

**THE RAPID RECOVERY OF
THREE-DIMENSIONAL ORIENTATION
FROM LINE DRAWINGS**

R.A. Rensink

Technical Report 92-25

September 1992

Abstract

A computational theory is developed that explains how line drawings of polyhedral objects can be interpreted rapidly and in parallel at early levels of human vision. The key idea is that a time-limited process can correctly recover much of the three-dimensional structure of these objects when split into concurrent streams, each concerned with a single aspect of scene structure.

The work proceeds in five stages. The first extends the framework of Marr to allow a process to be analyzed in terms of resource limitations. Two main concerns are identified: (i) reducing the amount of nonlocal information needed, and (ii) making effective use of whatever information is obtained. The second stage traces the difficulty of line interpretation to a small set of constraints. When these are removed, the remaining constraints can be grouped into several relatively independent sets. It is shown that each set can be rapidly solved by a separate processing stream, and that co-ordinating these streams can yield a low-complexity "approximation" that captures much of the structure of the original constraints. In particular, complete recovery is possible in logarithmic time when objects have rectangular corners and the scene-to-image projection is orthographic. The third stage is concerned with making good use of the available information when a fixed time limit exists. This limit is motivated by the need to obtain results within a time independent of image content, and by the need to limit the propagation of inconsistencies. A minimal architecture is assumed, viz., a spatiotopic mesh of simple processors. Constraints are developed to guide the course of the process itself, so that candidate interpretations are considered in order of their likelihood. The fourth stage provides a specific algorithm for the recovery process, showing how it can be implemented on a cellular automaton. Finally, the theory itself is tested on various line drawings. It is shown that much of the three-dimensional structure of a polyhedral scene can indeed be recovered in very little time. It also is shown that the theory can explain the rapid interpretation of line drawings at early levels of human vision.

Contents

Abstract	ii
List of Figures	vi
List of Tables	ix
Acknowledgements	x
1 Introduction	1
1.1 The Problem	2
1.2 The Approach	4
1.3 Limitations and Key Assumptions	6
2 Background	8
2.1 Rapid Parallel Processing	8
2.1.1 Computational Studies	9
2.1.2 Psychophysical Studies	19
2.1.3 Computational versus Psychophysical Studies	26
2.2 The Interpretation of Line Drawings	28
2.2.1 Computational Studies	28
2.2.2 Psychophysical Studies	34
2.2.3 Computational versus Psychophysical Studies	38
2.3 High-level versus Low-level Vision	39
2.3.1 The Structure of Low-level Vision	39
2.3.2 The Role of Rapid Parallel Recovery	42
2.4 The Analysis of Resource-Limited Processes	43
2.4.1 Marr's Framework	44

2.4.2	Extensions	45
2.4.3	A Revised Framework	47
2.5	Rapid Line Interpretation	50
2.5.1	Basic Terms	50
2.5.2	Formulation of the Problem	51
3	Low-Complexity Recovery	54
3.1	General Issues	55
3.1.1	Concurrent Streams	55
3.1.2	Reduction to Canonical Forms	56
3.1.3	Approximation Strategies	60
3.2	Individual Dimensions	61
3.2.1	Contiguity Labelling	62
3.2.2	Convexity Labelling	67
3.2.3	Slant Sign Labelling	70
3.2.4	Slant Magnitude Labelling	75
3.3	Integration of Dimensions	78
3.3.1	Convex Objects	79
3.3.2	Compound Convex Objects	81
3.3.3	Rectangular Objects	85
4	Computational Analysis	100
4.1	External Constraints	101
4.1.1	Image-to-Scene mapping	101
4.1.2	General Principles	103
4.1.3	Structural Assumptions	105
4.1.4	System of External Constraints	106
4.2	Internal Constraints	108
4.2.1	Processing Architecture	109
4.2.2	General Principles	110
4.2.3	Selection of Initial Candidates	118
4.3	The Rapid Recovery Process	124
4.3.1	Architectural Specifications	124
4.3.2	Robustness	126

4.3.3	Basic Operation	126
5	Algorithm and Implementation	129
5.1	The Cellular Processor	129
5.1.1	Basic aspects	130
5.1.2	Cellular Processors as Cellular Automata	131
5.1.3	Programming	133
5.2	Algorithm for Rapid Recovery	136
5.2.1	Determination of Basic Image Properties	137
5.2.2	Determination of Junction Properties	137
5.2.3	Initial Assignment of Interpretations	142
5.2.4	Propagation of Interpretations	142
5.2.5	Final Assignment of Results	144
5.3	Neural Implementation	144
6	Tests of the Theory	146
6.1	Performance on Line Drawings	146
6.1.1	Rectangular Objects	147
6.1.2	Nonconforming Objects	151
6.1.3	Impossible Objects	157
6.2	Preattentive Recovery of Scene Structure	162
6.2.1	Basic Assumptions	162
6.2.2	Explanation of Psychophysical Results	165
7	Summary and Conclusions	173

