "r", we note the following regarding the coordinates of "V":

(1) the first coordinate of "V" represents the least integer greater than or equal to the smallest x-coordinate of the primitive's endpoints, and thus specifies the leftmost vertical grid line that the primitive can cross;

(2) the second coordinate of "V" represents the greatest integer less than or equal to the largest x-coordinate of the primitive's endpoints, and thus specifies the rightmost vertical grid line that the primitive can cross;

(3) the third coordinate "0" initializes a counter which will keep track of coordinate axes translations as intersection points on grid lines are determined;

and (4) the fourth and fifth coordinates, "a" and "b" respectively, represent the coordinates of one of the endpoints of the primitive, which along with the sixth coordinate -- which represents the slope of the primitive -- provide sufficient information for constructing the "point-slope" equation for the straight line segment primitive.

In a similar manner, the first coordinate of symbol "H" specifies the lowest horizontal grid line that the primitive can cross, the second coordinate of "H" specifies the highest horizontal grid line that the primitive can cross, and the remaining four coordinates of "H" are identical to those of "V".

The rules in Group (III) of the algorithm are used in determining all vertical grid line crossings of the primitive, if any, corresponding to block (D) and a portion of block (E) in Figure 3.2. Each application of rule (III) - (a) corresponds to a translation of the origin of the coordinate system one unit to the right and one unit up (i.e., one unit along the 45^{o} diagonal of the grid). This decreases by "1" the values of all the coordinates of "V" specified relative to the coordinate system, which includes those values in the first, second, fourth, and fifth coordinates. The value of the third coordinate of "V", which represents a counter keeping track of the location of the translated coordinate system relative to the natural coordinate system of the grid of resolution "r", is increased by "1". The last coordinate of "V" representing the slope of the primitive remains unchanged, as the slope of a line is invariant under a translation of coordinates. Rule (III) - (a) is applied consecutively zero or more times until one of the rules (III) - (b), (III) - (c), or (III) - (d) is applicable. Rule (III) - (d) is applicable at this point if and only if both the

smallest and largest x-coordinates on the primitive (which occur exactly at the endpoints of this line segment primitive) are not integers themselves, and both lie strictly between the same two consecutive integer values. This is signified by the "1,0" combination in the first two coordinates of "V", respectively, which occurs here if and only if the leftmost vertical grid line which the primitive can cross (as determined in rule (II)) is to the right of the rightmost vertical grid line which the primitive can cross (also as determined in rule (II)). Thus rule (III) - (d) is applicable only when the primitive does not cross any vertical grid lines, corresponding to the "no" exit of block (D) in Figure 3.2. If rule (III) - (d) was not applicable at this point, this signifies that the "yes" exit from block (D) in Figure 3.2 would be taken. In this case, the "0" appearing as the first coordinate of "V" indicates that the value of the third coordinate of "n" represents the leftmost vertical grid line on the grid of resolution "r" which is intersected by the primitive. Thus, the applications of rule (III) - (a) which led to the "0" in the first coordinate of "V" have already accomplished block (J) of Figure 3.3. Rule (III) - (c) is now applicable if and only if the second coordinate of "V", as well as its first coordinate, has reached a value of "0", signifying that the y-axis of the translated coordinate system coincides with the rightmost vertical grid line intersected by the primitive. In this case, an

application of rule (III) - (c) generates the symbol "G_v" -- chosen to be suggestive of the terminology "intersection point on a vertical grid line" -- with two coordinates specified relative to the natural coordinate system of the grid of resolution "r". These two coordinates represent the abscissa and ordinate, respectively, of the intersection point of the primitive with the rightmost vertical grid line it crosses. However, this intersection point lies on the y-axis of the translated coordinate system, and thus constitutes the y-intercept of the primitive with coordinates (0,y-m·x) relative to the translated coordinate system, where: (1) "x" and "y" as the fourth and fifth coordinates of "V" represent the abscissa and ordinate, respectively, of an endpoint of the primitive relative to the translated coordinate system; and (2) "m" as the sixth coordinate of "V" represents the slope of the primitive.[5] Therefore, adding the translation factor "n" (the third coordinate of "V") to the coordinates of this y-intercept specified relative to the translated coordinate system results in the coordinates "(n,y-m·x+n)," which

---

[5] If "m" = ∞, then the only time it will appear in the coordinates of the y-intercept -- or in the coordinates of "G_v" -- is when the value of "x" has become identically zero. In this case, the expression "m·x" takes the mathematically undefined form "∞·0", which should here be interpreted as the value "0". This is explained in detail in Subsection 3.3.6.

then represent the coordinates of "G_V" specified relative to the natural coordinate system of the grid of resolution "r". The coordinates of "G_V" thus represent the point of intersection of the primitive with the grid line x=n on the grid of resolution "r". The application of rule (III) - (c) corresponds to the final pass through block (K) of Figure 3.3, and also signifies that the "no" exit will be taken from block (Q) of Figure 3.3 the next time it is reached since the non-terminal symbol "V" is eliminated. If rule (III) - (c) is not applicable, rule (III) - (b) should be applied until the second coordinate of "V" does become "0" and rule (III) - (c) can be applied. Rule (III) - (b) is used to determine all intersection points of the primitive with each vertical grid line it crosses except the rightmost one. Each application of rule (III) - (b) performs the operations of both rules (III) - (a) and (III) - (c), with minor modifications, as follows:

(1) the symbol "G_V" is generated with the appropriate coordinates representing the intersection point of the primitive with the vertical grid line x=n (never the rightmost one crossed) by using the method of rule (III) - (c);

and (2) the coordinate system is translated one additional unit along the 45^o diagonal of the grid by regenerating "V" on the right side of "→" with the appropriately modified coordinates as done in rule (III) - (a) -- with the exception

that the first coordinate of "V" is maintained at the value "0" (not further decremented) to signify that the translated y-axis is to the right of the leftmost vertical grid line intersected by the primitive.

The regeneration of "V" with modified coordinates by each application of rule (III) - (b) allows the intersection point of the primitive with the next vertical grid line to the right to be generated. Linking rule (III) - (b) to the flow chart of Figure 3.3, each application of rule (III) - (b) -- because of its operation in (1) above -- corresponds to all but the final pass through block (K) of Figure 3.3. Additionally, because of the operation of regenerating "V" in (2) above, each application of rule (III) - (b) provides for block (R) in Figure 3.3 after insuring that the "yes" exit is taken from block (Q) when it is next reached.

Rule (V) of the algorithm is used to generate the shaded grid squares which result from each intersection point of the primitive with a vertical grid line, and thus corresponds to blocks (L), (M), and (N) in the flow chart of Figure 3.3. Each application of rule (V) converts the coordinates "(n,y)" of "G_V" -- which represent the point of intersection of the primitive with a vertical grid line -- into a set of either four or two specifications of shaded grid squares, depending respectively upon whether or not the coordinates "(n,y)" represent a grid lattice

point. In either case, though, any grid square shaded as a result of this intersection point "(n,y)" will be immediately to the left or right of the vertical grid line x=n, and thus has the x-coordinate of its center as either "n-.5" or "n+.5". And since any intersection point on a vertical grid line causing a grid square on one side of the grid line to be shaded also causes the adjacent grid square at the same vertical level on the opposite side of the grid line to be shaded, all that remains is the determination of all possible values for the y-coordinate of the center of any grid squares shaded as a result of the intersection point "(n,y)." If the value of y is an integer, signifying that the intersection point "(n,y)" is a lattice point, then the y-coordinate of the center of each grid square shaded by this intersection point is either "y+.5" or "y-.5". But since "y" is an integer, $\lfloor y \rfloor = \lceil y \rceil = y$, and so "$\lfloor y \rfloor$+.5" and "$\lceil y \rceil$-.5" represent the two distinct y-coordinates "y+.5" and "y-.5", respectively. By generating all four possible combinations of these two distinct x-coordinates and two-distinct y-coordinates as the x- and y-coordinates, respectively, of the terminal symbol "g", rule (V) thus specifies the set of four grid squares shaded as a result of the lattice point intersection of the primitive with a vertical grid line. On the other hand, if the value of y is not an integer, this signifies that the intersection point "(n,y)" is not a lattice point but a point which lies between two horizontal

grid lines on a vertical grid line. In this case, the y-coordinate of the center of each grid square shaded as a result of the intersection will be located at the single vertical level which is midway between the same two horizontal grid lines that the intersection point "(n,y)" lies between. This level is represented by either "$\lfloor y \rfloor + .5$" or "$\lceil y \rceil - .5$" -- both of which have the same value since "y" is not an integer -- as the y-coordinate of the center of a shaded grid square. So here the four coordinate combinations generated for the terminal symbol "g" by rule (V) results in only two distinct elements in the set, representing the two grid squares shaded as a result of the non-lattice point intersection. The duplicate specifications generated for each of these shaded grid squares -- due to the fact that "$\lfloor y \rfloor + .5$" = "$\lceil y \rceil - .5$" -- are automatically eliminated by the interpretation that rule (V) generates a <u>set</u> of terminals. Therefore rule (V) inherently includes both the "yes" and "no" exits from block (L) in Figure 3.3 -- as signified, respectively, by the value of the y-coordinate of the intersection being an integer or not -- when it is applied, as well as accomplishing the appropriate operation in either block (M) or block (N) of Figure 3.3.

The function of block (P) in Figure 3.3 is automatically accomplished by the set union operators "$\cup$" which have been, and which will be, generated by any application(s) of rules (I) - (a), (II), and/or (III) - (b), as

well as those which will also be generated by any application(s) of rule (IV) - (b). The rules of group (IV), and rule (VI), will now be discussed briefly.

The rules in group (IV) of the algorithm are used in determining all horizontal grid line intersections of the primitive, if any, corresponding to block (F) and a portion of block (G) in Figure 3.2. Each of the rules (IV) - (a), (IV) - (b), (IV) - (c), and (IV) - (d) is completely analogous in both structure and application to the corresponding rule in group (III) -- rule (III) - (a), (III) - (b), (III) - (c), and (III) - (d), respectively. It is however necessary to bear in mind that the rules in group (IV) are interpreted with respect to horizontal grid lines, while the rules in group (III) are interpreted with respect to vertical grid lines. This means that the concepts of "horizontal," "below," "lowest," "above," "highest," "x-intercept," and the like are used when interpreting the rules in group (IV) in places analogous to those where the concepts of "vertical," "left of," "leftmost," "right of," "rightmost," and "y-intercept," respectively, were used in interpreting the rules in group (III). In particular, note that the symbol "G_h" is suggestive of the terminology "intersection point on a horizontal grid line," and that the coordinates generated for "G_h" are based on the x-intercept $(x-\frac{1}{m}\cdot y, 0)$ of the

primitive relative to a coordinate system translated "n" units along the 45^{o} diagonal of the grid of resolution "r".[6]

The analogy drawn between the rules in groups (III) and (IV) of the algorithm is continued with a similar analogy existing between rules (V) and (VI). Rule (VI) of the algorithm is used to generate the shaded grid squares which result from each intersection point of the primitive with a horizontal grid line. Each application of rule (VI) converts the coordinates "(x,n)" of "G_h" -- which represent the point of intersection of the primitive with a horizontal grid line -- into a set of either four or two specifications of shaded grid squares, depending respectively upon whether or not the coordinates "(x,n)" represent a grid lattice point. The shaded grid squares generated will be symmetrically positioned above and below the horizontal grid line on which the intersection point lies. The coordinates of their centers will be determined using the method introduced in rule (V), with the role of "y" in "$G_v(n,y)$" being assumed by "x" in "$G_h(x,n)$", while the role of "n" remains the same except for its

---

[6] If "m" = 0, then the only time it will appear in the coordinates of the x-intercept -- or in the coordinates of "G_h" -- is when the value of "y" has become identically zero. In this case, the expression "$\frac{1}{m} \cdot y$" takes the mathematically undefined form "$\frac{1}{0} \cdot 0$," which should here be interpreted as the value "0". This is explained in detail in Subsection 3.3.6.

appearance in rule (VI) in a different coordinate position than in rule (V).

It should now be apparent that a flow chart, completely analogous to the one shown in Figure 3.3 for the application(s) of the rules in groups (III) and (V) of the algorithm, could be drawn to depict the application(s) of the rules in groups (IV) and (VI) of the algorithm. Because this flow chart would detail block (G) of Figure 3.2 in the same way that Figure 3.3 details block (E), it has been omitted.

At this point we have described the interpretation for each rule, and a sequence of application(s) for these rules, in generating the encodement of one primitive in a T-F scene view on the grid of resolution "r". The encodement of each of the remaining primitives is accomplished by returning to rule (I) - (a) to initiate the encodement of the next primitive, and then by again repeating all applications of the rules from that point on as already discussed for the first primitive encoded. Note that rule (I) - (a) will be applicable at the beginning of any such repetition if and only if the first coordinate of "S" is non-empty, signifying the "yes" exit from block (H) of Figure 3.2. Thus the entire rule application process from rule (I) - (a) on is repeated a total of "p" times (including the initial time) to encode all the primitives of a T-F scene view having "p" primitives, after which the first coordinate of symbol "S" has been reduced to the

empty set "$\emptyset$". This signifies that no more primitives remain to be encoded, and that rule (I) - (b) can now be applied. This corresponds to the "no" exit from block (H) of Figure 3.3, which terminates the algorithm.

The entire sequence just presented for applying the rules of the algorithm is summarized by the flow chart whose components appear in both Figures 3.6 and 3.7. It should be reemphasized here that this sequential order of application for the rules was introduced to enhance the clarity of the logic of the algorithm, and does not represent the only order of application which could be used to generate a set of terminals. In fact, any rule which is applicable at any stage during a derivation with the algorithm can be applied at that stage as the next step of the algorithm. Any derivation so constructed which converts the starting type "S" (including its properly specified initial coordinates) into a set of terminal symbols (with coordinates) and nothing else, has produced the proper encodement for the specified T-F scene view. This is guaranteed by the structure of the rules of the algorithm, which permits a set of terminals (with coordinates) to be the complete result of a derivation only when each rule in the algorithm has been consistently applied with its intended interpretation as given earlier. This fact is obvious when any non-terminal symbol, including its coordinates, appears during a derivation and there is exactly one rule of the algorithm which can be

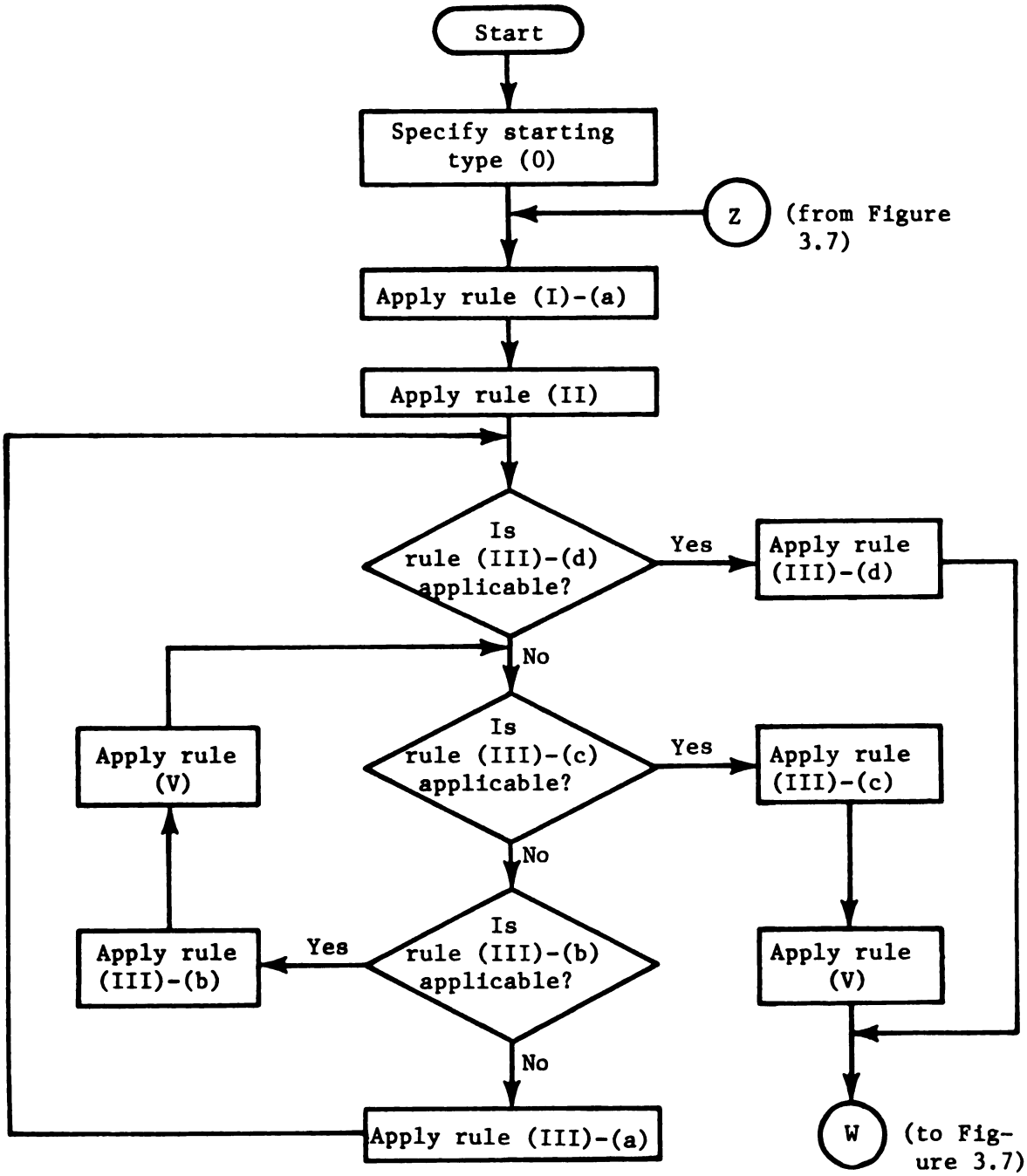


Figure 3.6. Initial component of flow chart for "linguistic" algorithm application.