List of Figures

1.1	Early recovery of three-dimensional structure.	3
2.1	Linkage between zone and surrounding locations.	10
2.2	Types of junctions.	31
2.3	Huffman-Clowes labelling set.	32
2.4	Penrose triangle.	36
2.5	Extended computational framework.	48
3.1	Linking of local constraints.	59
3.2	Separation into individual dimensions.	62
3.3	Contiguity labelling.	63
3.4	Set of contiguity constraints.	64
3.5	Inconsistent drawing with consistent contiguity labelling.	65
3.6	Reformulation of contiguity constraints.	66
3.7	Set of convexity constraints.	68
3.8	Inconsistent drawing with consistent convexity labelling.	68
3.9	Reformulation of convexity constraints.	69
3.10	Slant sign labelling.	71
3.11	Constraints on isolated L-junctions.	72
3.12	Slant sign constraints for arrow- and Y-junctions.	73
3.13	Slant sign labellings for rectangular corners.	74
3.14	Huffman-Clowes labellings for convex objects.	80
3.15	Examples of compound convex objects.	82
3.16	Huffman-Clowes labellings for compound convex objects.	82
3.17	Free chain complexes.	84
3.18	Examples of rectangular objects.	85

3.19	Constraints on L-junctions.	86
3.20	Huffman-Clowes labellings for rectangular objects.	86
3.21	Planarity constraint.	87
3.22	Interior angle constraint.	87
3.23	Contiguity constraints on obtuse L-junction combinations.	89
3.24	Termination configurations.	90
3.25	Contiguity constraints on Y-junction combinations.	92
3.26	Slant sign constraints applied to impossible figures.	95
3.27	Conditions of shallow slant.	97
3.28	Combinations of angles into corners.	97
3.29	Rescaling of image angles.	98
4.1	Isolation of inconsistency in contiguity labelling	103
4.2	System of external constraints	107
4.3	Example of complementary labelling.	112
4.4	Example of reformulation of bijective constraint.	115
4.5	Example of reformulation of nonbijective constraint.	115
4.6	Initial contiguity interpretations.	120
4.7	Initial convexity interpretations.	122
4.8	Initial slant-sign interpretations.	123
4.9	Example of the rapid recovery process.	128
5.1	Cellular processor architecture.	130
5.2	Calculation of orientation differences.	139
5.3	Determination of contiguity relations.	141
6.1	Interpretation of convex rectangular object.	149
6.2	Interpretation of nonconvex rectangular object.	150
6.3	Interpretation of occluded rectangular objects.	152
6.4	Interpretation of nonrectangular object.	154
6.5	Interpretation of origami object.	155
6.6	Interpretation of nonplanar object.	156
6.7	Interpretation of object of inconsistent contiguity and convexity.	158
6.8	Interpretation of object of inconsistent slant.	160

6.9 Interpretation of object of inconsistent depth.	161
6.10 Results explained by theory.	163
6.11 Slant estimates for Condition A.	166
6.12 Slant estimates for Condition B.	167
6.13 Slant estimates for Condition C.	169
6.14 Slant estimates for Condition E.	170
6.15 Slant estimates for Condition G.	172

List of Tables

2.1 Complexities of coherence classes.	15
--	----

Chapter 1

Introduction

Those aspects of human vision most directly involved with the incoming image have a characteristic mode of operation: they are rapid (usually completed within several hundred milliseconds), spatially parallel (operating simultaneously across the visual field), and automatic (unaffected by changes in goals during the course of processing). This has led to an assumption that these “early” processes determine only simple geometric and radiometric properties of the image, e.g., line orientation, color, and contrast. There is considerable support for this assumption on computational grounds — these are the only kinds of properties can be reliably determined by spatially-limited processors operating within a fixed amount of time. To reliably determine properties of the corresponding scene, therefore, a later stage of more time-consuming operations is needed.

This division into early and later processes has formed the basis for many computational and psychophysical studies of the human visual system. However, the underlying assumption is false — for some images, recovery of scene properties *can* be done at early stages of processing, rapidly and in parallel [Ram88, ER90a, ER91]. In figure 1.1(a), for example, the drawing of the block with a unique three-dimensional orientation can be detected almost immediately. However, this is not possible when these drawings are altered slightly (figure 1.1(b)), showing that this phenomenon is not due to simple image properties alone, but to some aspect of the recovered scene structure.

The goal of this thesis is to explain how properties of the scene can be recovered rapidly and in parallel at early levels of visual processing. In particular, it develops a computational theory of how the human visual system can rapidly interpret line drawings to obtain the three-dimensional structure of the corresponding polyhedral objects. Since the general problem of

line interpretation is NP-complete [KP88], a great deal of time may sometimes be required for its solution, even when parallel processing is used. If recovery is to be rapid, therefore, it cannot be based on this mapping, but rather must be based on an approximation in which the reliability and completeness of the output have been lowered to some degree.

The central idea developed here is that a good approximation can be obtained by splitting the recovery process into several quasi-independent streams, each based on a set of constraints that can be quickly solved. It is shown that relatively few constraints need to be altered in order to achieve this decomposition, and that the resulting “quick and dirty” process can recover a substantial amount of scene structure in very little time. It is also shown that this model can explain the recovery of three-dimensional structure at early levels of human vision.