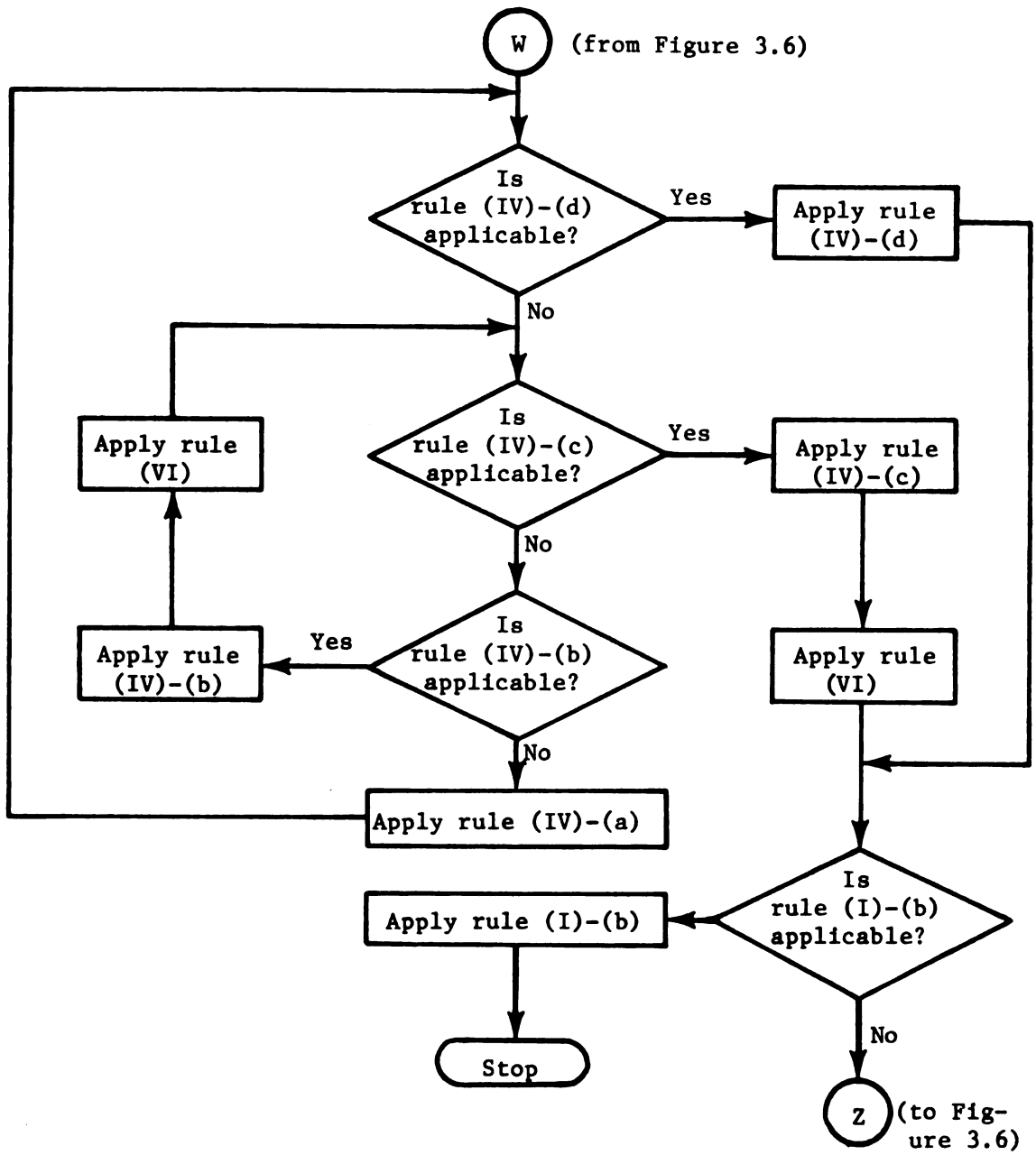


Figure 3.7. Terminal component of flow chart for "linguistic" algorithm application.

applied to that occurrence of that symbol. But the cases where more than one rule can be applied to an occurrence of a symbol in a derivation, and thus where a choice must be made, need further explanation. These cases are exhaustively enumerated for the algorithm as follows:

case(1): whenever rule (III) - (b) is applicable, rule (III) - (a) is also applicable;

case(2): whenever rule (III) - (c) is applicable, rule (III) - (a) is also applicable;

case(3): whenever rule (III) - (d) is applicable, rule (III) - (a) is also applicable;

case(4): whenever rule (III) - (c) is applicable, rule (III) - (b) is also applicable;

case(5): whenever rule (IV) - (b) is applicable, rule (IV) - (a) is also applicable;

case(6): whenever rule (IV) - (c) is applicable, rule (IV) - (a) is also applicable;

case(7): whenever rule (IV) - (d) is applicable, rule (IV) - (a) is also applicable;

and case(8): whenever rule (IV) - (c) is applicable, rule (IV) - (b) is also applicable.

For the application of the rules of the algorithm to be consistent with their intended interpretations, the rule first mentioned in each of the above eight case descriptions should be the rule chosen for application whenever that respective case arises in a derivation. The consequence of choosing to apply rule (III) - (a) in either

case (1) or case (2) would be the generation of the non-terminal symbol "V" with a first coordinate having a negative value. The only rule subsequently applicable for rewriting a "V" with a negative first coordinate would again be rule (III) - (a), whose application would only result in another appearance of the symbol "V" with some negatively valued first coordinate. As this can only continue, the non-terminal symbol "V" having some negatively valued first coordinate will always be present in any subsequently derived string. Similarly, the consequence of choosing to apply rule (IV) - (a) in either case (5) or case (6) would be the generation of the non-terminal symbol "H", appearing with a negatively valued first coordinate and persisting with some negatively valued first coordinate in all subsequently derived strings. The consequence of applying rule (III) - (a) in case (3), or rule (III) - (b) in case (4), would be the generation of the non-terminal symbol "V", appearing with a negatively valued second coordinate and persisting with some negatively valued second coordinate in all subsequently derived strings. Analogously, we finally have the consequence of applying rule (IV) - (a) in case (7), or rule (IV) - (b) in case (8), as the generation of the non-terminal symbol "H", appearing with a negatively valued second coordinate and persisting with some negatively valued second coordinate in all subsequently derived strings. So the conclusion in any of these eight

cases is that the choice and application of the "wrong" rule guarantees the existence of a non-terminal symbol in any string derived. Thus, the application of a rule of the algorithm with other than its intended interpretation at any point in a derivation precludes the possibility that a set of terminal symbols will be the exclusive result of that derivation.

As a result of the preceding extensive discussion, the algorithm has been "proven" in the following sense:

Theorem 3.3

Consider a completely specified starting type consisting of the terminal symbol "S" followed by a pair of parentheses enclosing two coordinates separated by a comma, where the first coordinate represents a T-F scene view "U" using its set notational description relative to the standard screen, and where the second coordinate represents the grid resolution "r" at which "U" is to be encoded. Additionally, let "T" denote the unique set of terminal symbols (with coordinates) which represents the encodement of "U" on the grid of resolution "r", where each element of "T" consists of the symbol "g" followed by two coordinates which specify the abscissa and ordinate of the center of one of the shaded grid squares.

Then there exists some sequential application of the rules of the algorithm (in which each rule may appear -- consecutively or non-consecutively -- zero or more times) which converts this completely specified starting type into exactly the set "T" and nothing else. Further, any set of terminal symbols (with coordinates) derived as the exclusive result of applying the algorithm to the completely specified starting type will be precisely the set "T".

3.3.6 Semantic Interpretation: Two Special Cases

When the algorithm is applied to encode a T-F scene view, situations can arise in which a special interpretation of coordinate values is required to insure that the algorithm generates the proper encodement. Such a situation occurs whenever a primitive of a T-F scene view coincides along its entire length with one of the grid lines. The interpretation problem results when conventionally undefined mathematical expressions arise in coordinates due to the "0" or "∞" value of the slope of the primitive. Since these slope values occur if and only if a primitive is either a horizontal or vertical line segment in a T-F scene view, the consideration of all special coordinate interpretations can be included in two cases -- the case where a primitive is a vertical line

segment, and the case where a primitive is a horizontal line segment. We first consider the vertical primitive case.

In the case where a primitive is a vertical line segment, it is parallel to the vertical grid lines and hence the coordinates "a" and "c" of symbol "L" in rule (II) of the algorithm are equal in value. This implies that the sixth coordinate generated by rule (II) for symbol "V" (and also for symbol "H") will have the form "$\frac{d-b}{0}$", where the value of "d-b" is non-zero since every scene primitive must have finite length. Since this sixth coordinate of "V" (and also of "H") represents the slope of the primitive, the expression "$\frac{d-b}{0}$" is interpreted as "∞". Now a vertical primitive lies either strictly between two consecutive vertical grid lines, or directly on a vertical grid line. If the primitive lies strictly between two consecutive vertical grid lines, then rule (II) of the algorithm generates the symbol "V" with the value of its first coordinate exceeding the value of its second coordinate by "1". This means that rule (III) - (d) will be applicable after the zero or more applications of rule (III) - (a) required to reduce the second coordinate of "V" to zero. Hence rules (III) - (b) and (III) - (c) will never be used when encoding this primitive, eliminating any concern about the factor "∞" entering into the coordinate expressions of "G_v" which involve the factor "m" in the numerator. On the other hand, if the

primitive lies directly on a vertical grid line, then rule (II) of the algorithm generates the symbol "V" with its first, second, and fourth coordinates all having the same value "a", which represents the x-coordinate value along the vertical grid line on which the primitive lies. In this case, rule (III) - (c) becomes applicable after "a" applications of rule (III) - (a), which has consequentially reduced the fourth coordinate "x" of "V" to the value zero. The symbol "G_V" is then generated, with the expression "m·x" having the form "$\infty \cdot 0$" in its y-coordinate. By interpreting this conventionally undefined expression as having the value "0", the coordinates of "G_V" will be the coordinates of one of endpoints of the primitive on the vertical grid line. This then specifies one point which is taken as the only intersection point of the primitive with the vertical grid line which generates shaded grid squares. The grid squares shaded as a result of this point insure the proper encodement if the primitive lies strictly between two consecutive horizontal grid lines, and will merely be redundant if the primitive crosses one or more horizontal grid lines (i.e., the shaded grid squares generated due to the intersections of the primitive with one or more horizontal grid lines will include those already generated as a result of the single point of intersection with a vertical grid line as defined above for the primitive). The only additional place where "m" is involved in an expression which needs to be interpreted

is in the x-coordinate of symbol "G_h". In this case, however, "m" always appears in the denominator, so the value of "∞" for "m" causes no problems of interpretation. In fact, since the value of "y" in the x-coordinate of "G_h" will always be finite, the expression "$\frac{1}{m} \cdot y$" will always have the value "0" when interpreted with "m" = "∞".

An analogous discussion holds for the case where a primitive is a horizontal line segment. Here the primitive is parallel to the horizontal grid lines, and hence the coordinates "b" and "d" of symbol "L" in rule (II) of the algorithm are equal in value. This implies that the sixth coordinate generated by rule (II) for symbol "H" (and also for symbol "V") will have the form "$\frac{0}{c-a}$", where the value of "c-a" is non-zero since every primitive must have finite length. Thus the sixth coordinate of "V" (and also of "H"), which is the slope of the primitive, has the value "0". Now a horizontal primitive lies either strictly between two consecutive horizontal grid lines, or directly on a horizontal grid line. If the primitive lies strictly between two consecutive horizontal grid lines, then rule (II) of the algorithm generates the symbol "H" with the value of its first coordinate exceeding the value of its second coordinate by "1". This means that rule (IV) - (d) will be applicable after the zero or more applications of rule (IV) - (a) required to reduce the second coordinate of "H" to zero. Hence rules (IV) - (b) and (IV) - (c) will never be used when encoding this

primitive, eliminating any concern about the factor "0" entering into the coordinate expressions of "G_h" which involve the factor "m" in the denominator. On the other hand, if the primitive lies directly on a horizontal grid line, then rule (II) of the algorithm generates the symbol "H" with its first, second, and fifth coordinates all having the same value "b", which represents the y-coordinate value along the horizontal grid line on which the primitive lies. In this case, rule (IV) - (c) becomes applicable after "b" applications of rule (IV) - (a), which has consequentially reduced the fifth coordinate "y" of "H" to the value zero. The symbol "G_h" is then generated, with the expression "$\frac{1}{m}\cdot y$" taking the form "$\frac{1}{0}\cdot 0$" in its x-coordinate. By interpreting this conventionally undefined expression as having the value "0", the coordinates of "G_h" will be the coordinates of one of the endpoints of the primitive on the horizontal grid line. This then specifies one point which is taken as the only intersection point of the primitive with the horizontal grid line which generates shaded grid squares. The grid squares shaded as a result of this point insure the proper encodement if the primitive lies strictly between two consecutive vertical grid lines, and will merely be redundant if the primitive crosses one or more vertical grid lines. The only additional place where "m" is involved in an expression which needs to be interpreted is in the y-coordinate of the symbol "G_v". In this case, however,

"m" always appears in the numerator with a finite value of "x", so that the expression "m·x" has the conventional interpretation of "0" when "m" = "0".

In summary, the two cases which require special semantic interpretations for the values of coordinates are: (1) when the expression "$\infty \cdot 0$" appears in the y-coordinate of "G_v" as a result of encoding a primitive with infinite slope lying directly on a vertical grid line; and (2) when the expression "$\frac{1}{0} \cdot 0$" appears in the x-coordinate of "G_h" as a result of encoding a primitive with zero slope lying directly on a horizontal grid line. Both of the expressions "$\infty \cdot 0$" and "$\frac{1}{0} \cdot 0$" are conventionally mathematically undefined, but are both to be interpreted here as representing the numerical value "0" when they arise during application of the algorithm, which will insure proper encodements. These are the only two special coordinate value interpretations needed in any derivation which results exclusively in a set of terminals. If either of the two mathematically undefined expressions occur for any other reason than the ones given in (1) and (2) above, or if any of the coordinates of any occurrence of "G_v" or "G_h" have an interpreted value of "∞", then at least one rule of the algorithm has previously been applied contrary to its intended interpretation. Such an occurrence signifies that the derivation will not have a set of terminals as its exclusive result.

3.3.7 Grid Extension

When the algorithm is used to encode a T-F scene view which has at least one of its primitives touching a side border of the standard screen, the encodement generated by the algorithm will include one or more terminal symbols "g" having coordinates which specify a point lying outside the region of the encodement grid superimposed on the standard screen. This is due to the fact that the touching point of a primitive on a border of the standard screen is also an intersection point of that primitive with either a horizontal or vertical border line of the encodement grid. Hence, the algorithm (through rule (V) or (VI)) generates terminals whose coordinates represent the centers of shaded grid squares on both sides of that grid border line, although the grid has not yet been defined on one of those sides. In such a situation, then, the problem is that grids which are confined to the area of the standard screen, and whose borders must be aligned with those of the standard screen, are not adequate to represent the complete encodement of the T-F scene view.

A solution is provided by symmetrically extending the overall size of an encodement grid through an increase in the total number of squares in the grid, while maintaining the size of the individual grid squares and the alignment of the original grid (now symmetrically embedded in the "extended" grid) with the standard screen. This extension is accomplished for the grid of arbitrary resolution r --

as defined in Definition 2.7 -- by merely appending an additional grid square (of the same size as those already in the grid) along all sides and corners of the grid border. Note that this puts the borders of the standard screen strictly within the borders of the extended grid. Hence, any T-F scene view appearing on the standard screen -- even a view which touches the border of the standard screen -- will have its complete encodement contained entirely within the borders of the extended grid of any resolution r.

The concept of an "extended grid" is formalized as follows:

Definition 3.2

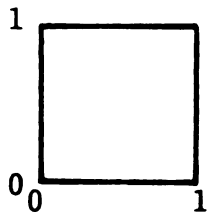
The extended grid of resolution r is the grid consisting of $(r+2)^2$ identical squares, formed by extending the grid of resolution r "1" grid unit (at resolution r) on all four sides (left, right, up, and down) and diagonally at all four corners.

The coordinate system of the extended grid of resolution r is formed by symmetrically extending both the x- and y-axes of the grid of resolution r by two grid units -- the x-axis being extended one grid unit in both the positive and negative x directions, and the y-axis being extended one grid unit in both the positive and negative y directions. The origin of the coordinate system is maintained in its

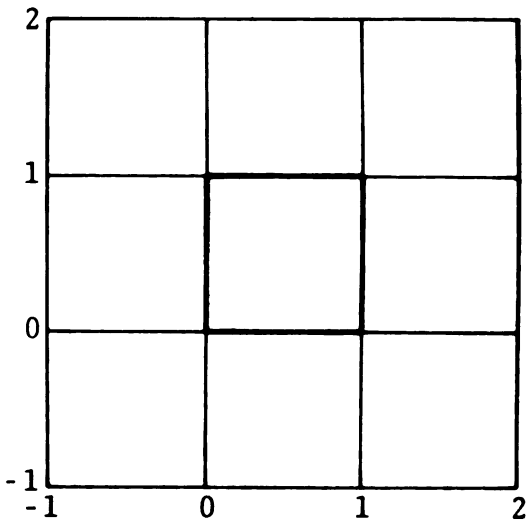
location before the extension, so that the origin appears at the lattice point located 1 grid unit right and 1 grid unit up from the lower left corner of the extended grid. Thus the coordinate axes labels for the extended grid of resolution r run from "-1" to "r+1".

Note that this definition insures that the grid and extended grid of resolution r both have the same size grid squares, and that the origins of their coordinate systems coincide. This preserves the definition of resolution for grids, and insures that all the results obtained in Section 3.2 -- which characterize the effects of grid resolution changes on T-F scene view descriptions -- are directly applicable for extended grids. Hence, it is no loss of generality to interpret an encodement generated by the algorithm on the appropriate extended grid. And such an interpretation will insure that the coordinates of all terminal symbols "g" generated by the algorithm will be within the coordinate system of the extended grid.

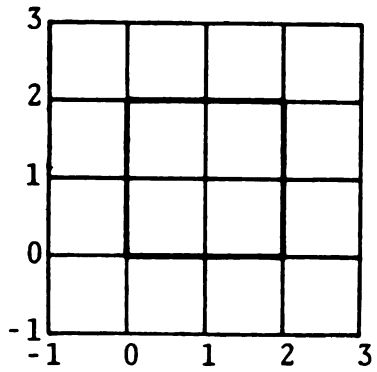
Some examples of finite extended grids with different resolutions defined relative to the same standard screen are shown in Figure 3.8. Note that these extended grids, unlike the corresponding (non-extended) grids discussed in Section 2.4, are not all of the same absolute size. However, the grid of resolution r is embedded in the corresponding extended grid of the same resolution, with its



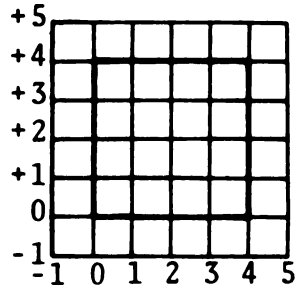
(a) The Standard Screen (Grid of Resolution 1).



(b) The Extended Grid of Resolution 1.



(c) The Extended Grid of Resolution 2.



(d) The Extended Grid of Resolution 4.

Figure 3.8. Examples of extended grids.

borders located "1" grid unit in from the corresponding borders of the extended grid. Further, bear in mind that the x and y coordinate axes are located "1" grid unit in from the bottom and left borders, respectively, of the extended grid. This is reflected by the coordinate labeling along both the bottom and left borders of the extended grids in Figure 3.8, which puts the origin of the coordinate system one grid unit up and right from the lower left corner.

Because the coordinate systems of both the grid of resolution r and the extended grid of resolution r are identical on the standard screen, no confusion should result if a grid is considered to be an extended grid of the same resolution whenever necessary or appropriate. Hence, we will hereafter use the terms "grid" and "extended grid" interchangeably, unless designated otherwise.