In common with other areas of computational analysis, this study is first and foremost concerned with how information can be used by a visual system. For rapid recovery, however, the structure of the problem is no longer dictated entirely by the optics of the situation — instead, limits on processing time must also be taken into account. This work shows how this perspective can be incorporated into a computational framework, and how it can lead to a new source of constraints on the representations and processes used in early vision.

1.1 The Problem

In what follows, the scene domain is taken to be the set of opaque polyhedral objects with trihedral corners. The term ‘trihedral’ is used here in a narrow sense, referring to corners formed from the intersection of three planar surfaces in such a way that only three edges can radiate from any vertex, and that the vertex cannot contact any other edge. The image domain is the corresponding set of drawings formed by the projection of these objects onto the image plane. The rapid recovery process must recover from these drawings as much of the scene structure as possible within some fixed amount of time. The goal of this work is to develop a computational theory of this process, one which accounts for those aspects of three-dimensional structure recovered in human early vision.

There are several reasons for this choice of problem. First, there is evidence that human vision actually *does* recover three-dimensional structure rapidly and in parallel at early levels [ER90b, ER91, ER92]. The phenomenon is a striking and robust one, with a strong sensitivity to the arrangement of the lines. As such, there is considerable potential for making predictions about the kinds of line arrangements for which recovery will and will not be successful.

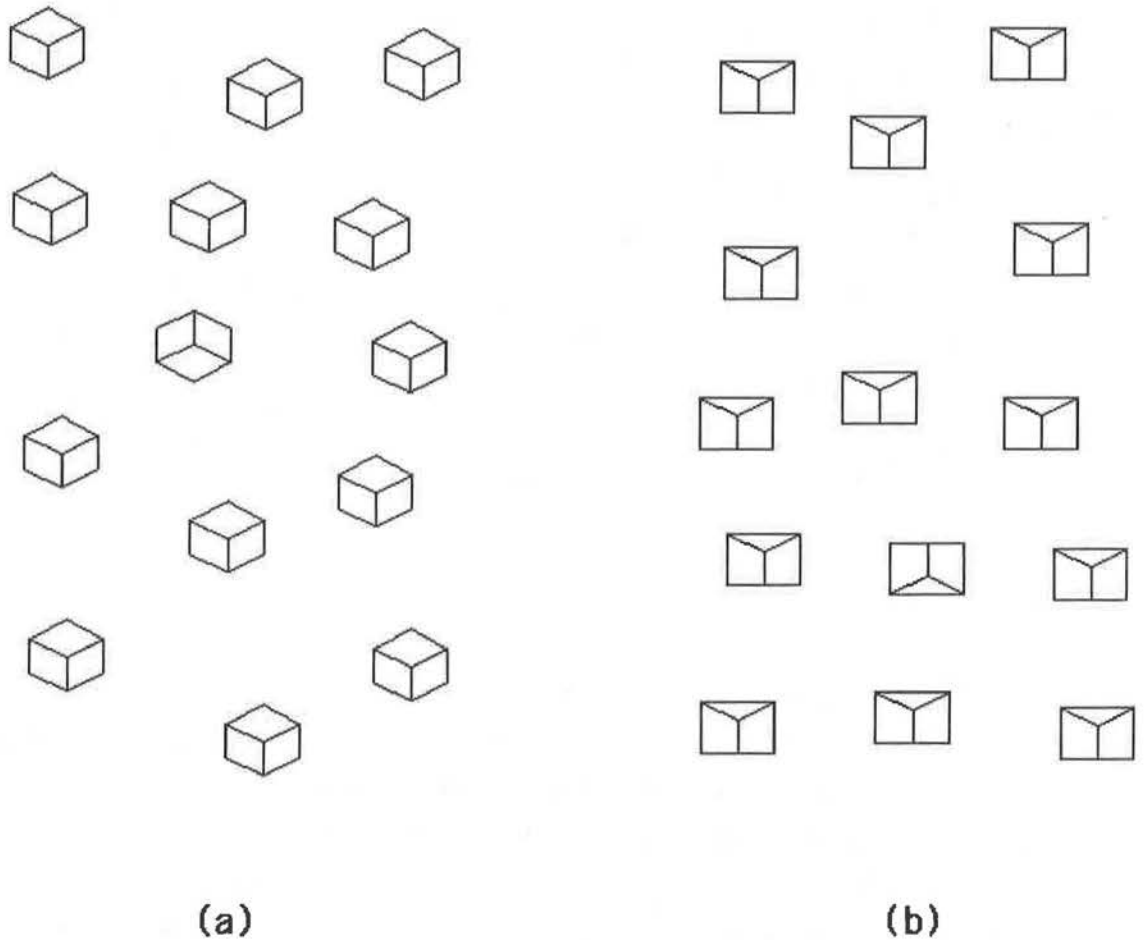


Figure 1.1: Early recovery of three-dimensional structure. A line drawing that corresponds to a distinct three-dimensional block can be detected almost immediately when the block slants upwards (a). Rotating the page so that this block slants downwards causes detection to become more difficult, showing that slant has an asymmetry typical of many properties of early vision (see [TG88, ER90b]). When line relations are slightly altered (b), detection is equally difficult under all conditions (also see [ER91]), indicating that slant is not recovered at all.

Second, a great deal is known about the limits to which three-dimensional structure can be recovered from line drawings,¹ this problem having been the focus of several decades of work in the area of computational vision (see section 2.2.1). Moreover, the general problem of line interpretation has been shown to be NP-complete [KP85]. Since the time required to solve an NP-complete problem can (in the worst case) increase exponentially with its size,² this rules out the possibility that the process can always be sped up by parallel processing alone.

Finally, of all the rapid recovery processes, line interpretation is perhaps that which most severely taxes the abilities of early vision. Relations between image and scene are more tenuous here than for most other recovery processes; indeed, many aspects of line interpretation are often considered to be learned conventions (see, e.g., [Sug86]). Thus, if a mechanism can be found for the rapid interpretation of line drawings, it becomes plausible that similar mechanisms might also exist for recovery processes based on more realistic associations between image and scene.

1.2 The Approach

For a time-limited process, the goal is no longer to extract all available information from an image, but rather to make good use of the available computational resources. Two factors are therefore of primary concern: (i) minimizing the sheer amount of data transformation and transmission that needs to be carried out in parallel, and (ii) maximizing the effectiveness of these transformations in extracting three-dimensional structure. This work examines how these two factors influence the structure of the recovery process at the levels of computation, algorithm, and implementation.

Chapter 2 provides the background material for this analysis. It begins with a survey of the major empirical and theoretical results on the limits of rapid parallel processing. This is followed by an overview of the important results concerning the recovery of three-dimensional structure from line drawings. A discussion is then presented of the ways in which these two

¹Theories of line interpretation, however, have rarely taken into account noise and other distortions of the image. Complications also arise from shadow edges and texture boundaries. In the interests of simplicity, these will not be discussed here.

²Although there remains a possibility that NP-complete problems are in class P (i.e., can be carried out in polynomial time in the worst case), this situation appears highly unlikely [GJ79, Joh90]. Even if $P=NP$, the possibility would still exist that such problems are P-complete, meaning that their speed could not be substantially increased by the use of parallel processing [GR88].

threads can be drawn together. Next, Marr's framework of computational analysis is extended to cover the case of resource-limited processes. Two sorts of computational constraints are distinguished: "external" constraints on the static form of the mapping between image and scene, and "internal" constraints that guide the course of the process that generates it.

The analysis of rapid recovery itself begins in chapter 3, which examines the ways in which the image-to-scene mapping used in the general problem of line interpretation can be replaced by an approximation of lower complexity. In particular, it shows that low-complexity recovery can be carried out by weakening the constraints to allow their separation into independent subsets, each concerned with a single aspect of the scene. Four such aspects are considered: the contiguity of edges, the positive convexity of edges, the sign of edge slants, and the magnitude of edge slants. It is shown that each of these subsets can be solved in sublinear time by a processing stream containing a sufficiently large set of parallel processors, and that this complexity is not increased when interaction between the streams involves only a one-way transmission of information. Although the interpretative power of the resultant mapping is somewhat reduced, a considerable amount remains; indeed, it is shown that contiguity, convexity, and slant can be recovered completely in logarithmic time when all corners are rectangular, i.e., composed of mutually orthogonal surfaces.

The next step is to develop constraints that maximize the likelihood of successful interpretation when a limit is placed on processing time. This is done in chapter 4. A fixed amount of time is assumed to be available. This choice is consistent with the limits typical for an early visual process, and also has the advantage that the propagation of inconsistencies is localized. In keeping with this minimalist vein, computational resources are limited to a mesh of simple processors. A set of external constraints is developed to limit the space of possible interpretations. Four principles are used for the choice of constraints: separation of dimensions, locality of constraints, local coordination of dimensions, and the structural assumption that the corners of the polyhedra are rectangular. Internal constraints are then developed that guide the course of the recovery process through this space of possible solutions. These are based on four principles: maintenance of interpretative power, locally irreversible computation, minimization of inconsistency, and an ordering of search to select preferred interpretations of maximum contiguity and convexity. Taken together, the external and internal constraints define a process capable of recovering a considerable amount of three-dimensional structure in very little time.

Although the external and internal constraints limit the way in which the process uses information, they do not completely specify an algorithm. Chapter 5 provides this specification, and implements the resulting algorithm on a mesh architecture. This is done via the device of a *cellular processor*. This mechanism is formed by partitioning the image into a set of disjoint “cells”, each governed by a finite-state processing element that can be programmed to execute a few simple operations on the contents of its cell, and that can communicate only with its immediate neighbor. As such, it obeys the general architectural limitations assumed for the computational analysis while simultaneously being easy to control and analyze. The resulting algorithm provides an existence proof that rapid line interpretation can be done on an architecture of the assumed type.