3.4 Algorithmic Encodement of Snapshots

The "linguistic" algorithm (so named because of its resemblance in both structure and application to formal grammars) of Section 3.3 can be considered to generate snapshots -- instead of encodements consisting of shaded grid squares in specific locations on a grid -- if the coordinate values of the terminals are interpreted in the proper manner. Recall from Chapter 2 that although a snapshot is defined with respect to some specified grid resolution, it is only the structural form (including the

overall orientation) of the entire pattern of shaded grid squares which defines the snapshot, irrespective of the relative translational position of that pattern on the grid. Thus, if a snapshot is completely represented (relative to some designated grid resolution) by a set of symbols, each of which has a unique pair of coordinates specifying the center of one of the shaded grid squares in the snapshot, then the absolute values of these coordinates are of no consequence with regard to their determination of the location of the pattern on the grid; it is only the relative positional relationships (location and orientation) existing between these coordinates which characterize the snapshot. This implies that any set of terminals (with coordinates) generated by the algorithm can be considered to represent a snapshot relative to the encodement resolution, under the interpretation that the coordinates of the terminal symbols only have positional significance relative to each other.

As a consequence of the preceding remarks, we note that distinct grid encodements (at the same resolution) which have the same relative positional relationships existing between their elements represent the same snapshot. A characterization of these equivalent encodement representations for a snapshot will be presented in Theorem 3.4, using the notation defined next.

Let "T" be a set of terminal symbols (with coordinates) as follows:

$$T=\{g(x_1,y_1),g(x_2,y_2),\ldots,g(x_q,y_q)\},$$

where q is a non-negative integer. Also, let "f" denote a coordinate transformation. Then we use the notation "f(T)" to denote the set

$$\{g(x_1',y_1'),g(x_2',y_2'),\ldots,g(x_q',y_q')\},$$

where $(x_i',y_i') = f(x_i,y_i)$ for i=1,2,...,q.

Theorem 3.4

Let T_1 and T_2 be two sets of terminal symbols (with coordinates), each of which is the exclusive result of a derivation using the "linguistic" algorithm of Section 3.3 with the same encodement resolution.

Then, T_1 and T_2 are equivalent encodement representations for the same snapshot (at the encodement resolution) if and only if

$$f(T_1) = T_2$$

for some coordinate transformation "f" of the form

$$f(x,y) = (x+a,y+b),$$

where "a" and "b" are integer constants (either of which may be positive, zero, or negative).

Proof

The proof is based on the fact that the form specified for coordinate transformation "f" in the theorem defines "f" as a translation. Therefore, the

application of "f" to any set of coordinates "T" merely redefines the absolute location of all the points represented by the coordinates in "T". The result f(T) is a set having the same number of elements as set "T", and all relative positional relationships (location and orientation) existing between the elements of "T" are maintained between the corresponding elements of f(T).

Now consider the application of f to the encodement T_1. The x-coordinate of each terminal symbol in T_1 will be consistently relocated by "a" units, while the y-coordinate of each terminal symbol in T_1 will be consistently relocated by "b" units. Hence the result $f(T_1)$ is a set consisting of the same number of terminals as T_1, with f defining a 1-1 correspondence between these terminals. Further, the relative positional location existing between the coordinates of any two terminals in encodement T_1 is maintained between the translated coordinates of the corresponding terminals in $f(T_1)$. But the coordinates of the terminals in $f(T_1)$ represent the centers of grid squares on the encodement grid, since they are derived by integer-valued translations -- through integers "a" and "b" of f -- of the coordinates of the elements in encodement T_1. Thus the set $f(T_1)$ can be interpreted as representing a set of shaded grid squares in an encodement, in which case the relative

positional relationships existing between any two shaded grid squares in encodement T_1 is the same as the relative positional relationships which exist between the two corresponding shaded grid squares in encodement $f(T_1)$. This means that T_1 and $f(T_1)$ are equivalent encodement representations for the same snapshot (at the encodement resolution).

Now if $f(T_1) = T_2$, then obviously $f(T_1)$ and T_2 are equivalent encodement representations for the same snapshot, since they are identical. And since we have already shown that T_1 and $f(T_1)$ are equivalent encodement representations for the same snapshot, it follows that T_1 and T_2 are equivalent encodement representations of the same snapshot (at the encodement resolution).

Conversely, assume that T_1 and T_2 are equivalent encodement representations for the same snapshot (at the encodement resolution). Then T_1 and T_2 have the same number of elements[7], and a 1-1 correspondence must exist between their elements such that the

---

[7] This is a necessary condition for two encodements to be equivalent representations for the same snapshot. For if the number of elements in encodements T_1 and T_2 were different, then one would represent a pattern containing more shaded grid squares than the other. It would then follow that the two shaded grid square patterns represented could not be the same snapshot.

relative positional relationships existing between any two shaded grid squares in encodement T_1 also exists between the two corresponding shaded grid squares in encodement T_2. Let "f'" denote the 1-1 transformation from the elements of T_1 to the elements of T_2 which realizes this 1-1 correspondence, so that $f'(T_1) = T_2$. Then f' must be a coordinate transformation which preserves in $f'(T_1)$ the relative positional relationships existing between the elements of T_1, implying that f' is a translation. Further, since $f'(T_1) = T_2$ and T_2 is an encodement, the coordinates of all the terminal symbols in $f'(T_1)$ must represent the centers of grid squares on the encodement grid. Similarly, since T_1 is an encodement, the coordinates of all the terminal symbols in T_1 also represent the centers of grid squares on the encodement grid. It thus follows that the translation increments of f' must be integer values for both the x-coordinate and y-coordinate relocations. Hence f' is a translation of the same form as f, with $f'(T_1) = T_2$.

Q.E.D.

The intuitive interpretation of Theorem 3.4 is that two T-F scene view encodements (on the same resolution grid) represent the same snapshot whenever the entire pattern of shaded grid squares defined by one of the encodements can be translated to coincide precisely with the

entire pattern of shaded grid squares defined by the other encodement. This suggests the existence of an effective procedure for determining whether or not two encodements represent the same snapshot, and the construction of such a procedure appears as the proof of the next theorem.

Theorem 3.5

There exists an effective procedure for determining whether or not two T-F scene view encodements (relative to the same resolution grid) represent the same snapshot at the encodement resolution.

Proof

Let T_1 and T_2 represent two encodements relative to the same resolution grid, where (using the notation of the "linguistic" algorithm)

$$T_1 = \{g(x_{1,1},y_{1,1}),g(x_{1,2},y_{1,2}),\ldots,g(x_{1,p},y_{1,p})\}$$

and

$$T_2 = \{g(x_{2,1},y_{2,1}),g(x_{2,2},y_{2,2}),\ldots,g(x_{2,q},y_{2,q})\},$$

with p and q non-negative integers. Using Theorem 3.4, it suffices to show that one can effectively determine whether or not there exists a translation f having the form defined in Theorem 3.4 for which $f(T_1) = T_2$. A procedure for accomplishing this is now described.

First compare the values of p and q, which represent the number of elements in T_1 and T_2,

respectively. If $p \neq q$, then T_1 and T_2 cannot be equivalent encodement representations for the same snapshot (see Footnote 7), and we terminate this procedure with that conclusion. On the other hand if $p = q$, then compute the values of "a" and "b" as follows:

$$a = \min\{x_{2,1}, x_{2,2}, \ldots, x_{2,q}\} - \min\{x_{1,1}, x_{1,2}, \ldots, x_{1,q}\}$$

and

$$b = \min\{y_{2,1}, y_{2,2}, \ldots, y_{2,q}\} - \min\{y_{1,1}, y_{1,2}, \ldots, y_{1,q}\}.$$

Note that "a" represents the horizontal distance and direction (relative to T_1) that exists between the leftmost shaded grid squares of T_1 and T_2, while "b" represents the vertical distance and direction (relative to T_1) that exists between the lowest shaded grid squares of T_1 and T_2. Further, both "a" and "b" are integers, since each of the x- or y-coordinates of any of the terminals in encodements T_1 and T_2 represents the middle line between two consecutive grid lines, making the difference between any two x-coordinates or any two y-coordinates an integer. So if T_1 and T_2 are encodements which represent the same snapshot, then the values of "a" and "b" represent the integer valued x and y distances, respectively, through which T_1 must be translated on the

encodement grid to coincide with T_2. Thus, we can define translation "f" by

$$f(x,y) = (x+a,y+b),$$

and determine $f(T_1)$. Then if $f(T_1) = T_2$, we conclude (using Theorem 3.4) that T_1 and T_2 are equivalent representations for the same snapshot at the encodement resolution. On the other hand, if $f(T_1) \neq T_2$, we conclude (again using Theorem 3.4) that T_1 and T_2 are not equivalent encodement representations for the same snapshot. In either case, the procedure is terminated.

Q.E.D.

Both theorems 3.4 and 3.5 state results which will be useful in examining the characterization of scene views through the snapshots they generate. We turn to such an examination next in Chapter 4.

Chapter 4

SCENE VIEW RECOGNITION WITH SNAPSHOT SIGNATURES

4.1 Scene View Recognition: General Considerations

This chapter will deal with the problem of the recognition of scene views from their encodements. The problem stems from the fact that the encodement of a scene view on a finite grid does not necessarily preserve the full structural detail which characterizes that scene view. It is apparent that grids of coarse resolution (lower resolution values) are less capable of faithfully encoding detail in a scene view than grids of fine resolution (higher resolution values) would be. Hence the problem becomes one of identifying a scene view from its approximate representation on an encodement grid of finite resolution, where the degree of difficulty in making such an identification is directly related to the relative coarseness of the grid.

The scene view recognition problem will be examined in the context of determining which of a finite number of known candidate scene views is represented by a given encodement (or encodements). Hence, we assume that a

finite number of prototype scene views are given, along with one or more encodements of a scene view which is an unknown one of the prototypes. The problem then is to identify which one of the prototype scene views generated the encodement (or encodements). Practically, an exact identification may not always be possible, in which case the goal is to keep the probability of making an identification error small.

We shall assume that any encodement of an unknown scene view results from some random positioning of an encodement grid over that scene view. This is a realistic assumption in many practical applications. For example, consider a situation in which a remote sensing "camera" is arbitrarily positioned (without tilting) to record the "picture" of some unknown object currently in view, after which the recorded "picture" is transmitted elsewhere for processing. In this case, the encodement of the unknown scene view is analogous to the "picture" of the unknown object transmitted from the camera. Further, the resolution of the encodement grid is analogous to the resolution with which the camera records the picture. Finally, the random positioning of the encodement grid over the unknown scene view corresponds to the arbitrary positioning of the camera before it records a picture of the unknown object.

The consequence of assuming the random positioning of an encodement grid over a scene view is that the scene

view can generate, and thus be associated with, two or more snapshots. This implies that the characterization of a scene view in terms of the encodements it can generate at a specified grid resolution must be based on more than one snapshot. This will become more evident in the discussion and examples of the next section.

4.2 Snapshot Sets

It was noted in Chapter 2 that the encodement which results from a grid placed randomly over a scene view is dependent upon, among other things, the relative translational position existing between the grid and scene view. This became even more apparent in Chapter 3, where the algorithm for generating the encodement for a scene view was dependent upon the translationally-fixed ("T-F") position of that scene view with respect to the encodement grid. Since we now wish to consider the encodement which results from a random positioning of a grid over a scene view, it might appear that the encodements produced by all possible T-F positionings of the scene view relative to the encodement grid would need to be examined. Fortunately however, we are interested only in the distinct snapshots which are represented by all these encodements. In this case, due to the symmetry of the encodement grid, any two T-F scene views which represent integer valued translations -- both horizontally and vertically -- of each other on the encodement grid will generate encodements

which represent the same snapshot. This observation is rephrased in Theorem 4.1 in terms of a restricted set of encodement grid translations which will produce all possible snapshots of a scene view on that grid. The proof will utilize the following notation. Let "D" represent the set notational description of a T-F scene view relative to some encodement grid, where

$$D = \left\{ \begin{matrix} \{(a_1,b_1),(c_1,d_1)\},\{(a_2,b_2),(c_2,d_2)\}, \\ \ldots,\{(a_p,b_p),(c_p,d_p)\} \end{matrix} \right\}$$

with p a positive, non-zero integer. Also let "f" denote a two-dimensional coordinate transformation. Then the notation "f(D)" denotes the set

$$\left\{ \begin{matrix} \{f(a_1,b_1),f(c_1,d_1)\},\{f(a_2,b_2),f(c_2,d_2)\}, \\ \ldots,\{f(a_p,b_p),f(c_p,d_p)\} \end{matrix} \right\} .$$

Theorem 4.1

All possible snapshots of a scene view on an (extended) encodement grid are represented by the encodements which result from all possible translations of the encodement grid which move the scene view no more than $\frac{1}{2}$ grid unit, in both the horizontal and vertical directions, away from its (arbitrary) initial T-F position on that encodement grid.

Proof

Let "D_r" denote the set notational description of some T-F scene view in its initial position

relative to an (extended) encodement grid of resolution r. Further, let "M" be a set of coordinate translations as follows:

$$M=\{f \mid f(x,y)=(x+a,y+b), \text{ with } a, b \in [-.5,.5]\},$$

where

$$[-.5,.5] = \{z \mid -.5 \leq z \leq .5\}.$$

Then the set

$$V = \{f(D_r) \mid f \in M\}$$

denotes all the T-F scene views which result when the encodement grid is translated such that the scene view is kept $\frac{1}{2}$ grid unit or less, in both the horizontal and vertical directions, from its initial T-F position on that encodement grid.

Now consider some translation of the encodement grid which moves the scene view more than $\frac{1}{2}$ grid unit either horizontally or vertically (or both) away from its initial position on the grid. Then the resulting T-F scene view is given by $f'(D_r)$, where f' is a coordinate translation of the form

$$f'(x,y) = (x+a',y+b'),$$

where either $a' \notin [-.5,.5]$ or $b' \notin [-.5,.5]$, or both. Hence $f'(D_r) \notin V$. But note that

$$a' = a' - 0 = a' - \lfloor a' + .5 \rfloor + \lfloor a' + .5 \rfloor$$

$$= (a' - \lfloor a' + .5 \rfloor) + \lfloor a' + .5 \rfloor ,$$

where the quantity in parentheses denotes some value in [-.5,.5], and where $\lfloor a' + .5 \rfloor$ denotes an integer.

Similarly,

$$b' = (b' - \lfloor b' + .5 \rfloor) + \lfloor b' + .5 \rfloor ,$$

where the quantity in parentheses denotes some value in [-.5,.5], and where $\lfloor b' + .5 \rfloor$ denotes an integer. Let f_1 denote the translation

$$f_1(x,y) = (x + a' - \lfloor a' + .5 \rfloor , y + b' - \lfloor b' + .5 \rfloor),$$

and f_2 denote the translation

$$f_2(x,y) = (x + \lfloor a' + .5 \rfloor , y + \lfloor b' + .5 \rfloor).$$

Then $f'(D_r) = f_2(f_1(D_r))$. But $f_1(D_r) \in V$, and $f_2(f_1(D_r))$ -- which is $f'(D_r)$ -- denotes the translation of T-F scene view $f_1(D_r)$ some integer number of grid units in both the horizontal and vertical directions on the encodement grid. Such a translation (i.e., f_2) guarantees that corresponding points (under f_2) along the primitives in both $f_1(D_r)$ and $f'(D_r)$ will have the same relative positional location with respect to the pattern formed by the vertical and horizontal lines of the encodement grid. In particular, the intersection points with grid lines of the primitives in $f_1(D_r)$ will be mapped (under f_2) exactly onto the intersection points with grid lines of primitives in $f'(D_r)$, with the corresponding points of intersection of the primitives in $f_1(D_r)$ and $f'(D_r)$ appearing on the same type of grid line (vertical or horizontal) at the same relative distance (and in the same relative direction) from the grid line of opposite type which is nearest each

respective point. Thus the corresponding points of intersection with grid lines of the scene primitives in $f_1(D_r)$ and $f'(D_r)$ cause precisely the same pattern of grid squares to be shaded. Further, since $f_1(D_r)$ and $f'(D_r)$ are related by a translation (i.e., f_2), the relative positional relationships existing between corresponding points of intersection of the scene primitives with grid lines will be the same in both $f_1(D_r)$ and $f'(D_r)$. Hence, if T_1 and T' represent the encodements (as a set of terminals with coordinates which denote the centers of the shaded grid squares on the encodement grid, as in Chapter 3) produced by T-F scene views $f_1(D_r)$ and $f'(D_r)$, respectively, then $f_2(T_1) = T'$. Hence, by Theorem 3.4, it follows that T_1 and T' are equivalent encodement representations for the same snapshot on the grid of resolution r.

Thus, the theorem has been proved, since translation f' was arbitrary.

Q.E.D.

Note that the limited range of translations of the encodement grid specified by Theorem 4.1 as being sufficient to generate all the possible snapshots for a scene view also insures that every such snapshot can be entirely represented within the borders of the extended grid of the encodement resolution. This follows because any scene,

and hence any scene view, can be contained entirely within the borders of the standard screen. Letting some T-F position of a scene view on the standard screen define the initial T-F position of that scene view relative to the (extended) encodement grid will then insure that any point on any primitive of the scene view is at least 1 grid unit (at resolution r) away from the border of the extended grid of arbitrary resolution r. Hence, any translation of the encodement grid from this initial position which has its horizontal and vertical components of motion limited to $\frac{1}{2}$ grid unit (at the encodement resolution) will produce only T-F scene views which, at any point, are at least $\frac{1}{2}$ grid unit away from any border of the extended grid of the encodement resolution. Thus, since no border line of this extended grid is touched, the entire encodement -- and hence the snapshot it represents -- lies within its borders.

The set of all the distinct snapshots which a scene view can generate on an encodement grid can be considered as a collective representation of that scene view relative to that grid. This suggests the following terminology.

Definition 4.1

A snapshot set of a scene view, relative to an encodement resolution r, is the collection of all snapshots which can result by encoding that scene

view (in different relative translational positionings) on the extended grid of resolution r.

The snapshot set of a scene view relative to any finite encodement resolution r will itself be finite, since the maximum number of binary encodements -- each representing a (not necessarily distinct) snapshot -- which can be formed by the grid squares of the extended grid of resolution r is $2^{(r+2)^2}$. Of course, not all of these possible snapshots will appear in the snapshot set of a particular scene view. In fact, the number of distinct snapshots in a snapshot set of a scene view can be relatively small, as illustrated in Figure 4.1. The shaded grid square patterns labeled (a) through (g) in Figure 4.1 represent the seven snapshots in the snapshot set of an isosceles triangle relative to an encodement grid whose resolution makes the base and altitude of the triangle 2 grid units and 1 grid unit, respectively. Similarly, the shaded grid square patterns labeled (I) through (IV) in Figure 4.1 represent the four snapshots in the snapshot set of a square relative to an encodement grid whose resolution makes the side length of the square 2 grid units. The appearance of the figure itself (triangle or square) in each of the snapshots is intended to illustrate its fixed orientation throughout the encodement process. Also, the relative translational positioning existing between the figure and the grid lines shown in each snapshot illustrates how that snapshot might

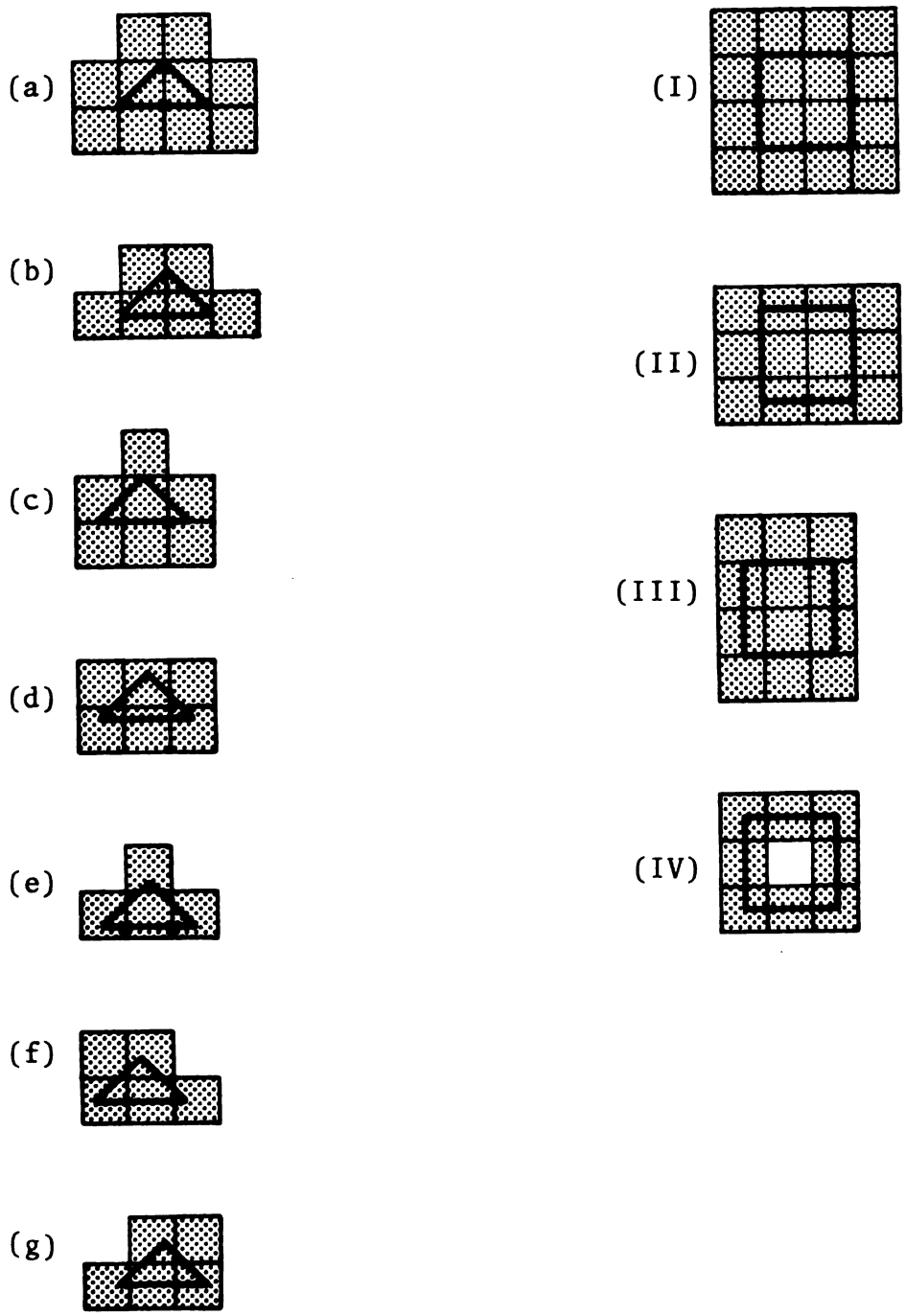


Figure 4.1. Two examples of snapshot sets.

be generated in grid encodements of that figure. However, remember that the figure itself is not part of the snapshot -- only the shaded grid squares are.

For scene views having a more complex structure than those just considered in Figure 4.1, the determination of a snapshot set -- especially with respect to high encodement resolutions -- can be a formidable task. However, a practical method exists for approximating a snapshot set in such cases. Essentially, the method consists of applying the "linguistic" algorithm of Chapter 3 a finite number of times to produce the T-F scene view encodements which result from randomly chosen relative translational positionings between the encodement grid and scene view. The random sample of encodements which results for that scene view can then be examined using the effective procedure given in the proof of Theorem 3.5 to eliminate any duplicate specifications of the same snapshot within the sample. The resulting collection of encodements, interpreted as a collection of distinct snapshots, then represents an approximation for the snapshot set of the scene view relative to the resolution of the encodement grid.

A method for choosing the random relative translational positionings between the encodement grid and scene view in the snapshot set approximation procedure will now be explained. Let "v" represent the scene view whose snapshot set is to be approximated, and let "v_o" denote some T-F position of "v" whose set notational description

relative to the standard screen is denoted by "D". Consider the set "V" given by

$$V = \{f(D) \mid f(x,y)=(x+a,y+b) \text{ and } a,b\varepsilon[-\tfrac{1}{2r},\tfrac{1}{2r}]\},$$

where $[-\frac{1}{2r},\frac{1}{2r}] = \{z \mid -\frac{1}{2r} \leq z \leq \frac{1}{2r}\}$, and r is a positive, non-zero integer. Each of the elements of V is the set notational description of a T-F position of "v" relative to the standard screen, produced by some translation which moves "v" away from its position at "v_o" no more than $\frac{1}{2r}$ grid units, in both the horizontal and vertical directions, on the standard screen. But since a distance of $\frac{1}{2r}$ grid units on the standard screen corresponds to a distance of $\frac{1}{2}$ grid unit on the (extended) grid of resolution r, each of the elements of V can be interpreted as the set notational description of a T-F position of "v" relative to the standard screen in which "v" has been translated away from its position at "v_o" no more than $\frac{1}{2}$ grid units, in both the horizontal and vertical directions, on the (extended) grid of resolution r. Hence, it follows from Theorem 4.1 that all possible snapshots of "v" relative to resolution r are generated by encoding all the T-F scene views in V on the extended grid of resolution r. But V is an infinite set, so it is not practically possible to individually encode each of the T-F scene views in V. However, any two randomly and independently chosen T-F scene views in "V" have the same probability of encoding any given snapshot of "v". Hence, it seems

reasonable to expect that the snapshot set of "v" can be usefully approximated by the collection of snapshots which results from encoding all the T-F scene views in some randomly chosen finite subset V' of V, provided that the number of elements in V' is sufficiently large.[1] Note that a (non-unique) set V' of n elements can be formed from V by independently choosing each $v' \varepsilon V'$ as follows: (1) randomly select two independent real values "a" and "b" from the interval $[-\frac{1}{2r},\frac{1}{2r}]$, and then (2) determine v' as f(D), where f is the coordinate translation defined by f(x,y) = (x+a,y+b). The elements of V' can then be individually encoded at resolution r by applying the "linguistic" algorithm of Chapter 3 "n" different times.

It should be apparent that the randomness in the snapshot set approximation procedure just described can be incorporated directly within the structure of the "linguistic" algorithm of Chapter 3 by appropriately modifying its "productions". One such modification will be presented which keeps the same starting type S, but introduces an additional non-terminal symbol "S'". Additionally, the

---

[1] The number of elements chosen for V' should certainly depend on the relative complexity of scene view "v" -- the more complex the structure of "v", the larger the set V' should be. Further, if the set V' is large enough, the only snapshots of "v" which are likely to be omitted by the snapshot set approximation computed from V' would be those which have a low probability of being the encodement result from a random translational positioning of the encodement grid over "v". These occurrence probabilities for snapshots are discussed in Section 4.3.

symbol "ran" will appear in the coordinates to denote a randomly chosen real value from the set $\{z|0\leq z\leq 1\}$, where that value is chosen independently for each occurrence of the symbol "ran". The symbols "r_1" and "r_2" are used to represent two independently generated random real values between 0 and 1. Now consider the following group of three rules:

(I)-(a)' $S(\{L_1,L_2,\ldots,L_p\},r) \rightarrow$

$S'(\{L_1,L_2,\ldots,L_p\},r,ran,ran)$,

(I)-(b)' $S'(\{L_1,L_2,\ldots,L_k\},r,r_1,r_2) \rightarrow$

$S'(\{L_1,L_2,\ldots,L_{k-1}\},r,r_1,r_2)$

$\cup\, L(r\cdot a_k+r_1-\frac{1}{2},r\cdot b_k+r_2-\frac{1}{2},r\cdot c_k+r_1-\frac{1}{2},r\cdot d_k+r_2-\frac{1}{2})$

(I)-(c)' $S'(\emptyset,r,r_1,r_2) \rightarrow \emptyset$.

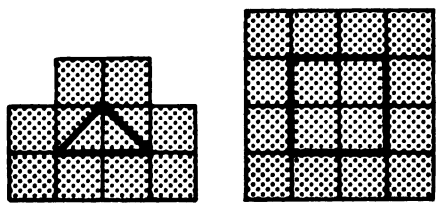
Modify the "linguistic" algorithm of Chapter 3 by replacing rule (I) - (a) and (I) - (b) with the rules (I) - (a)', (I) - (b)', and (I) - (c)'. Then for some scene view "v", let the first coordinate of starting symbol "S" represent the set notational description for some T-F position "v_o" of "v" on the standard screen. The second coordinate of starting symbol "S" should be the desired encodement resolution for "v". Then any set of terminals which is the exclusive result of a derivation using the modified algorithm represents the snapshot which results from some

random translation of "v" within $\frac{1}{2}$ grid units, both horizontally and vertically, from its initial position "v_o" on the encodement grid. Note that this random translation of "v" is done independently, but consistently, for each of its primitives by the coordinates of symbol "L" in rule (I) - (b)'.

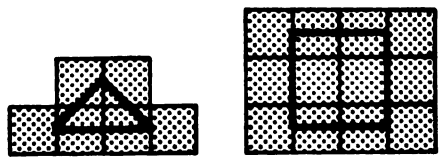
Suppose a simple or composite scene view is divided into several component parts, and the snapshot sets for each of these component parts then determined separately on the encodement grid of resolution r. Contrary to what might be expected, the snapshot set of the total scene view relative to resolution r is not necessarily represented by the set of all possible combinations which can be formed by taking one element from each of the snapshot sets of its component parts. This is illustrated with Figures 4.1 and 4.2. In Figure 4.1 the snapshot sets separately determined for each of two object views are represented -- the 7 snapshots of the triangle define its snapshot set, while the 4 snapshots of the square define its snapshot set. This represents 28 possible combinations of two snapshots (one taken from each snapshot set), all of which might be expected to appear in some composite scene view containing both of the objects. However, Figure 4.2 illustrates the same triangle and square having a fixed positioning relative to each other, thus forming a composite scene view of the two objects which has only the 7 (not 28) snapshots shown in Figure 4.2 in its snapshot

Composite Scene View
(with objects from
Figure 4.1)

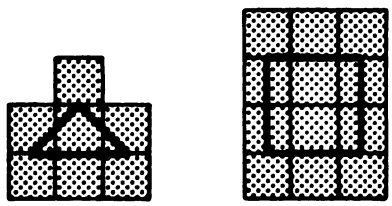
Bottoms of square and triangle are at the same vertical level, with the square to the right of the triangle. The objects are separated horizontally by 3 grid units at the encodement resolution.



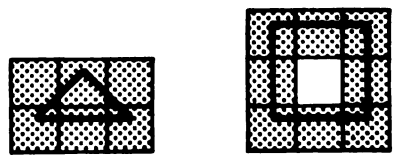
(a) - (I)



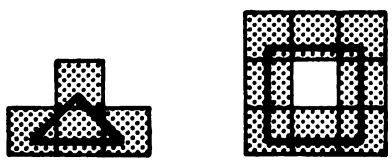
(b) - (II)



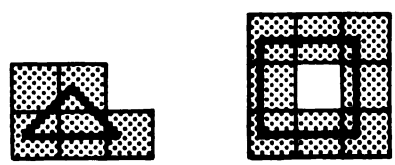
(c) - (III)



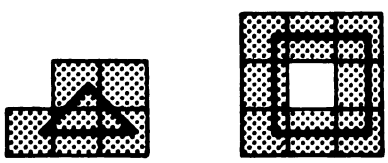
(d) - (IV)



(e) - (IV)



(f) - (IV)



(g) - (IV)

Figure 4.2. A composite scene view and its snapshot set.

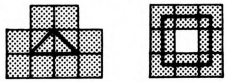
set relative to the encodement grid. The labeling of each snapshot of the composite scene in Figure 4.2 reflects the two snapshots from Figure 4.1 -- one from each object -- which combined to form that snapshot of the composite scene view. Figure 4.3 illustrates that the snapshot set of a composite scene is dependent upon the relative positioning between its component objects. The triangle and square of Figure 4.1 have a fixed position relative to each other in Figure 4.3 which is different from their positioning in Figure 4.2, thus forming a different composite scene view which has the 8 snapshots shown in Figure 4.3 as part of its snapshot set on the encodement grid.[2] The comparison of Figure 4.3 with Figure 4.2 thus reflects the effect that the relative positioning of the component objects in a composite scene view can have on the snapshot set -- in terms of both the number and form of its elements -- of that composite scene view. The overall conclusion to be drawn from these illustrations is that the "superposition" principle doesn't hold in the formation of a snapshot set -- i.e., the snapshot set of a simple or composite scene view "v" cannot be constructed by forming all possible combinations of snapshots from the snapshot sets of

---

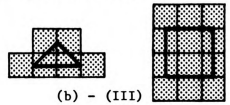
[2] The snapshots of the composite scene view in Figure 4.3 which are formed from combinations involving snapshots (e), (f), and (g) of Figure 4.1 have not been shown in Figure 4.3.

Composite Scene View
(with objects from
Figure 4.1)

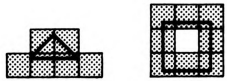
Bottom of square is $\frac{1}{2}$ grid unit below bottom of triangle, with the square to the right of the triangle. The objects are separated horizontally by $3\frac{1}{2}$ grid units at the encodement resolution.



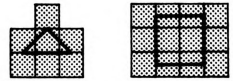
(a) - (IV)



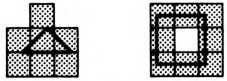
(b) - (III)



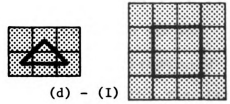
(b) - (IV)



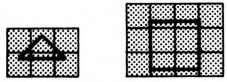
(c) - (II)



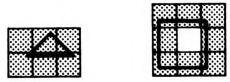
(c) - (IV)



(d) - (I)



(d) - (II)



(d) - (IV)

Figure 4.3. Another composite scene view and part of its snapshot set.

the component parts of the scene view considered individually. While some subset of all these possible snapshot combinations will define the snapshot set of "v", exactly which subset it will be is determined by the interconnection and relative positioning constraints which "v" imposes on its components. Explicit consideration of these constraints can be avoided by merely determining the snapshot set of a simple or composite scene view from encodements made with the entire scene view intact.

4.3 Snapshot Signatures

Note that it is possible for two distinct scene views to have the same snapshot sets relative to some encodement resolution. This is illustrated in Figure 4.4, where the snapshot sets for both a triangle and a left-right symmetrical trapezoid are shown relative to an encodement resolution which makes their base lengths 2 grid units, their altitudes 1 grid unit, and the top length of the trapezoid $\frac{1}{2}$ grid unit. Both of the views of these objects shown in Figure 4.4 have the same 7 snapshots in their snapshot sets. Such an example demonstrates that snapshot sets alone are inadequate to completely characterize scene views relative to some encodement resolution, since a single snapshot set may be generated by two (or more) distinct scene views at that resolution.

A more complete characterization of a scene view "v" in terms of its grid encodements at resolution r is

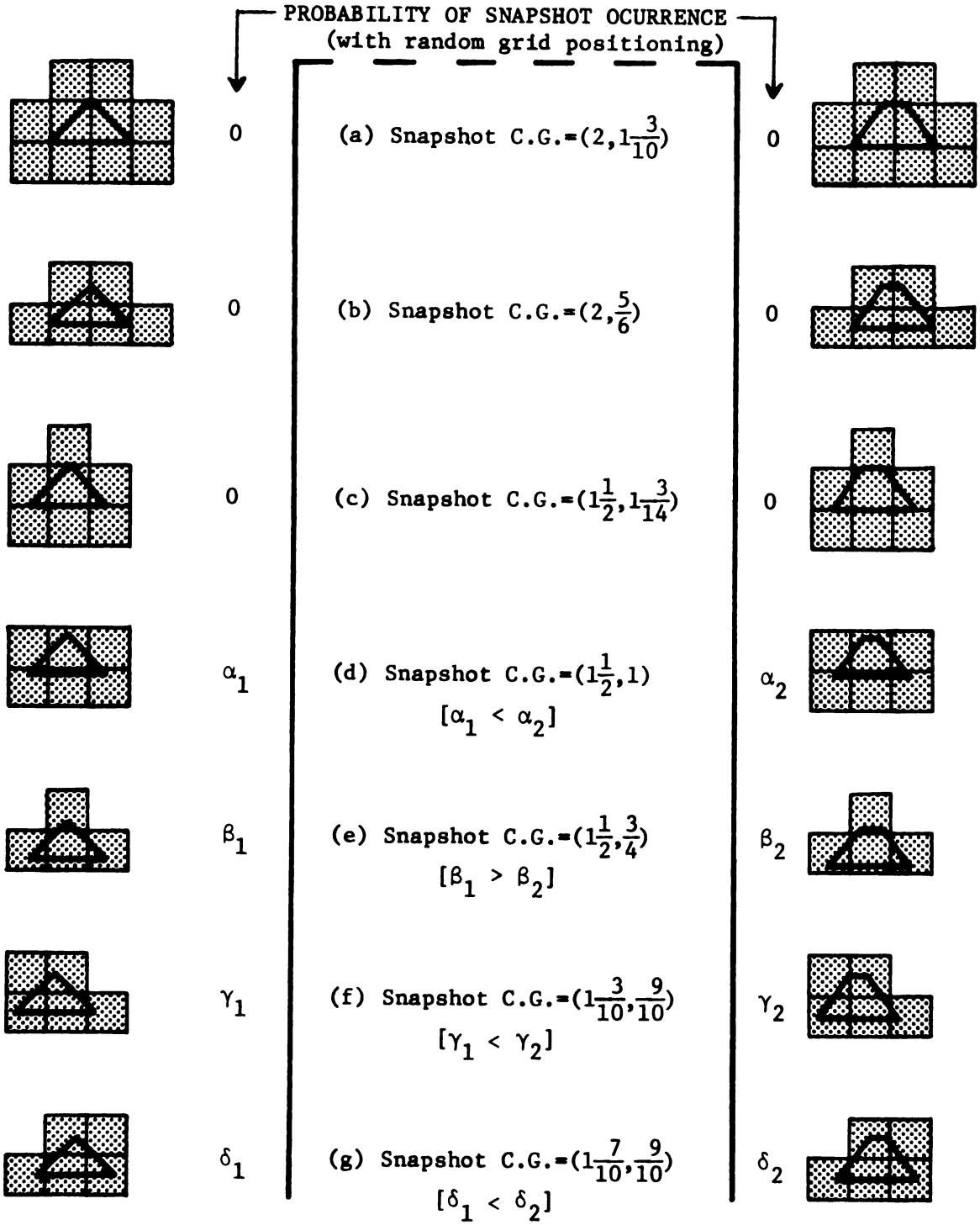


Figure 4.4. Example of two object views with the same snapshot set, but different snapshot signatures.

possible by associating a probability distribution with the snapshot set S -- where $S = \{s_1, s_2, \ldots, s_n\}$ -- of "v" relative to resolution r. This probability distribution is realized by a probability function "p" which assigns to each snapshot $s_i \varepsilon S$ a value "$p(s_i)$" which represents the probability that s_i will be the snapshot encoded as a result of some random positioning of an encodement grid of resolution r over the scene view "v". We will call "$p(s_i)$" as defined above the "probability of occurrence of snapshot s_i" relative to a specified scene view and encodement resolution.