The final step of the work is to test the theory on actual line drawings. In chapter 6, the process is tested on domains that range from those in which all underlying assumptions are obeyed to impossible figures which cannot correspond to any kind of polyhedron at all. It is shown that a considerable amount of three-dimensional structure can indeed be recovered in very little time, and that this process degrades gracefully as the underlying structural assumptions about the scene domain are violated. These results are then used as the basis of predictions about the kinds of line drawings that can and cannot be rapidly detected by the human visual system. The theory is shown to be capable of explaining the ability of early human vision to recover three-dimensional structure rapidly and in parallel.

1.3 Limitations and Key Assumptions

Before embarking on the development of the theory, it is important to acknowledge a number of limitations and assumptions that could potentially limit its relevance. First of all, the treatment here is concerned exclusively with the rapid recovery of three-dimensional structure from line drawings. The advantage of this approach is that the problem domain is small and has a simple mathematical description, making the analysis of the recovery process as simple as possible. But this domain is a highly artificial one — the figures contain no gaps or any other kind of noise, nor do they describe cracks and markings which are found on just about any real surface. Indeed, these stimuli are so artificial that the analysis runs the risk of saying nothing at all about processes that interpret more realistic images.

The scope of the theory is also limited by the architectural constraints required for the computational analysis. The analysis here assumes only a two-dimensional array of relatively

simple processors, each connected only to its immediate neighbors (chapter 4). Since this limitation is relatively severe, the resulting process provides a lower bound to what might be reasonably expected from a spatiotopic array of processors. But the assumption of a minimal processing architecture also means that the predictions are applicable only to the extent that such an architecture actually is representative of that used in human early vision.

There is also considerable latitude in the choice of the finer details of the theory. Several of the choices made here are somewhat tentative, intended only to show that such a theory *can* be developed. Consequently, they are unlikely to withstand the test of time.

Insofar as the theory can explain the rapid recovery of three-dimensional orientation by the early human visual system, it assumes that this system actually *does* carry out this process. Results to date [ER90b, ER91, ER92] show that the recovery of three-dimensional orientation at early levels is sufficient to explain most known results concerning the sensitivity of early vision to line drawings of opaque polyhedra. But while this sensitivity cannot be explained in terms of simple operations on the image (e.g., spatial filtering), there still remains the possibility of some other “image-based” explanation, e.g., a sensitivity to particular spatial relations between the lines, or the “loading-in” of a complete object model via lookup that based on image features (e.g. [PE90]).

Finally, even if three-dimensional orientation actually is computed at these early levels, there is still no guarantee that the process is in any way attempting to make good use of available computational resources. Evolution often produces biological systems that are adequate rather than optimal (see, e.g., [Ram85, Gou89]), and it may well be that rapid recovery falls into this category. If so, its operation is governed by constraints other than those based on effectiveness, and the computational model developed here will be largely irrelevant for explaining human performance.

A scientific theory, however, ultimately succeeds or fails to the degree that it explains phenomena in a succinct way, and suggests new avenues of research to explore (e.g., [Lak78]). As the following chapters show, the theory developed here is able to account for the recovery of three-dimensional structure at early levels of human vision, and can make predictions as to what other kinds of line drawings can and cannot be recovered in this way. Furthermore, it does so by developing principles that are potentially applicable to other areas of perception and cognition, and to both artificial and biological systems.

Chapter 2

Background

The problem of rapid line interpretation is a fusion of two concerns that historically have been quite separate: (i) determining the extent to which properties can be extracted rapidly and in parallel from an image, and (ii) determining the extent to which line drawings can be interpreted as opaque polyhedral objects. Thus, a good way to begin is to survey the principal developments in each of these areas. It is then shown how aspects of both can be usefully combined into a rapid recovery process, and how this process can be analyzed by an extension of the computational framework of Marr [Mar82].

2.1 Rapid Parallel Processing

The earliest stages of visual processing are characterized by the uniform application of relatively simple operations at each location in the visual field (see, e.g., [Zuc87b, LBC89]). It is evident that the problems solved at these levels make great use of parallelism, with one or more processors assigned to each patch of the image. It is less evident, however, what the limits of this kind of processing might be. This section surveys some of the main results pertaining to rapid parallel processing. Theoretical results are presented first, with discussion focusing on the way in which a problem's structure determines its complexity on a parallel processor. This is followed by an overview of what is known about the extent of rapid parallel processing in human early vision.

2.1.1 Computational Studies

Two different routes can be taken when studying the limits of parallel processing. The first starts with a given architecture and then determines its suitability for various classes of problems. Such “processor-dependent” analysis is widely used, particularly to ascertain the capabilities of an existing machine (e.g., [PD84, LBC89]). But the emphasis here is on *problems* rather than architectures *per se*. Consequently, a “dual” approach is taken: a class of problems is specified and the suitability of various architectures for this class then examined. This approach can be based on the amount of coherence in the mapping between input and output image. It is shown that this coherence has a large influence on the limits to which an operation can be sped up by a parallel architecture.