Definition 4.2

The snapshot signature of scene view "v" relative to encodement resolution "r" consists of the snapshot set of "v" relative to "r" along with an associated probability distribution which defines the probability of occurrence (relative to "r") of each snapshot in the snapshot set of "v".

Examples of two snapshot signatures are illustrated in Figure 4.4. The snapshot signature of the triangle relative to the encodement resolution is given by the snapshot set {(a),(b),(c),(d),(e),(f),(g)} with the associated probability distribution: $p((a))=p((b))=p((c))=0$, $p((d))=\alpha_1$, $p((e))=\beta_1$, $p((f))=\gamma_1$, and $p((g))=\delta_1$. The snapshot signature of the trapezoid relative to the encodement

resolution is given by the snapshot set {(a),(b),(c),(d), (e),(f),(g)} with the associated probability distribution: $p((a))=p((b))=p((c))=0$, $p((d))=\alpha_2$, $p((e))=\beta_2$, $p((f))=\gamma_2$, and $p((g))=\delta_2$. The probability of occurrence of each of the snapshots (a) and (c), for both the triangle and trapezoid, is 0 because these snapshots are generated only when the base of the figure lies exactly along a horizontal grid line. Such a result has probability 0 of occurring with some random positioning of the encodement grid over either figure. Similarly, the probability of occurrence of snapshot (b) for either the triangle or trapezoid is 0 because the snapshot is generated only when both endpoints of the base of the figure just touch two vertical grid lines -- again a result which has probability 0 of occurring with some random positioning of the encodement grid over either figure. Now snapshot (d) is produced for either the triangle or the square whenever a horizontal line of the encodement grid divides the figure into an upper and lower part, and each part intersects exactly two vertical grid lines. Such a result occurs with non-zero probability whenever an encodement grid is randomly positioned over either figure. Further, notice that if a horizontal grid line divides both the triangle and trapezoid at the same vertical level with respect to their bases, then the bottom parts of both figures will have the same maximum horizontal width, while the top part of the triangle will have a maximum horizontal width which is

less than the maximum horizontal width of the top part of the trapezoid. Hence, for equivalent horizontal positionings (with respect to vertical grid lines) of the bases of both figures, there are less vertical positionings (with respect to horizontal grid lines) for the triangle than there are for the trapezoid which produce snapshot (d). So it follows that the probability of snapshot (d) occurring for some random positioning of the grid with respect to the figure is less for the triangle than for the trapezoid -- i.e., $\alpha_1 < \alpha_2$. Similar considerations, based on the fact that the top of the trapezoid is wider than the top of the triangle, result in the conclusions that $\beta_1 > \beta_2$, $\gamma_1 < \gamma_2$, and $\delta_1 < \delta_2$. Thus, the snapshot signatures of the triangle and trapezoid relative to the encodement resolution shown in Figure 4.4 are distinct, even though their snapshot sets are identical at that encodement resolution. This demonstrates that snapshot signatures -- in contrast with snapshot sets -- represent a more complete characterization of scene views in terms of their grid encodements. This snapshot signature characterization of scene views will prove useful when scene views are to be recognized from their encodements at coarse resolutions.

The determination of the theoretical probability distribution for the snapshot signature of a scene view "v" relative to some encodement resolution "r" will now be discussed. We first make a definition of terminology.

Definition 4.3

A unit region relative to resolution r is the set of all points contained in a square-bordered region of area 1 on the grid of resolution r.

A standard unit region relative to resolution r is a unit region relative to resolution r which has its borders parallel to the grid lines (i.e., horizontal and vertical).

Let "p" denote a reference point on scene view "v". Then each relative translational positioning of v with respect to the encodement grid of resolution r is characterized by exactly one relative location of p with respect to the pattern of horizontal and vertical grid lines of the encodement grid of resolution r. In turn, each such location of p relative to the pattern of grid lines at resolution r is characterized by exactly one of the points contained in an arbitrary standard unit region "R" on the encodement grid of resolution r. Hence a random positioning of the encodement grid of resolution r over the scene view v is equivalent to the random selection of a location for p from R. Note that the points of R are associated with the snapshot set "S" of v relative to resolution r by the function "h" defined as follows: for $p_o \varepsilon R$, $h(p_o) \varepsilon S$ such that $h(p_o)$ represents the snapshot of v that would be encoded on the grid of resolution r with v positioned so that its reference point p is at location

p_o. Thus $S = \{h(p_o) | p_o \varepsilon R\}$. Now each snapshot in S can be characterized in terms of the set of points $R_i \subseteq R$ which are associated with it under "h" -- i.e., for $S=\{s_1, s_2, \ldots, s_n\}$, $R_i = \{p_o | p_o \varepsilon R \text{ and } h(p_o) = s_i\}$ for $i=1,2,\ldots,n$. Further, each R_i can be interpreted as defining a subregion (not necessarily connected) of R having an area (interpreted relative to the real Cartesian plane) denoted by "$A(R_i)$", for $i=1,2,\ldots,n$. Analogously, let "A(R)" denote the area of region R, so that A(R)=1 (recall R is a standard unit region). Then the probability that $h(p_o)=s_i$ for a randomly chosen $p_o \varepsilon R$ is just the probability that $p_o \varepsilon R_i$, which is given by

$$\frac{A(R_i)}{A(R)} = \frac{A(R_i)}{1} = A(R_i) \text{ , for } i=1,2,\ldots,n.$$

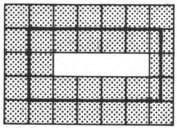
But the probability that $p_o \varepsilon R_i$ is just the probability of occurrence of snapshot s_i -- i.e., $p(s_i)$ -- relative to scene view v and resolution r. Hence, the snapshot signature of "v" relative to r is given by the snapshot set

$$S = \{h(p_o) | p_o \varepsilon R\} = \{s_1, s_2, \ldots, s_n\}$$

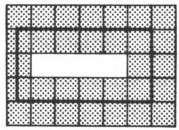
with associated probability distribution

$$p(s_i) = A(R_i) \text{ , for } i=1,2,\ldots,n.$$

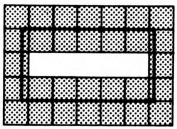
Figure 4.5 illustrates the theoretical snapshot signature of a rectangle relative to an encodement grid whose resolution makes the rectangle 3 grid units high and $5\frac{1}{2}$ grid units wide. Its snapshot signature is given by snapshot set $S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$ with associated



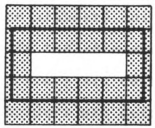
Snapshot s_1
$p(s_1) = 0$



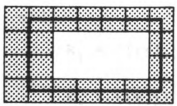
Snapshot s_2
$p(s_2) = 0$



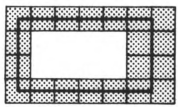
Snapshot s_3
$p(s_3) = 0$



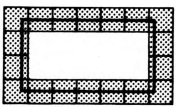
Snapshot s_4
$p(s_4) = 0$



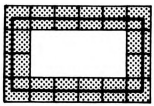
Snapshot s_5
$p(s_5) = 0$



Snapshot s_6
$p(s_6) = 0$



Snapshot s_7
$p(s_7) = \frac{1}{2}$



Snapshot s_8
$p(s_8) = \frac{1}{2}$

Figure 4.5. Snapshot signature of a 3 x $5\frac{1}{2}$ grid unit rectangle.

probability distribution

$$p(s_k)=0 \text{ for } k=1,2,\ldots,6, \text{ and } p(s_7)=p(s_8)=\tfrac{1}{2}.$$

The determination of this probability distribution can be illustrated by picking some reference point p on the rectangle, and some standard unit region R on the encodement grid. Let the upper left corner of the rectangle be reference point p. Also, let R be the standard unit region defined by one of the squares in the encodement grid[3] -- say the one whose vertical borders are defined by x=i and x=i+1, and whose horizontal borders are defined by y=j and y=j+1. Then

$$R = \{(x,y) \mid i<x\leq i+1 \text{ and } j\leq y<j+1\},$$

and

$$R_1 = \{(i+1,j)\} \qquad R_2 = \{(i+\tfrac{1}{2},j)\}$$

$$R_3 = \{(x,j) \mid i+\tfrac{1}{2}<x<i+1\}$$

$$R_4 = \{(x,j) \mid i<x<i+\tfrac{1}{2}\}$$

$$R_5 = \{(i+1,y) \mid j<y<j+1\}$$

$$R_6 = \{(i+\tfrac{1}{2},y) \mid j<y<j+1\}$$

---

[3] Only one vertical and one horizontal border line are considered as part of a standard region R, which insures that each point in R characterizes a unique relative positioning with respect to the corresponding grid. In this case, we choose the lower and right borders of the grid square to be included in R. Points on the left and top borders of the grid square are then not elements of R.

$$R_7 = \{(x,y) \mid i+\tfrac{1}{2}<x<i+1 \text{ and } j<y<j+1\}$$

$$R_8 = \{(x,y) \mid i<x<i+\tfrac{1}{2} \text{ and } j<y<j+1\}.$$

Each of these subregions R_k, k=1,2,...,8 are illustrated in Figure 4.6 as either a point, line segment, or shaded area, relative to a dashed square-bordered region -- which includes the lower and right border lines, but not the upper and left border lines -- which represents R. With A(R)=1, note that

$$A(R_1)=A(R_2)=A(R_3)=A(R_4)=A(R_5)=A(R_6)=0,$$

while

$$A(R_7)=A(R_8)=\tfrac{1}{2}.$$

If the upper left shaded grid square of each snapshot shown in Figure 4.5 is interpreted as region R, then the rectangle shown in each snapshot s_k has its upper left corner point p in a representative location from R_k, for k=1,2,...,8.

This theoretical procedure for determining the probability distribution associated with the snapshot set of a scene view can become extremely difficult to apply -- especially at high encodement resolutions -- for scene views having greater structural complexity than the rectangle of Figure 4.5. The subregion of a standard unit region which is associated with a particular snapshot may be odd shaped or even disconnected, and hence quite hard to identify. Computing the area of such a subregion may

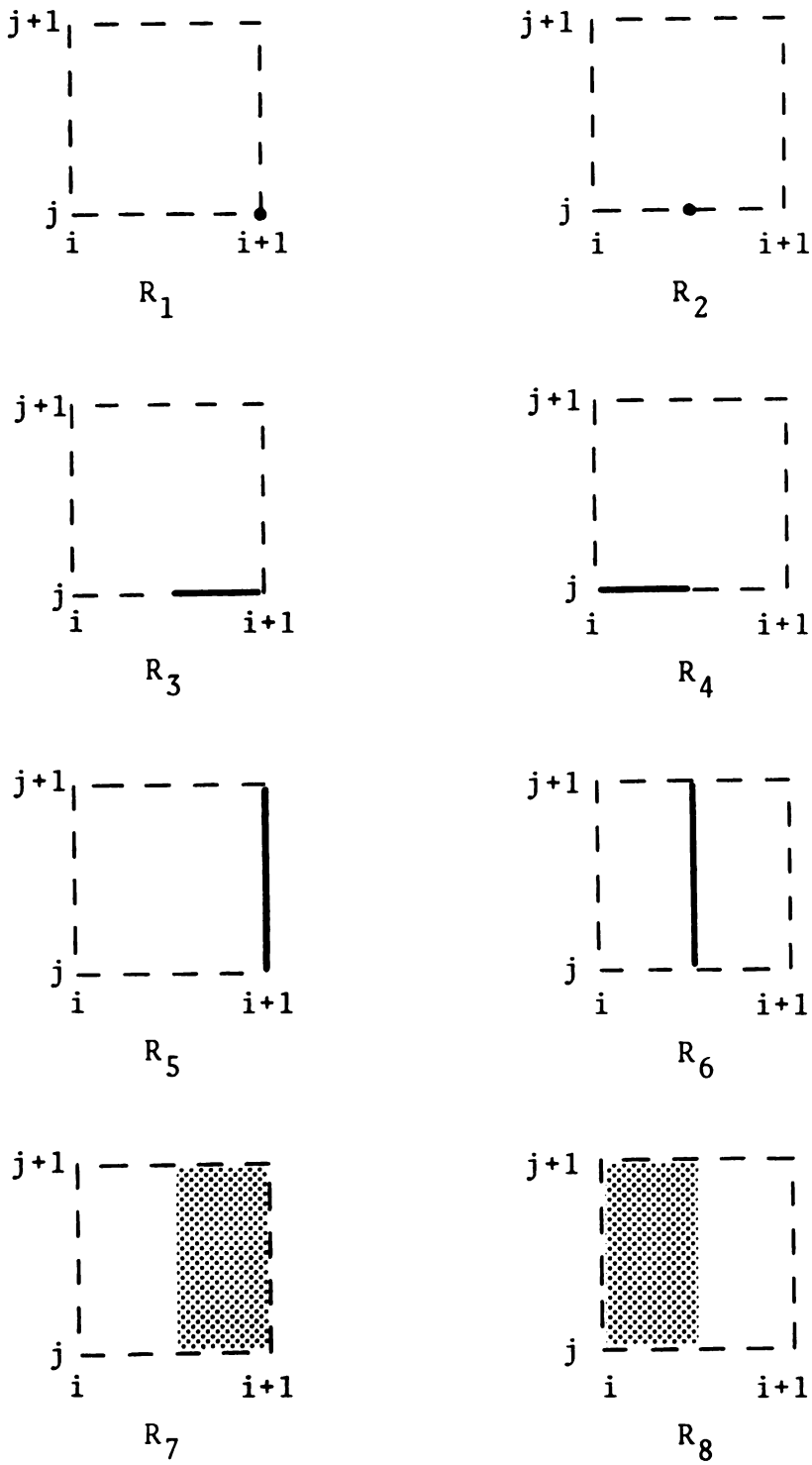


Figure 4.6. The eight subregions of R which define the probability distribution for the snapshot signature of the rectangle in Figure 4.5.

likewise be difficult. Fortunately, the snapshot set approximation procedure discussed in Section 4.2 provides a practical method for approximating the probability distribution of the snapshot set, as well as the snapshot set itself, in such cases. Let r, v, v_o, D, V, and V' be as defined for the snapshot set approximation procedure of Section 4.2. Note that when scene view v is considered in all its T-F positions defined by set V, any reference point p on v can assume precisely those locations in some standard unit region R on the encodement grid of resolution r. Thus, considering some randomly chosen finite subset V' of V is equivalent to considering some randomly chosen finite subset R' of points from R, where V' and R' have the same number of elements, say n. So it is reasonable to expect that the relative frequency with which a given snapshot occurs in the collection of encodements of the elements of V' would provide a useful approximation to the probability of occurrence of that snapshot for scene view v at encodement resolution r -- provided that n is sufficiently large. This forms the basis of the following procedure for approximating the snapshot signature of a scene view "v" relative to encodement resolution r:

(1) apply the modified "linguistic" algorithm (the "linguistic" algorithm of Chapter 3 with rules (I)-(a) and (I)-(b) replaced by rules (I)-(a)', (I)-(b)' and (I)-(c)', as described in Section 4.2) n times -- where n is a large, but finite,

integer number -- to generate a collection of n random encodements of "v" at resolution r;

(2) using the effective procedure of Theorem 3.5, determine the set $S=\{s_1,s_2,\ldots,s_q\}$ of distinct snapshots which are represented in the n encodements, and the number n_i of those n encodements which represent each of the snapshots s_i, for $i=1,2,\ldots,q$;

(3) determine f_i, the relative frequency of occurrence of the snapshot s_i in the n encodements, by $f_i = \frac{n_i}{n}$ for $i=1,2,\ldots,q$;

and (4) approximate the snapshot signature of v relative to r by the snapshot set $S=\{s_1,s_2,\ldots,s_q\}$ with the associated probability distribution $p(s_i)=f_i$, for $i=1,2,\ldots,q$.

Again, the larger the value of n, the better this approximation will be.

4.4 The Utility of Snapshot Signatures

As noted in the preceding section, snapshot signatures have the capability to characterize scene views at encodement resolutions which are inadequate to produce structurally different encodements of distinct scene views. For if an encodement resolution r is such that a set of two (or more) distinct scene views have the same snapshot set relative to r, the scene views can still be distinguished at resolution r if each scene view associates a

unique probability distribution with that snapshot set. This characterization capability of snapshot signatures will form the basis for the scene view recognition approach described in the remainder of this chapter.

4.4.1 Scene View Recognition: The Comparison of Snapshot Signatures

Consider a scene recognition situation in which there are only a finite number of scene views which can ever occur. These scene views will be called the prototypes, and will be denoted by the set $P=\{v_1,v_2,\ldots,v_k\}$. Further, for $i=1,2,\ldots,k$, let $v_i \varepsilon P$ have its theoretical snapshot signature relative to encodement resolution r given by (1) its snapshot set $S_i^r=\{s_{i,1},s_{i,2},\ldots,s_{i,t_i}\}$ relative to resolution r with (2) associated probability distribution $p_i^r(s_{i,j})$ for $j=1,2,\ldots,t_i$. Now suppose an unknown scene view $v \varepsilon P$ is to be recognized based on a random sample determined from the following random experiment E^r: a grid of resolution r is superimposed over v in a random translational position, and v then encoded to produce snapshot e^r of v. An outcome e^r of E^r can be a snapshot in any snapshot set S_i^r for $i=1,2,\ldots,k$, since v may be any one of the elements of P, and hence the sample space S^r of E^r is given by

$$S^r=S_1^r \cup S_2^r \cup \ldots \cup S_k^r \quad .$$

If the experiment E^r is repeated independently n times, the random sample that is produced will be denoted by the

n-dimensional vector

$$(e_1^r, e_2^r, \ldots, e_n^r) ,$$

where $e_i^r \epsilon S^r$ denotes the outcome of the i th repetition of experiment E^r, for i=1,2,...,n. From such a random sample an approximation to the snapshot signature of v relative to r can be derived, where the approximation consists of snapshot set $\{e_1^r, e_2^r, \ldots, e_n^r\}$, with each distinct snapshot in this set assigned a probability of occurrence equal to its relative frequency of occurrence in the random sample. Then the problem of identifying v can be considered as the problem of identifying which of the known (theoretical or approximated) snapshot signatures (relative to r) of the scene views in P is the one represented by the approximate snapshot signature determined from the random sample. However, since the snapshot signature of v is only an approximation (it has been determined from a finite random sample), and also because one or more of the "known" snapshot signatures may also be an approximation (see Section 4.3), an exact match between the approximate snapshot signature of v and one of the "known" snapshot signatures will not necessarily occur. Hence, the unknown scene view v must then be identified as that element of P whose "known" snapshot signature most closely matches the approximate snapshot signature represented by the random sample. In this case, the particular criteria used to measure the "closeness" of a match between two snapshot signatures

will be a critical factor in determining the degree of certainty with which an identification of v is made. It is generally impossible to choose closeness criteria which insure that v will be identified correctly with 100% certainty, due to the fact that random samples and approximations are involved. However, it is possible to minimize or control the probability of making an identification error using standard statistical decision criteria. Such criteria can be effectively applied here as a measure of the degree of closeness between snapshot signatures (relative to the same resolution r) based upon their respective probability distributions. These decision criteria are the ones which arise in statistical hypothesis testing problems (see [H-4], [M-4], [W-3], or any standard text on statistics), so the scene view recognition problem will now be reformulated in that context.

4.4.2 Scene View Recognition: The Testing of Hypotheses

Consider the finite set P of prototype scene views, and a random sample generated by repeated application of experiment E^r with some unknown scene view $v \in P$. The identification of scene view v is considered in a hypothesis testing context by making two (or more) conjectures concerning the identity of v such that exactly one of the conjectures (hypotheses) must be true. For example, if the unknown scene view $v \in P$ is to be identified

precisely, the appropriate multiple hypotheses testing problem consists of testing k simple hypotheses H_i, where

$$H_i: \quad v=v_i \quad \text{for } i=1,2,\ldots,k.$$

Or if it is only necessary to determine whether or not scene view v represents a specific scene view $v_{i_o} \in P$, then the appropriate hypotheses testing problem consists of testing a simple hypothesis H_1 against a composite alternative H_2, where

$$H_1: \quad v=v_{i_o} \quad \text{and} \quad H_2: \quad v \neq v_{i_o}.$$

In any case, the relative validity of each hypothesis (conjecture) is tested using information from the random sample, and all but one of the hypotheses rejected. The single hypothesis accepted represents that conjecture about the identity of v which is most likely to be true, based on the particular criteria used for testing the relative validity of the hypotheses. One such criterion which is commonly used is the likelihood ratio, which we will discuss relative to k=2 (i.e., P contains just two scene views). Although this represents a relatively simple situation, it will suffice to demonstrate both the utility of snapshot signatures and the use of a standard statistical decision criterion in scene view recognition.

4.4.3 The Likelihood Ratio and Sequential Scene View Identification

In the case where k=2, the set P of prototype scene views is $\{v_1,v_2\}$. Thus, any hypotheses testing problem for determining the identity of v becomes a test between two simple hypotheses H_1 and H_2, where

$$H_1: \quad v=v_1 \quad \text{and} \quad H_2: \quad v=v_2.$$

The test is based on a random sample $(e_1^r,e_2^r,\ldots,e_n^r)$ derived from n independent repetitions of experiment E^r, and it is assumed that the snapshot signatures of v_1 and v_2 relative to resolution r are known. [Recall the notation defined in Subsection 4.4.1, where the symbols S_1^r, p_1^r, S_2^r, and p_2^r represent the theoretical snapshot set and its associated probability function relative to resolution r for v_1 and v_2, respectively, and where $S^r=S_1^r \cup S_2^r$ represents the sample space of E^r.] The likelihood ratio L_n can be used as the testing criterion (see [H-4] or [M-4]), where

$$L_n = \frac{\text{Probability of random sample } (e_1^r,e_2^r,\ldots,e_n^r) \text{ occurring if } H_2 \text{ is true}}{\text{Probability of random sample } (e_1^r,e_2^r,\ldots,e_n^r) \text{ occurring if } H_1 \text{ is true}}$$

$$= \frac{p_2^r(e_1^r)\cdot p_2^r(e_2^r)\cdot \ \ldots \ \cdot p_2^r(e_n^r)}{p_1^r(e_1^r)\cdot p_1^r(e_2^r)\cdot \ \ldots \ \cdot p_1^r(e_n^r)}$$

$$= \prod_{i=1}^{n} \frac{p_2^r(e_i^r)}{p_1^r(e_i^r)} \quad ,$$

and where

$$p_j^r(e_i^r) = 0 \quad \text{if} \quad e_i^r \notin S_j^r \quad \text{for } j=1,2 \text{ and } i=1,2,\ldots,n.$$

Intuitively, L_n gives the relative likelihood that H_2 is true as compared to H_1, for the given random sample. If $L_n=1$, both hypotheses can be considered equally likely to be true. If $0 \leq L_n < 1$, H_1 can be considered more likely to be true than H_2, with that likelihood being greater the closer the value of L_n is to 0. On the other hand, if $1 < L_n \leq \infty$, H_2 can be considered more likely to be true than H_1, with that likelihood being greater for larger values of L_n. With fixed sample size n, a threshold value c of L_n is picked such that the test accepts hypothesis H_1 as true if $L_n < c$ and rejects hypothesis H_1 (accepts hypothesis H_2) if $L_n > c$. Note that by making c larger, we can increase our certainty that the test will not reject H_1 when H_1 is true, but we also decrease our certainty that the test will not accept H_1 when H_1 is false. Putting this another way, by making c larger the probability that the test will reject H_1 when H_1 is true is made smaller, but the probability that the test will accept H_1 when H_1 is false is made larger. This reflects the two types of errors which must be considered in tests of hypotheses: (1) a type (I) error, which is made when the test rejects H_1 when H_1 is true; and (2) a type (II) error, which is made when the test accepts H_1 when H_1 is false. In conventional notation [M-4], p(I) is used to denote the probability of a

type (I) error occurring and p(II) is used to denote the probability of a type (II) error occurring, relative to a specified test. As noted above, it is only possible to control one of p(I) or p(II) if the likelihood ratio criterion is used relative to a fixed sample size, since a single value of c must be chosen. Generally, c is chosen such that p(I) is fixed at a certain acceptable value α (typically .05 or .01), which automatically assigns some value β to p(II). While it has been shown in the literature [M-4] that the likelihood ratio test minimizes the value of β (of p(II)) for a fixed value α (of p(I)) when a sample of (fixed) size n is being considered, the value β (of p(II)) may still be too large to be acceptable if this scheme were to be used for practical scene view identification.

However, in many practical applications involving scene view recognition, it is reasonable to assume that experiment E^r could be performed an arbitrary (but still finite) number of times on an unknown scene view v. For instance, if v was an unknown typewritten digit of a zip code, a mail sorter could reasonably have the capability to (rapidly) scan v an arbitrary number of times in attempting to identify the digit it represents. In such situations where the sample size n can be varied at will, it is possible to control both p(I) and p(II) at arbitrary pre-specified levels α and β, respectively. This is done by determining both a lower threshold $c_1<1$ and an upper

threshold $c_2>1$ for L_n such that the test accepts hypothesis H_1 with $p(II) = \beta$ if $L_n \leq c_1$, while the test accepts hypothesis H_2 (rejects H_1) with $p(I) = \alpha$ if $L_n \geq c_2$. If $c_1 < L_n < c_2$, then the sample size n is not yet large enough (assuming that the test is performed sequentially with L_1, $L_2, L_3, \ldots$, etc.) for the test to either reject or accept hypothesis H_1 with the required degree of certainty of not making either a type (I) or type (II) error. This test is performed in a sequential manner as given by the following sequence of rules:

(1) set i=1;

(2) make an observation e_i^r from experiment E^r;

(3) compute L_i, based on the random sample

$(e_1^r, e_2^r, \ldots, e_i^r)$;

(4) (a) if $L_i \leq c_1$, accept H_1 and terminate the test;

(b) if $L_i \geq c_2$, accept H_2 and terminate the test;

(c) if $c_1 < L_i < c_2$, increase i by 1 and repeat the test starting with step (2).

This test is known as the "sequential likelihood ratio test" [M-4], or alternatively as Wald's "sequential probability ratio test" (denoted "SPRT" in [W-3]). We now state several well-known properties of the above SPRT without proof (proofs can be found in [H-4], [M-4], [W-1], and [W-3]):

(a) simple and accurate approximations to the cutoff values c_1 and c_2 for L_n, with specified $p(I) = \alpha$ and $p(II) = \beta$, are given by

$$c_1 = \frac{\beta}{1-\alpha} \quad \text{and} \quad c_2 = \frac{1-\beta}{\alpha} \; ;$$

(b) the sample size n (i.e., the number of tests) required for the SPRT to terminate will be finite with probability 1, provided v_1 and v_2 have distinct probability distributions over the sample space $S^r = S_1^r \cup S_2^r$;

and (c) the approximate average sample size $E(n|v_i)$ for the SPRT when v_i is the actual scene view represented by v, for i=1,2, is

$$E(n|v_i) = \frac{P_f(v_i)\cdot\log(c_2)+(1-P_f(v_i))\cdot\log(c_1)}{E(z|v_i)},$$

where

$P_f(v_i)$ is the power of the test given by the probability that v_1 will be rejected when v_i is the actual scene view, so

$$P_f(v_i) = \begin{cases} \alpha & \text{for } i=1 \\ 1-\beta & \text{for } i=2 \end{cases},$$

and

$$E(z|v_i) = p_i^r(s_1^r)\cdot\log\frac{p_2^r(s_1^r)}{p_1^r(s_1^r)} + p_i^r(s_2^r)\cdot\log\frac{p_2^r(s_2^r)}{p_1^r(s_2^r)} +$$

$$\ldots + p_i^r(s_q^r)\cdot\log\frac{p_2^r(s_q^r)}{p_1^r(s_q^r)}$$

$$= \sum_{j=1}^{q} p_i^r(s_j^r) \cdot \log \frac{p_2^r(s_j^r)}{p_1^r(s_j^r)} ,$$

$$\text{for } S^r = S_1^r \cup S_2^r = \{s_1^r, s_2^r, \ldots, s_q^r\}$$

$$\text{with } p_i^r(s_j^r) = 0 \text{ if } s_j^r \notin S_i^r , \; j=1,2,\ldots,q.$$

The average sample size approximation $E(n|v_i)$, $i=1,2$, given in (c) above breaks down for the cases when $E(z|v_i)=0$ or when $E(z|v_i)$ doesn't exist (i.e., when $E(z|v_i)=\pm\infty$). However, on the average, the SPRT with $p(I)=\alpha$ and $p(II)=\beta$ will require a smaller sample size n for making a decision than would a non-sequential (fixed-sample size) likelihood ratio test which realizes the same $p(I)=\alpha$ and $p(II)=\beta$. And by (b), we know that this sample size n will be finite for the cases of interest. Hence, the SPRT provides a practical procedure for utilizing snapshot signatures in scene view recognition. This is illustrated in the two examples presented next.

4.4.4 Two examples

The expected sample size for the SPRT will be demonstrated for two examples in which an unknown scene view $v \varepsilon P = \{v_1, v_2\}$ is to be identified by testing the hypotheses

$$H_1: \; v=v_1 \quad \text{versus} \quad H_2: \; v=v_2$$

on the basis of a random sample from experiment E^r. In both examples, the resolution r will be chosen such that

v_1 and v_2 have the same snapshot set relative to r, but distinct probability distributions over that set. Further, it is assumed that p(I) and p(II) are set at acceptable levels α and β, respectively.

Example 1

Consider some arbitrary encodement resolution $r_o \geq 6$. Let v_1 be a rectangle with horizontal side length $5\frac{1}{2}$ grid units and vertical side length 3 grid units, and v_2 be a rectangle with horizontal side length $5\frac{1}{4}$ grid units and vertical side length 3 grid units -- all lengths being specified relative to r_o. The snapshot signature of v_1 is illustrated in Figure 4.5, and is given by snapshot set $S_1^{r_o}$ with associated probability distribution $p_1^{r_o}$, where

$$S_1^{r_o} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$$

with

$$p_1^{r_o}(s_i) = 0 \quad \text{for } i=1,2,\ldots,6, \text{ and}$$

$$p_1^{r_o}(s_7) = p_1^{r_o}(s_8) = \frac{1}{2} .$$

The snapshot signature of v_2 is given by snapshot set $S_2^{r_o}$ with associated probability distribution $p_2^{r_o}$, where

$$S_2^{r_o} = S_1^{r_o} = \{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$$

with

$$p_2^{r_o}(s_i) = 0 \quad \text{for } i=1,2,\ldots,6,$$

$$p_2^{r_o}(s_7) = \frac{1}{4} \text{ , and } p_2^{r_o}(s_8) = \frac{3}{4} .$$

Then the sample space S^{r_o} of experiment E^{r_o} is given by

$$S^{r_o} = S_1^{r_o} \cup S_2^{r_o} = \{s_1,s_2,s_3,s_4,s_5,s_6,s_7,s_8\}.$$

Now, the average size of the sample needed by the SPRT to identify v if v_1 is the actual scene view is given by:

$$E(n|v_1) = \frac{\alpha\cdot\log(\frac{1-\beta}{\alpha}) + (1-\alpha)\cdot\log(\frac{\beta}{1-\alpha})}{E(z|v_1)} ,$$

where

$$E(z|v_1) = \sum_{j=1}^{8} p_1^{r_o}(s_j)\cdot\log\frac{p_2^{r_o}(s_j)}{p_1^{r_o}(s_j)}$$

$$= 0 + 0 + 0 + 0 + 0 + 0 + \frac{1}{2}\cdot\log\frac{1/4}{1/2} + \frac{1}{2}\cdot\log\frac{3/4}{1/2}$$

$$= -.14384 .$$

For $\alpha = \beta = .05$,

$$E(n|v_1) = \frac{.05\ \log(19)+.95\ \log(\frac{1}{19})}{-.14384} = \frac{-.85000}{-.14384}$$

$$= 5.91 \simeq 6 .$$

Similarly, the average size of the sample needed by the SPRT to identify v if v_2 is the actual scene view is given by:

$$E(n|v_2) = \frac{(1-\beta)\cdot\log(\frac{1-\beta}{\alpha}) + \beta\cdot\log(\frac{\beta}{1-\alpha})}{E(z|v_2)} \quad ,$$

where

$$E(z|v_2) = \sum_{i=1}^{8} p_2^{r_o}(s_j)\cdot\log\frac{p_2^{r_o}(s_j)}{p_1^{r_o}(s_j)}$$

$$= 0 + 0 + 0 + 0 + 0 + 0 + \frac{1}{4}\cdot\log\frac{1/4}{1/2} + \frac{3}{4}\cdot\log\frac{3/4}{1/2}$$

$$= .13082 \quad .$$

For $\alpha = \beta = .05$,

$$E(n|v_2) = \frac{.95\ \log(19) + .05\ \log(\frac{1}{19})}{.13082} = \frac{.85000}{.13082}$$

$$= 6.48 \simeq 7 \quad .$$

Thus, for the identification of v with 95% certainty of being correct (i.e., the probability that v will be mis-identified is .05), the SPRT will require, on the average, 6 observations of v based on E^{r_o} if v_1 is the actual scene view and 7 observations from E^{r_o} if v_2 is the actual scene view.

If it is desired to identify v with a 99% certainty of being correct, the average sample size required by the SPRT can be determined, using $\alpha = \beta = .01$, as

$$E(n|v_1) = \frac{.01\ \log(99)+.99\ \log(\frac{1}{99})}{-.14384} = \frac{-4.5032}{-.14384}$$

$$= 31.4 \simeq 32 \quad ,$$

and

$$E(n|v_2) = \frac{.99\ \log(99)+.01\ \log(\frac{1}{99})}{.13082} = \frac{4.5032}{.13082}$$

$$= 34.3 \simeq 35 \quad .$$

As would be expected, the average sample sizes required by the SPRT are significantly larger if v is to be identified with a 99% certainty of being correct than if v is to be identified with a 95% certainty of being correct.

Example 2

A horizontal line "h" of length "a" on the standard screen will have length "ra" on the (extended) grid of resolution r (see Theorem 3.2). Then h can intersect either $\lfloor ra \rfloor$ or $\lfloor ra \rfloor + 1$ vertical grid lines at resolution r, causing either $\lfloor ra \rfloor + 1$ or $\lfloor ra \rfloor + 2$ consecutive horizontal grid squares, respectively, to be shaded in its encodement. Further, if h lies along a horizontal grid line at resolution r, its encodement will be 2 shaded grid squares high along its entire length; while if h lies between two horizontal grid lines at resolution r, its encodement will be exactly 1 shaded grid square high along its

entire length. Hence, the snapshot set S_h^r of h at resolution r consists of the 4 snapshots $s_{h,1}^r$, $s_{h,2}^r$, $s_{h,3}^r$, and $s_{h,4}^r$, where:

$s_{h,1}^r$ is the solid shaded grid square pattern which is 2 grid units high and $\lfloor ra \rfloor$ +1 grid units wide;

$s_{h,2}^r$ is the solid shaded grid square pattern which is 2 grid units high and $\lfloor ra \rfloor$ +2 grid units wide;

$s_{h,3}^r$ is the solid shaded grid square pattern which is 1 grid unit high and $\lfloor ra \rfloor$ +1 grid units wide;

and $s_{h,4}^r$ is the solid shaded grid square pattern which is 1 grid unit high and $\lfloor ra \rfloor$ +2 grid units wide.

Now if a grid of resolution r were randomly translationally positioned over h, (1) the probability that h would coincide with a horizontal grid line is "0" while the probability that h would lie between two horizontal grid lines is "1", and independently (2) the probability that h would cross $\lfloor ra \rfloor$ grid lines is "1 - (ra- $\lfloor ra \rfloor$)" while the probability that h would cross $\lfloor ra \rfloor$ +1 grid lines is "ra- $\lfloor ra \rfloor$ ". Hence, the probability of occurrence $p_h^r(s_{h,i}^r)$ of each snapshot $s_{h,i}^r$ at resolution r, for i=1,2,3,4, is given by

$$p_h^r(s_{h,1}^r) = p_h^r(s_{h,2}^r) = 0 ,$$

$$p_h^r(s_{h,3}^r) = 1 - (ra - \lfloor ra \rfloor) ,$$

and $p_h^r(s_{h,4}^r) = ra - \lfloor ra \rfloor$.