A. Basics

To establish what is meant by the “suitability” of an architecture for a particular problem, consider first a network of Turing machines joined together by high-bandwidth connections. Such a “maximal” architecture obviously allows the greatest use to be made of parallelism, regardless of problem structure. Its generality, however, means that computational resources are often wasted. A *natural architecture* is therefore defined as one which best matches the given problem, i.e., which uses a minimal set of resources to carry out the task. Such an architecture can be obtained (conceptually, at least) by starting with a maximal architecture and then weakening the power of the individual processors and the communication patterns between them until a change occurs in the time or space required. The minimal configurations at each of these transitions are exactly the natural architectures. Since different choices of time and space bounds are possible, there is usually more than one natural architecture for a given problem.

In general, finding the natural architecture for a problem is difficult — even the mapping of neighbors in the problem space to neighbors in the architecture is NP-hard¹ [NKP87]. But when problems are based on the mapping of images to images² the coherence in the mapping simplifies matters considerably (see, e.g., [Sto87, Sto88]). Finding a natural architecture for a problem then reduces to determining its mapping coherence and relating it to the time and

¹This means that the problem is at least as hard as any NP-complete task.

²Without loss of generality, the calculation of a lower-dimensional result can be expressed as a mapping in which the output image contains repeated instances of the result. For example, calculation of the average value in an image can be expressed in terms of an output image containing the average at each location.

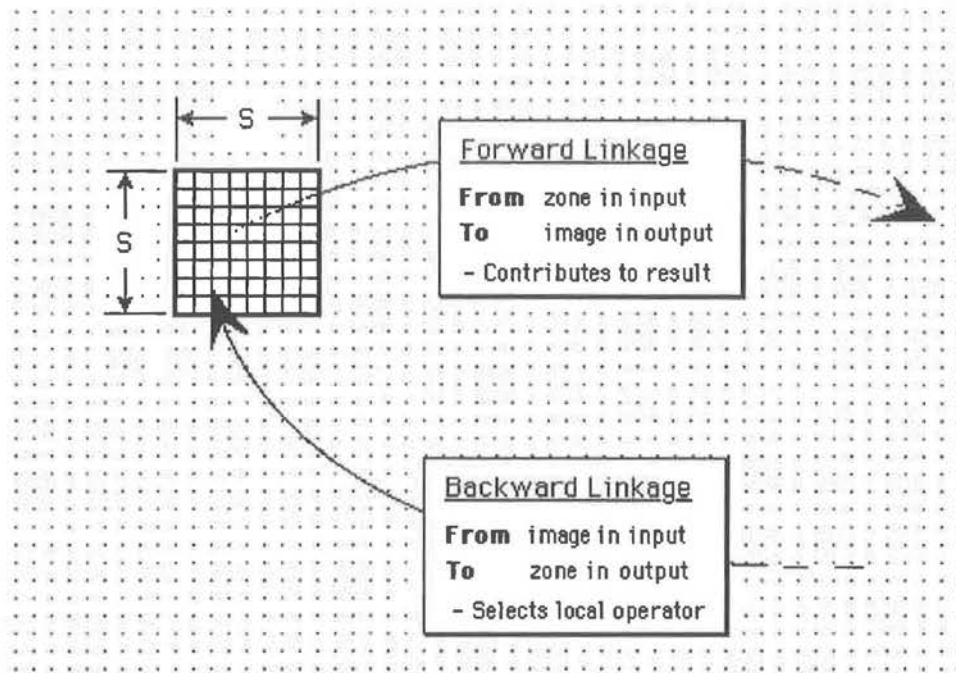


Figure 2.1: Linkage between zone and surrounding locations.

space required on various kinds of architectures.

i) Mapping Coherence

To begin with, let a *zone* be some $s \times s$ patch of pixels in the image. Zones may overlap, so that each $m \times m$ image contains $(m - s)^2$ zones. Mapping coherence is described here in terms of how the input inside a zone influences the output image in the surrounding locations (figure 2.1). A zone is said to *interact with* the rest of the image if there is at least one direction in which the range of this influence is unlimited. The *linkage* of the mapping is defined as the number of degrees of freedom in this interaction.³

For strictly local operations, such as those typical of early vision, no interaction exists between a zone and locations sufficiently far away in the output. These are consequently zero-linkage problems – any changes within a zone are propagated only a finite distance away. At the opposite extreme, consider the sorting of image intensities. Here, changing *any* of the pixel values in the zone potentially changes the value of the output at a location arbitrarily far away. The linkage is therefore proportional to zone area.

³Many of the ideas presented here regarding mapping coherence have their origin in the work of Stout [Sto87, Sto88]. However, the definition of linkage used here is considerably different, being based on degrees of freedom in a more uniform fashion. Distinctions such as unidirectional and bidirectional linkage, as well as the resulting set of coherence classes, are also novel.

Linkages can run both ways, however, so that two kinds of problem can be distinguished. In *unidirectional* problems, the information transmitted from the zone can be computed purely locally — no pixel values are needed from the rest of the image beyond some finite surrounding region. For example, determining the average pixel intensity requires only one parameter (the sum of pixel intensities) to be transmitted from any zone. Since the pixels outside the zone do not affect this value, this interaction is clearly unidirectional, with no outside information needed.