So, the snapshot signature of h relative to resolution r is given by snapshot set

$$S_h^r = \{s_{h,1}^r, s_{h,2}^r, s_{h,3}^r, s_{h,4}^r\}$$

with associated probability distribution p_h^r.

Let v_1 and v_2 be horizontal lines of length .91 and .98, respectively, on the standard screen. Then the snapshot signature of v_1 relative to resolution r is given by snapshot set

$$S_1^r = \{s_{1,1}^r, s_{1,2}^r, s_{1,3}^r, s_{1,4}^r\}$$

(which is just S_h^r with 1 substituted for h and .91 substituted for a)

with associated probability distribution p_1^r where

$$p_1^r(s_{1,1}^r) = p_1^r(s_{1,2}^r) = 0 ,$$

$$p_1^r(s_{1,3}^r) = 1 - (.91r - \lfloor .91r \rfloor) ,$$

and $p_1^r(s_{1,4}^r) = .91r - \lfloor .91r \rfloor$.

Also, the snapshot signature of v_2 relative to resolution r is given by snapshot set

$$S_2^r = \{s_{2,1}^r, s_{2,2}^r, s_{2,3}^r, s_{2,4}^r\}$$

(which is just S_h^r with 2 substituted for h and .98 substituted for a)

with associated probability distribution p_2^r where

$$p_2^r(s_{2,1}^r) = p_2^r(s_{2,2}^r) = 0 ,$$

$$p_2^r(s_{2,3}^r) = 1 - (.98r - \lfloor .98r \rfloor) ,$$

and $p_2^r(s_{2,4}^r) = .98r - \lfloor .98r \rfloor$.

Now note that $\lfloor .91r \rfloor = \lfloor .98r \rfloor$ for r=1,2,...,11, and $\lfloor .91r \rfloor \neq \lfloor .98r \rfloor$ for $r \geq 12$. This implies that v_1 and v_2 generate the same snapshot sets for r=1,2,...,11, but different snapshot sets for $r \geq 12$. We restrict our attention to values of r between 1 and 11 for the remainder of this example, to insure that the approximation formulas for the average sample size of the SPRT are applicable (for $r \geq 12$, $E(z|v_i)$ will not exist for either i=1 or i=2, or both). For $r \leq 11$, note that $S_1^r = S_2^r = \{s_1^r, s_2^r, s_3^r, s_4^r\}$, where $s_i^r = s_{1,i}^r = s_{2,i}^r$ for i=1,2,3,4. Hence, the sample space S^r of experiment E^r is given by

$$S^r = S_1^r \cup S_2^r = \{s_1^r, s_2^r, s_3^r, s_4^r\} \text{ for } r \leq 11.$$

Consider r = 2. Then the sample space S^2 of experiment E^2 is given by

$$S^2 = \{s_1^2, s_2^2, s_3^2, s_4^2\} ,$$

where

$$p_1^2(s_1^2) = p_1^2(s_2^2) = 0 ,$$

$$p_1^2(s_3^2) = 1 - (1.82 - \lfloor 1.82 \rfloor) = 1 - .82 = .18 ,$$

$$p_1^2(s_4^2) = 1.82 - \lfloor 1.82 \rfloor = .82 \quad ,$$

and

$$p_2^2(s_1^2) = p_2^2(s_2^2) = 0 \ ,$$

$$p_2^2(s_3^2) = 1 - (1.96 - \lfloor 1.96 \rfloor) = 1 - .96 = .04$$

$$p_2^2(s_4^2) = 1.96 - \lfloor 1.96 \rfloor = .96 \quad .$$

Then with $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_1 is the actual scene view is given by:

$$E(n|v_1) = \frac{.05 \log(19) + .95 \log(\frac{1}{19})}{E(z|v_1)} \quad ,$$

where

$$E(z|v_1) = \sum_{j=1}^{4} p_1^2(s_j^2) \cdot \log \frac{p_2^2(s_j^2)}{p_1^2(s_j^2)}$$

$$= 0 + 0 + .18 \log \frac{.04}{.18} + .82 \log \frac{.96}{.82}$$

$$= -.14147 \quad .$$

Hence,

$$E(n|v_1) = \frac{-.85000}{-.14147} \simeq 6 \quad .$$

Similarly, for $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_2 is the actual scene view is given by:

$$E(n|v_2) = \frac{.95 \log(19) + .05 \log(\frac{1}{19})}{E(z|v_2)} \quad ,$$

where

$$E(z|v_2) = \sum_{j=1}^{4} p_2^2(s_j^2) \cdot \log \frac{p_2^2(s_j^2)}{p_1^2(s_j^2)}$$

$$= 0 + 0 + .04 \log(\tfrac{.04}{.18}) + .96 \log(\tfrac{.96}{.82})$$

$$= .09116 \quad .$$

Hence,

$$E(n|v_2) = \frac{.85000}{.09116} = 9.33 \simeq 10 \quad .$$

Thus, for the identification of v at r=2 with 95% certainty of being correct (i.e., the probability that v will be mis-identified is .05), the SPRT will require, on the average, 6 observations of v based on E^2 if v_1 is the actual scene view and 10 observations from E^2 if v_2 is the actual scene view.

To examine the effect of increased resolution on the expected sample size, consider r=6. Then the sample space S^6 of experiment E^6 is given by

$$S^6 = \{s_1^6, s_2^6, s_3^6, s_4^6\}$$

where

$$p_1^6(s_1^6) = p_1^6(s_2^6) = 0, \; p_1^6(s_3^6) = .54, \; p_1^6(s_4^6) = .46,$$

and

$$p_2^6(s_1^6) = p_2^6(s_2^6) = 0, \; p_2^6(s_3^6) = .12, \; p_2^6(s_4^6) = .88 \; .$$

Then, with $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_1 is the actual scene view is given by

$$E(n|v_1) = \frac{-.85000}{E(z|v_1)} = \frac{-.85000}{.54 \log(\frac{.12}{.54}) + .46 \log(\frac{.88}{.46})}$$

$$= \frac{-.85000}{-.51380} = 1.66 \simeq 2 \quad .$$

Similarly, with $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_2 is the actual scene view is given by

$$E(n|v_2) = \frac{.85000}{E(z|v_2)} = \frac{.85000}{.12 \log(\frac{.12}{.54}) + .88 \log(\frac{.88}{.46})}$$

$$= \frac{.85000}{.39037} = 2.18 \simeq 3 \quad .$$

Thus, for the identification of v at r=6 with 95% certainty of being correct (i.e., the probability that v will be mis-identified is .05), the SPRT will require, on the average, 2 observations of v based on E^6 if v_1 is the actual scene view and 3 observations from E^6 if v_2 is the actual scene view.

For one additional examination of increased resolution effects on the expected sample size, consider r=10. Then the sample space S^{10} of experiment E^{10} is given by

$$S^{10} = \{s_1^{10}, s_2^{10}, s_3^{10}, s_4^{10}\}$$

where

$$p_1^{10}(s_1^{10}) = p_1^{10}(s_2^{10}) = 0, \; p_1^{10}(s_3^{10}) = .90,$$

$$p_1^{10}(s_4^{10}) = .10,$$

and

$$p_2^{10}(s_1^{10}) = p_2^{10}(s_2^{10}) = 0,\ p_2^{10}(s_3^{10}) = .20,$$

$$p_2^{10}(s_4^{10}) = .80.$$

Then, with $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_1 is the actual scene view is given by

$$E(n|v_1) = \frac{-.85000}{E(z|v_1)} = \frac{-.85000}{.90 \log(\frac{.20}{.90}) + .10 \log(\frac{.80}{.10})}$$

$$= \frac{-.85000}{-1.14572} = .74 \simeq 1 \quad .$$

Similarly, with $\alpha = \beta = .05$, the average size of the sample needed by the SPRT to identify v if v_2 is the actual scene view is given by

$$E(n|v_2) = \frac{.85000}{E(z|v_2)} = \frac{.85000}{.20 \log(\frac{.20}{.90}) + .80 \log(\frac{.80}{.10})}$$

$$= \frac{.85000}{1.36274} = .622 \simeq 1 \quad .$$

Thus, for the identification of v at r=10 with 95% certainty of being correct, the SPRT will require, on the average, 1 observation of v based on E^{10}. While this result may seem somewhat surprising, it should be remembered that $E(n|v_i)$ represents an <u>average</u> value for n, based on n being continuous. In reality of course, n is discrete and can only assume values 1,2,3,...,etc., and the approximations for $E(n|v_i)$ must be interpreted with this in mind.

This example confirms that as the encodement resolution r is increased, the average number of samples required by the SPRT to identify v (as either v_1 or v_2) decreases. This is intuitively appealing, since one would expect that distinguishing a length difference (which is the only difference between v_1 and v_2) would be easier at resolutions for which that length difference is closer to 1 grid unit than at coarser resolutions for which that length difference is closer to 0 grid units.

4.4.5 Practical Considerations

The two examples of the previous section demonstrate that snapshot signatures can be used practically for scene view identification in situations where distinct scene views generate the same snapshot sets relative to some encodement resolution. Such cases occur when the resolution of the encodement grid is too coarse to reflect the structural differences of the distinct scene views, regardless of the particular relative translational position of the encodement grid. Hence, features which are based on the structure reflected in an arbitrary encodement of an unknown one of these distinct scene views -- even after smoothing or other image enhancement techniques -- cannot provide any information which can help to identify which one of these scene views is represented. This implies that the "classical" pattern

recognition approach based on feature extraction from a single encodement of an unknown picture will not be productive at such coarse resolutions. Hence, the scene view identification scheme based on snapshot signatures can be utilized at coarse encodement resolutions where "classical" feature extraction techniques of identification cannot be productively applied. This is particularly significant when physical constraints -- such as the size or number of photocells which may be available to realize a grid (see Section 2.3) -- may prevent the encodement grid resolution from being made fine enough for feature extraction identification techniques (based on a single encodement of an unknown) to be effective.

Even when the resolution to be used for encoding an unknown scene view can be made fine enough to allow feature extraction identification techniques to be used productively, it may still be advantageous to use a relatively coarse resolution with an identification scheme based on snapshot signatures. The choice is based on memory vs. time considerations, and also upon whether or not it is possible -- or desirable -- to encode an unknown scene view more than once. Of course, if an unknown scene view is only scanned (encoded) once, snapshot signature identification techniques can only be based on the probability of occurrence which each prototype scene view associates with that single snapshot -- a

procedure which does not fully utilize the identification capabilities of snapshot signatures. Hence, we will restrict our attention to those cases where the unknown scene view to be identified can be scanned as many times as desired (with the encodement grid randomly and independently translationally positioned for each scan). Note that each such scan of the unknown scene view at resolution r generates a binary encodement with $(r+2)^2$ bits of information -- one bit representing each (shaded or non-shaded) square on the extended grid of resolution r. Hence, if an identification procedure requires n scans (encodements) of an unknown scene view at resolution r, then $(r+2)^2 \cdot n$ total bits of information must be processed. For Example 2 of Subsection 4.4.4, we can thus draw the following conclusions about the average number of bits of information which are needed to identify an unknown $v \in \{v_1, v_2\}$ using the SPRT:

(a) for r = 2, the average number of bits to be processed is

$$(2+2)^2 \cdot E(n|v_1) \simeq 16 \cdot 6 = 96 \quad \text{if } v_1 \text{ is the actual scene view,}$$

and is

$$(2+2)^2 \cdot E(n|v_2) \simeq 16 \cdot 10 = 160 \quad \text{if } v_2 \text{ is the actual scene view;}$$

(b) for r = 6, the average number of bits to be processed is

$$(6+2)^2 \cdot E(n|v_1) \simeq 64 \cdot 2 = 128 \quad \text{if } v_1 \text{ is the actual scene view,}$$

and is

$$(6+2)^2 \cdot E(n|v_2) \simeq 64 \cdot 3 = 192 \quad \text{if } v_2 \text{ is the actual scene view;}$$

and (c) for r = 10, the average number of bits to be processed is

$$(10+2)^2 \cdot E(n|v_i) \simeq 144 \cdot 1 = 144 \quad \text{for } i=1,2.$$

Now the first resolution in Example 2 of Subsection 4.4.4 for which v_1 and v_2 can generate a different snapshot is r = 12. Hence, this is the first resolution at which a feature extraction identification scheme based on a single encodement of an unknown $v \varepsilon P$ can be effective. At r = 12, v_1 can generate snapshots which are 11 and 12 shaded grid squares wide, while v_2 can generate snapshots which are 12 and 13 shaded grid squares wide. Hence, at least a 13 x 13 grid (having grid square size equal to that of the grid of resolution 12) is needed to represent an arbitrary encodement of an unknown scene view $v \varepsilon \{v_1, v_2\}$. In turn, this implies that at least $(13)^2 \cdot 1 = 169$ bits of information must be processed for the identification of v using feature extraction techniques. Note, however, that at r = 12 the snapshots generated by both v_1 and v_2 which are

12 shaded grid squares in length are identical. And the probability that such snapshot occurs is .92 (which is $p_1^{12}(s_4^{12})$) if v_1 is the actual scene view and .24 (which is $p_2^{12}(s_3^{12})$) if v_2 is the actual scene view. Hence, the chance at r = 12 that a random encodement of an unknown $v\varepsilon\{v_1,v_2\}$ will reflect a structural characteristic which can be used to differentiate between v_1 and v_2 is fairly small. So the probability is fairly high that a feature extraction technique which uses structural characteristics of the encodement to identify v will make an error. In fact, it is not until r = 23 -- where v_1 can generate only snapshots which are 21 and 22 grid units wide, and v_2 can generate only snapshots which are 23 and 24 grid units wide -- that an arbitrary encodement of an unknown $v\varepsilon\{v_1,v_2\}$ will always reflect the structural differences between v_1 and v_2. At r = 23, feature extraction techniques could be used to identify v as either v_1 or v_2 (with 100% certainty of being correct) by processing the $(23+1)^2 \cdot 1 = 576$ bits of information in any arbitrary encodement of v. In any event, the feature extraction method of identifying an unknown $v\varepsilon\{v_1,v_2\}$ from a single arbitrary encodement requires the processing of at least 169 bits of information (at r = 12) to be at all effective, and even more (for r > 12) if that identification is to be made with a high degree of certainty. Comparing these requirements with the average number of bits required to

identify v with 95% certainty using the SPRT with snapshot signatures at lower resolutions r (r = 2,6, and 10), the feature extraction method will in many cases require significantly more bits to be processed to identify v with the same level of certainty. To make an exact comparison, it would be necessary to consider the probabilities with which the scene views v_1 and v_2 themselves occur, the degree of certainty with which an unknown scene view is to be identified, and the specific features and decision rule used by the feature extraction method. However, it should be apparent from this discussion that the use of snapshot signatures with the SPRT offers an effective alternative to the "classical" feature extraction methods (based on a single arbitrary encodement of an unknown) of identification. Not only can the identification be done at lower resolutions using snapshot signatures, but there may also be an overall savings in the average number of bits of information (i.e., memory) needed for identifying unknown scene views.

The smaller average number of bits which may be required by the SPRT to identify an unknown scene view is not the only savings in memory which can result from using snapshot signatures. Feature extraction methods require a relatively fine resolution, say r_1, to insure that significant structural differences will be reflected in an arbitrary encodement of any of the scene views under

consideration. And since the entire encodement must be available for extracting features, $(r_1+1)^2$ bits of information must be simultaneously accessible (i.e., stored in memory). On the other hand, the use of snapshot signatures and the SPRT may permit a relatively coarse resolution, say $r_o(<r_1)$, to be used for identifying an unknown one of the scene views under consideration. And since the SPRT only requires that one encodement at a time be available, only $(r_o+2)^2$ bits of information must be simultaneously accessible (i.e., stored in memory). If r_o is significantly smaller than r_1, a sizeable savings in required memory is thus possible using the SPRT at the coarser resolution. Of course, considerations regarding memory savings must be weighed against any increase in time which may be involved in taking a number of encodements (at r_o) instead of just a single encodement (at r_1).

Although both the examples of the previous section considered prototype scene views which differed only in some length measurement, the methods that were illustrated apply equally as well when the prototype scene views have other structural differences (e.g., shape, or overall size). Additionally, the use of snapshot signatures with the SPRT to identify an unknown scene view does not require that the prototype scene views have the same snapshot set. In fact, the expected sample size needed by the SPRT to identify an unknown one of prototype scene views

having distinct snapshot sets may be quite small (e.g., if one prototype generates a snapshot with relatively high probability, while the other prototype cannot generate that snapshot at all). Finally, the method of using snapshot signatures with the SPRT to identify an unknown one of two prototype scene views can be naturally extended to the case where there are more than two prototypes by using the "generalized sequential probability ratio test" (denoted "GSPRT") [F-4] with snapshot signatures. Of course, any other statistical method of testing multiple or composite hypotheses could also be used for scene view identification in such cases, as mentioned in Subsections 4.4.1 and 4.4.2.

It should be noted that the use of snapshot signatures with the SPRT (or GSPRT) in scene view identification might advantageously be combined with other pattern recognition methods in a picture recognition situation. For instance, a feature extraction method might be used on an arbitrary encodement of an unknown scene view v at some given resolution to eliminate part of the prototype scene views from further consideration. Then additional encodements of the unknown scene view could be made (not necessarily at the same resolution) and snapshot signatures with the SPRT (or GSPRT) used to identify v as one of the remaining prototypes. The effectiveness of such a procedure would obviously depend on the resolution constraints, the degree of certainty with which an identification is to

be made, the features and decision rule used in the feature extraction, and the decrease in the average sample size that could be expected by using a sequential test on a reduced set of prototypes in that particular picture recognition situation.

4.4.6 Blurry Images

An interesting interpretation of the snapshot signature of a scene view relative to resolution r results when the snapshots in the snapshot set are superimposed one over the other to form a single composite representation of that scene view. In such a superimposition, the probability of occurrence associated with each of the snapshots can be used to represent an intensity level for the shading associated with its grid squares. Then, any point (or region) in the single composite representation of the scene view can be assigned an overall intensity level which is the sum of the shading intensities of all the snapshots which cover that point (or region) in the superimposition.