In *bidirectional* problems, on the other hand, the interaction between a zone and its surroundings runs both ways: not only do changes in the zone influence the rest of the image, but the rest of the image influences what is required of the zone. More precisely, the information to be transmitted from a zone cannot be determined in isolation for bidirectional problems, since values from the rest of the image are needed to select the appropriate quantity to be calculated locally. An example of this is the calculation of the median intensity of an image in which the range of pixel values is unlimited (e.g., random variables with a gaussian distribution). A single degree of freedom can be assigned to the local output: the number of pixels above (or below) the global median. But the value of this median must first be transmitted to the zone, and this value may be affected by changes in pixel intensities at locations arbitrarily far away.⁴ In essence, the kind of linkage back from the image to the zone reflects the amount of contextual information needed to select the appropriate local operation (figure 2.1).

Given this characterization of mapping coherence, problems can be grouped according to the strength and directionality of their linkage.⁵ Four classes are considered here: zero linkage, constant linkage, linkage proportional to zone perimeter, and linkage proportional to zone area. Constant- and perimeter-linkage problems are further divided into unidirectional and bidirectional subclasses. Any operation involved in visual processing can be placed into one of these classes,⁶ and it is shown below that placement into a class puts bounds on its complexity on various kinds of architectures.

⁴In a sense, the difference between unidirectional and bidirectional problems corresponds to that between deterministic and nondeterministic problems: in the unidirectional case, the outputs of isolated zones are sufficient to *produce* the solution, whereas in the bidirectional case, they are sufficient only to *verify* it. (For a discussion of the relation between deterministic and nondeterministic problems see, e.g., [GJ79].)

⁵The strength of the linkages can be different in the two directions. But the simple classification into unidirectional and bidirectional problems is sufficient for present purposes.

⁶Operations having a structure that does not match well with the examples discussed here (e.g., those involving fractal quantities) will require a finer division of coherence classes. But as a first approximation, lower bounds for such problems can be obtained by “rounding down” to the nearest coherence class.

ii) Complexity Measures

The complexity of a problem can be analyzed in a relatively processor-independent way via the methods of complexity theory (see, e.g., [Baa78, GJ79, Joh90]). Here, the basic unit is taken to be the time required to merge two independent quantities into one — for example, the addition or multiplication of two numbers, or the testing of their equality. The complexity of a given algorithm is then measured by the number of such operations required for the most difficult case in the problem set.⁷ The complexity of a given *problem* is that of the least-complex algorithm capable of solving it on a given architecture.

Differences in the speed of basic operations — such as arise in different mechanical or biological systems — are eliminated by the use of O -notation, which describes the time only to within a constant factor. An algorithm is said to require $O(f(n))$ time if there exist positive constants c, d and N such that for any input of size $n > N$, the time $cf(n) \leq T(n) \leq df(n)$. If the algorithm is the least complex known to solve the problem, the complexity of the problem is said to have an upper bound of $O(f(n))$. Similarly, the problem has a lower complexity bound $\Omega(g(n))$ when any algorithm to solve it must have a complexity of *at least* $O(g(n))$. If a problem is bounded above by $O(f(n))$ and below by $\Omega(f(n))$, it has (exact) complexity $\Theta(f(n))$ (see, e.g., [Baa78, Har87]).

Defined in this way, the time needed to solve a problem is — to within a polynomial factor — independent of the particular set of instructions of the machine (see, e.g., [Har87]). This quantity is therefore an invariant of the *problem*, (see, e.g., [Baa78, TWW88]), and so can be used for abstract, machine-indifferent analysis [RP91].

As for the case of mapping coherence, processes can be grouped into various complexity classes. (For a good survey of these classes, see [Joh90].) One of these is the set P of processes that can be carried out in polynomial time; such processes may have a different complexity on different (serial) architectures, but this complexity will always be polynomial (see, e.g., [GJ79, Har87]). This class can be further subdivided according to the degree that complexity is lowered by the introduction of parallel processing. The class NC is defined as the set of problems having sublinear complexity when a sufficient number of processors are provided.⁸

⁷Complexity measures can also be based on average-case analysis, as well as on a probabilistic analysis that ignores exceptional cases of small measure (see [TWW88]). However, worst-case measures are those most often used, in part because of the relative ease of analysis. These measures are also preferred here since they avoid the need to develop extra procedures to handle cases in which the computational limitations are exceeded.

⁸More precisely, these are problems of complexity $O(\log^k n)$ when $O(n^p)$ processors are available (where exponents $k, p \in Z$).

In contrast, a class of “P-complete” problems has been found that is apparently incapable of being sped up this way; in essence, these problems remain “serial” no matter how many processors are allowed (see, e.g. [GR88]).