Such a single composite representation -- which will be denoted as a "blurry image" -- is illustrated in Figure 4.7 for both of the snapshot signatures in Figure 4.4. The snapshots shown in Figure 4.4 have been superimposed with respect to their centers of gravity (denoted "C.G.", and defined relative to the lower left corner of each snapshot in Figure 4.4), which is represented by the

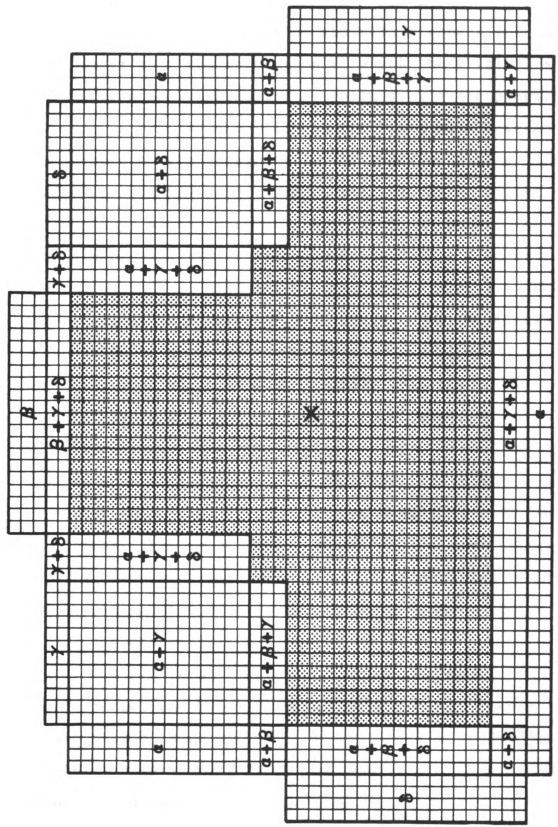


Figure 4.7. Blurry image represented by the snapshot signature of Figure 4.4.

crossing point of the symbol "X" appearing in the center of Figure 4.7. The resolution shown in Figure 4.7 is 20 times finer than the resolution at which the original snapshots were shown in Figure 4.4, which avoids the problem of having a grid square which is only partially covered by one or more of the snapshots in the superimposition. The shaded area of Figure 4.7 represents the region of the blurry image which is covered by all the snapshots (having non-zero probability of occurrence) of Figure 4.4, and hence has shading intensity "1" (= $\alpha + \beta + \gamma + \delta$). The other regions in Figure 4.7 are labeled with their shading intensities by a sum of one or more of α,β,γ, or δ, depending upon which of the snapshots (d), (e), (f), and/or (g) (in Figure 4.4), respectively, cover that region in the superimposition. The subscripts on α,β,γ, and δ have been dropped so that Figure 4.7 can represent the blurry image of either object in Figure 4.4, depending upon which set of probabilities -- $\alpha_1,\beta_1,\gamma_1$, and δ_1 for the triangle, or $\alpha_2,\beta_2,\gamma_2$, and δ_2 for the trapezoid -- is used to define the shading intensity levels. Since the shading intensities associated with snapshots (a), (b), and (c) of Figure 4.4 are zero, these snapshots do not affect the blurry image, and hence are omitted from consideration. Notice that by superimposing the snapshots with respect to their centers of gravity, the symmetry of the original scene view appears to be preserved in the high intensity

(shaded) region of the blurry image, with some of its significant structural form (e.g., wide base, narrow top) also preserved in this region.

The blurry image can be considered as a photographic "multiple exposure" which is taken of a scene view relative to some encodement resolution r. Theoretically, the blurry image can be considered as the result of taking "snapshots" of the scene view with the encodement grid (photographic film with a "coarse grain") in every possible relative translational position with respect to the scene view, and then reproducing all these snapshots (positioned so that their C.G.'s line up) on the same photographic print (with a fine grain for good resolution) to form a single "multiple-exposure" image. The relative shading intensities in the various regions of this single image would then proportionately reflect the shading levels of the corresponding regions of the blurry image.

Practically, a blurry image can be approximated by taking a finite sample of n "snapshots" of the scene view with the encodement grid in n independent, randomly chosen relative translational positions with respect to the scene view. These snapshots can then be superimposed with respect to their C.G.'s to form a single composite image, with each point in this image assigned a shading intensity level equal to the relative number of (i.e., the fraction of) the n snapshots which cover that point. Then the

entire scene view recognition problem can be viewed as a "template matching" problem [D-2] between the approximate blurry image of an unknown scene view and the theoretical blurry images of the prototype scene views. And effectively, this template matching is what the identification procedure using the SPRT with snapshot signatures accomplishes, sequentially increasing the number of snapshots n in the random sample until the blurry image approximation for the unknown scene view matched one of the prototype blurry images (templates) closely enough to make an identification with the required level of certainty.

Chapter 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Summary and Conclusions

The recognition of scenes from their encodements is the central problem throughout the fields of pattern recognition and picture processing. The various techniques which have been used to approach this problem range from deterministic and statistically based decision-theoretic methods to the formal structural analysis methods of syntactic (linguistic) picture processing. All such approaches, however, are forced to deal with encodement images of actual scenes, which represent some level of degradation of the structural features of these scenes. Such approximate or inexact representations result from the finite discretization of scenes which is done to facilitate automatic processing by machines. And physical constraints which may restrict the resolutions with which a scene can be encoded may further amplify the distortion effects of the discretation, particularly when the maximum possible encodement resolution is still too coarse to reflect significant fine structural detail in the scenes which are to be considered.

Traditionally, pattern recognition methods have dealt with a single arbitrary encodement of scenes in identification situations. However, such an approach is not always effective, especially when the encodement resolution is relatively coarse and the structural form of an encodement highly dependent upon the relative positioning between the encodement grid and scene. For such situations, an alternative approach is suggested in this thesis which is based on characterizing a scene view (a scene in fixed orientation) relative to any encodement resolution in terms of all the distinct encodement patterns (individually called "snapshots", or collectively the "snapshot set") which can result by encoding that scene view in various (arbitrary) relative translational positions with respect to the encodement grid. Additionally, this characterization associates with each snapshot in the snapshot set a probability which reflects the relative likelihood of that snapshot being the one which would occur if an encodement grid were randomly positioned over that scene. This entire characterization of a scene view -- i.e., the snapshot set relative to resolution r with its associated probability distribution -- is termed the "snapshot signature" of that scene view relative to resolution r. It is possible to theoretically determine the snapshot signature of a scene view, and this determination is illustrated for a simple line drawing. For more complex scene views in the class of line drawings, a procedure exists for approximating their snapshot

signatures which is based on a linguistic description scheme for generating the encodement of a scene in a fixed, but arbitrarily specified, rotational and translational positioning. This linguistic description scheme (denoted as an "interpretive coordinate grammar") is presented and discussed in detail in Chapter 3, and could be considered as a generalization of the "picture-processing grammars" of Chang [C-1].

The utility of the snapshot signature characterization of a scene view in a scene identification context is discussed in Chapter 4 within the framework of statistical hypothesis testing. Two examples are presented in which the coarseness of the encodement resolution makes it impossible to distinguish between any of the scene views being considered on the basis of the structural characteristics of any single encodement of an unknown one of those scene views. Yet a sequential pattern recognition method based on snapshot signatures is capable of identifying these scene views with high reliability in this situation. The only assumption that has to be made is that the unknown scene view can be encoded repeatedly with independent and randomly selected translational positionings of the encodement grid. Hence, the advantage of the snapshot signature characterization in scene view identification at coarse encodement resolutions becomes evident. Further, it is shown that a memory savings is possible in some scene identification situations -- in terms of both the (average)

total amount of information which must be processed, and the amount of information which needs to be available at any one time -- by using scene identification techniques based on snapshot signatures at lower resolutions, rather than equally reliable scene identification techniques based on a single encodement examination at higher resolutions.

So, there can be definite benefits gained by using the multiple encodement characterizations of scenes in scene identification situations. The general methods presented for line drawings apply equally as well to other types of scene structures (e.g., textures), although the linguistic description scheme for generating encodements would no longer be applicable. It is also important to note that scene views (scenes in fixed orientation) are considered instead of scenes, because of the decreased complexity involved in processing scene views. Such considerations suffice to demonstrate the practical advantage of using multiple encodements at a given resolution. However, the general methods developed for characterizing scene views by their multiple encodements could be applied equally as well to characterizing scenes by their multiple encodements, albeit with a considerable increase in complexity.

5.2 Suggestions for Future Work

The extension of snapshot signature characterizations to arbitrary scenes (i.e., those without a fixed

orientation) would be highly desirable. Rotational considerations based on scene views alone are awkward, at best. For instance, if an arbitrary scene were to be identified using the snapshot signature approach based on translational considerations only, a finite number of fixed rotational orientations of that scene would have to be used to define separate views of that scene, with each such scene view then considered individually in the identification process. By making the rotational increments which define each such scene view small enough, it may be feasible to use this scheme to recognize a scene in any arbitrary rotational orientation. However, a more direct approach is suggested whereby the snapshot signature characterization of a scene view is extended to include all the snapshots it can generate for all possible relative translational and rotational positionings of an encodement grid over that scene view. These "rotationally extended" snapshot signatures can then be used to characterize the entire scene, rather than just a view of that scene, relative to some encodement resolution. Identification procedures could thus be based on the extended characterization of a scene. The theoretical computation of a "rotationally extended" snapshot signature will undoubtedly be quite complex (especially in regard to determining the probability distribution over the snapshot set) for all but the simplest of scene structures. But for binary encodements of line

drawings, the linguistic algorithm of Chapter 3 can be applied directly to generate the encodement of a scene in any fixed, but arbitrary, translational and rotational positioning with respect to the encodement grid. Hence, an effective procedure for approximating the "rotationally extended" snapshot signature of a scene can easily be developed, based on making repeated encodements of that scene for various independently and randomly chosen rotational orientations and relative translational positionings of that scene with respect to the encodement grid.

Another area suggested for future work is the investigation of syntactic picture identification based on a formal linguistic description of the multiple encodement characterizations of scene views (or scenes) relative to some resolution. Unfortunately, the linguistic description scheme presented in Chapter 3 is not readily suited to conventional grammatical parsing techniques, since many of its rules ("productions") are really "production schemata", each of which represents an infinite number of rules (based on the different possible values of the coordinates). The determination of the equivalence classes of those scene views which can generate the same snapshot sets at a given resolution can be helpful in this respect, as there will be only a finite number of such equivalence classes at a given

resolution.[1] A linguistic description scheme could then be based on this finite number of equivalence classes of scene views, instead of on an infinite number of scene views. Of course, such a description scheme would not be useful for reflecting the probability distribution which each scene view (or scene) associates with its snapshot set, as individual scene identities would be lost in the equivalence classes.

Fu and Huang ([F-5] and [H-6]) have presented a method for incorporating probabilities of scene view occurrences directly into the production rules of a linguistic scene description scheme. The result is that any scene description generated from the linguistic scheme has its probability of occurrence automatically associated with it. The extension of this method to a linguistic description scheme which could produce scene encodements (snapshots) along with their associated probability of occurrence suggests an interesting area for further research.

---

[1] This is because there are only a finite number of distinct snapshots possible at a finite encodement resolution, and hence only a finite number of distinct sets of these snapshots.

BIBLIOGRAPHY

BIBLIOGRAPHY

[A-1] Andrews, Harry C. Computer Techniques in Image Processing. New York: Academic Press, 1970.

[C-1] Chang, Shi-Kuo. "Analysis and Synthesis of Two-dimensional Patterns Using Picture-processing Grammars." Ph.D. Dissertation, University of California at Berkeley, 1969.

[D-1] Dacey, M. F. "The Syntax of a Triangle and some other Figures." Pattern Recognition (Special Issue on Conceptual Aspects of Two-Dimensional Picture Processing), Vol. 2, No. 1 (Jan., 1970), pp. 11-31.

[D-2] Duda, R. O. "Elements of Pattern Recognition." Adaptive, Learning and Pattern Recognition Systems. Edited by J. M. Mendel and K. S. Fu. New York: Academic Press, 1970. pp. 3-33.

[F-1] Feder, Jerome. "Languages of Encoded Line Patterns." Information and Control, Vol. 13, No. 3 (Sept., 1968), pp. 230-244.

[F-2] Fischler, M. A., and Elschlager, R. A. The Representation and Matching of Pictorial Structures. Lockheed Missles and Space Company Information Sciences Palo Alto Research Laboratory Technical Report No. LMSC-D243781. Palo Alto, Calif.: Lockheed Missles and Space Co., Sept. 1971.

[F-3] Fu, K. S. Sequential Methods in Pattern Recognition and Machine Learning. New York: Academic Press, 1968.

[F-4] Fu, K. S. "Statistical Pattern Recognition." Adaptive, Learning and Pattern Recognition Systems. Edited by J. M. Mendel and K. S. Fu. New York: Academic Press, 1970. pp. 35-79.

[F-5] Fu, K. S. On Syntactic Pattern Recognition and Stochastic Languages. Purdue University School of Electrical Engineering Technical Report No. TR-EE 71-21. Lafayette, Indiana: Purdue University, Jan., 1971.

[F-6] Fu, K. S., and Swain, P. H. "On Syntactic Pattern Recognition." Software Engineering. Vol. 2 of the Proceedings of the Third International Symposium on Computer and Information Sciences (Miami, Fla., 1969). Edited by Julius T. Tou. 2 vols. New York: Academic Press, 1971. pp. 155-182.

[G-1] Gans, David. Transformations and Geometries. New York: Appleton-Century-Crofts, 1969.

[G-2] Ginsburg, Seymour. The Mathematical Theory of Context Free Languages. New York: McGraw-Hill Book Co., 1966.

[H-1] Harary, Frank. Graph Theory. Reading, Mass.: Addison-Wesley Publishing Co., 1969.

[H-2] Harlow, Charles; Synder, Arthur; and Lee, Shin Maan. "Image Analysis and Line Identification." Proceedings of the UMR-MERVIN J. KELLY Communications Conference. Rolla, Mo.: University of Missouri, October 5-7, 1970, pp. 6-3-1 thru 6-3-9.

[H-3] Harlow, Charles; Caudill, Patrick; Lehr, J.; and Parkey, R. "On Image Analysis." Proceedings of the UMR-MERVIN J. KELLY Communications Conference. Rolla, Mo.: University of Missouri, October 5-7, 1970, pp. 6-4-1 thru 6-4-6.

[H-4] Hoel, Paul G.; Port, Sidney C.; and Stone, Charles J. Introduction to Statistical Theory. Boston: Houghton Mifflin Co., 1971.

[H-5] Hopcroft, John E., and Ullman, Jeffrey D. Formal Languages and Their Relation to Automata. Reading, Mass.: Addison-Wesley Publishing Co., 1969.

[H-6] Huang, T., and Fu, K. S. "On Stochastic Context-Free Languages." Information Sciences, Vol. 3, 1971, pp. 201-224.

[L-1] Lang, Serge. Basic Mathematics. Reading, Mass.: Addison-Wesley Publishing Co., 1971.

[L-2] Levine, Martin D. "Feature Extraction: A Survey." Proceedings of the IEEE, Vol. 57, No. 8 (August, 1969), pp. 1391-1407.

[M-1] Meyer, Paul L. Introductory Probability and Statistical Applications. Reading, Mass.: Addison-Wesley Publishing Co., 1965.

[M-2] Michie, Donald. "The Intelligent Machine." Science Journal, Vol. 6, No. 10 (October, 1970), pp. 50-54.

[M-3] Minsky, Marvin. "Steps Toward Artificial Intelligence." Computers and Thought. Edited by Edward A. Feigenbaum and Julian Feldman. New York: McGraw-Hill Book Co., 1963. pp. 406-450.

[M-4] Mood, Alexander M., and Graybill, Franklin A. Introduction to the Theory of Statistics. 2d ed. New York: McGraw-Hill Book Co., 1963.

[N-1] Narasimhan, R. "Picture Languages." Picture Language Machines. Edited by S. Kaneff. London: Academic Press, 1970. pp. 1-30.

[N-2] Narasimhan, R., and Reddy, V. S. N. "A Syntax-Aided Recognition Scheme for Handprinted English Letters." Pattern Recognition, Vol. 3, No. 4 (Nov., 1971), pp. 345-361.

[P-1] Pattern Recognition Society. Pattern Recognition, (Special Issue on Optical Character Recognition), Vol. 2, No. 3 (Sept. 1970).

[P-2] Pattern Recognition Society. Pattern Recognition, (Special Issue on Syntactic Pattern Recognition -- Part One), Vol. 3, No. 4 (Nov., 1971), pp. 341-448.

[P-3] Pattern Recognition Society. Pattern Recognition, (Special Issue on Syntactic Pattern Recognition -- Part Two), Vol. 4, No. 1 (Jan., 1972), pp. 1-144.

[P-4] Pfaltz, John L., and Rosenfeld, Azriel. "Web Grammars." Proceedings of the International Joint Conference on Artificial Intelligence. Edited by Donald E. Walker and Lewis M. Norton. Washington, D.C., May 7-9, 1969, pp. 609-619.

[R-1] Ribak, Richard. "N-Dimensional Grammar for Pattern Recognition." Ph.D. Dissertation, The University of Connecticut, 1971.

[R-2] Rosenberg, B. "Character Blurring." The Australian Computer Journal., Vol. 3, No. 1 (February, 1971), pp. 43-46.

[R-3] Rosenfeld, Azriel. Picture Processing by Computer. New York: Academic Press, 1969.

[R-4] Rosenfeld, Azriel. "Picture Processing by Computer." Computing Surveys, Vol. 1, No. 3 (Sept. 1969), pp. 147-176.

[S-1] Shaw, Alan C. "Parsing of Graph-Representable Pictures." Journal of the Association for Computing Machinery, Vol. 17, No. 3 (July, 1970), pp. 453-481.

[T-1] Tou, J. T. "Feature Extraction in Pattern Recognition." Pattern Recognition, Vol. 1, No. 1 (July, 1968), pp. 3-11.

[V-1] Viglione, S. S. "Applications of Pattern Recognition Technology." Adaptive, Learning, and Pattern Recognition Systems. Edited by J. M. Mendel and K. S. Fu. New York: Academic Press, 1970. pp. 115-162.

[W-1] Wald, Abraham. Sequential Analysis. New York: John Wiley and Sons, 1947.

[W-2] Weiman, Carl F. R., and Rothstein, Jerome. Pattern Recognition by Retina Like Devices. Ohio State University Computer and Information Science Research Center Technical Report No. OSU-CISRC-TR-72-8. Columbus, Ohio: Ohio State University, July, 1972.

[W-3] Wetherill, G. Barrie. Sequential Methods in Statistics. London: Methuen and Co., 1966.

General References

Chang, S. K., and Shelton, G. L. "Two Algorithms for Multiple View Binary Pattern Reconstruction." Proceedings of the UMR-MERVIN J. KELLY Communications Conference. Rolla, Mo.: University of Missouri, October 5-7, 1970, pp. 20-2-1 thru 20-2-8.

Fink, Donald G., and Lutyens, David M. The Physics of Television. Garden City, New York: Anchor Books, Doubleday and Co., 1960.

Glasford, Glenn M. Fundamentals of Television Engineering. New York: McGraw-Hill Book Co., 1955.

Gordon, Richard, and Herman, Gabor T. "Reconstruction of Pictures from Their Projections." Communications of the ACM, Vol. 14, No. 12 (Dec., 1971), pp. 759-768.

IEEE Computer Society. IEEE Transactions on Computers (Special Issue on Two-Dimensional Signal Processing), Volume C-21, No. 7 (July, 1972), pp. 633-836.

Kaneff, S. (ed.). Picture Language Machines. London: Academic Press, 1970.

Knuth, Donald E. The Art of Computer Programming. Vol. I: Fundamental Algorithms. Reading, Mass.: Addison-Wesley Publishing Co., 1968.