Note that this view of complexity is based on the number of operations needed to *combine* data and so the time needed for data *transmission across space* is often ignored. While this is suitable for many situations, it is less so for others, especially for operations on images, where data is often moved around a considerable distance during the course of the computation. Transmission delays are severe in biological systems (where speeds are typically on the order of 1m/s [She83]), and are also a factor in the operation of machine systems [Uhr87, p. 261]. When applying complexity measures to image-processing problems,⁹ therefore, the effects of transmission delay must be kept in mind.

iii) Architectural Parameters

Given that the complexity of an image-processing problem depends on the underlying architecture, it is important to establish what the relevant parameters might be. In what follows, architectures are described by graphs where each node represents a separate processing element (PE) and each edge a direct connection between the corresponding PEs. Thus, a maximal architecture corresponds to a complete graph in which each PE (equivalent to a Turing machine) is directly connected to all the others. This model is superficially different from the parallel random access machine (PRAM) often used in theoretical studies of parallel processing (e.g., [GR88]), since the PRAM is defined as an abstract machine with a shared memory immediately accessible to any of the processors.¹⁰ But this shared memory allows direct communication between PEs, and so the PRAM and maximal architectures are essentially equivalent.

In this formulation, the complexity of a problem can be analyzed by tracing the flow of information through the network. The path taken by each piece of information can be represented by a path through the graph that begins at the point where it is picked up in the image, and terminates at its final position(s) in the output. The nodes of the graph are

⁹As used here, the terms ‘image-processing problem’ and ‘image operation’ are largely synonymous. The only difference is that the specification of a problem does not necessarily contain an explicit rule to obtain the output from the input, whereas this is generally true of the term ‘operation’.

¹⁰Strictly speaking, this characterizes only the most powerful variant: the PRIORITY concurrent read - concurrent write (CRCW) PRAM. Since no other PRAM variants will be considered here, this qualification will not be explicitly mentioned.

assigned weights representing the time required for local computation.¹¹ A weight can be assigned to each data path by accumulating the weights of all nodes encountered along the way. The processing time for a computation is then the maximum weight of all the data paths in the computation. Thus, complexity is largely governed by two sets of parameters: (i) those concerned with the processing resources available to each PE, and (ii) those concerned with the pattern of data transmission between PEs.

A wide variety of processing elements are possible for a parallel architecture. At one extreme, each PE has the power of a Turing machine and can operate completely independently of the others. This is basically the multiple instruction, multiple data-stream (MIMD) architecture [Fly72], in which each PE may carry out a different set of instructions. Weakening the power of the PEs decreases their ability to respond to different signals so that they become less able to respond to the structure of the image and less flexible in communicating with their neighbors. In the extreme case this becomes a single instruction, multiple data-stream (SIMD) architecture [Fly72], where all PEs operate in lockstep, carrying out the same operation everywhere in the network.

A similar spectrum of possibilities exists for data transmission. The simplest network is a two-dimensional $\sqrt{n} \times \sqrt{n}$ array of n processing elements. Here, each of the processors is assigned to some particular zone or set of zones in the image, and operates in complete isolation from the others. This is the kind of architecture generally thought to exist at the very earliest stages of visual processing, i.e., the retina and the striate cortex (e.g., [Rob80]).

The simplest form of processor-processor interaction occurs in the *mesh*, a network, where each PE in the array is connected with its nearest neighbors (e.g., [Ros83]). Here, data can be sent from any PE to any other PE with time proportional to the distance in the mesh.

Transfer can be greatly sped up (at least in terms of the number of switches involved) by way of a *pyramid* network, in which a hierarchical communication structure is used. The basic $\sqrt{n} \times \sqrt{n}$ mesh forms the lowest level of this hierarchy. This mesh is then partitioned into a set of $k \times k$ nonoverlapping sections, with the PEs in each section then connected to a single PE in a higher-level $\sqrt{n}/k \times \sqrt{n}/k$ mesh. This higher-level mesh is in turn sectioned and connected to the PEs in a still higher-level mesh, this process continuing until only one PE exists at the highest level (see, e.g. [Ros86]). The resulting structure is hierarchical, allowing any two PEs separated by distance m to communicate through $O(\log m)$ switches.

¹¹In this view, switches and memories are regarded as nodes corresponding to simple computations.

Linkage	Array	Mesh	Pyramid	Hypercube
Zero	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$	$\Theta(1)$
Constant (unidirectional)	–	$\Theta(n)$	$\Theta(\log n)$	$\Theta(\log n)$
Constant (bidirectional)	–	?	$O(\log^3 n)$?
Perimeter (unidirectional)	–	$\Theta(n)$	$\Theta(n^{1/2})$	$\Theta(\log n)$
Perimeter (bidirectional)	–	$O(n^k)$	$O(n)$?
Area	–	?	$\Omega(n)$	$O(\log n)$

Table 2.1: Complexities of coherence classes.

Pyramid networks have been proposed for several of the more “global” processes of vision, such as line tracing [Ede87] and selective visual attention [KU84, Tso90]

A more highly-connected architecture is that of the *hypercube* (e.g., [Hil84]). A d -dimensional hypercube has 2^d corners; if the positions of neighbors in the hypercube differ by some constant distance $a > 0$ along any dimension, corners will be separated by at most a distance of ad . Thus, if $n = 2^d$ PEs are connected such that each corresponds to a different corner of the hypercube, then two PEs can communicate within $O(\log n)$ time. Although this is the same as for the pyramid, the greater number of possible paths yields a greater effective bandwidth, which allows the hypercube to avoid the bottlenecks that can arise at the higher levels of the pyramid [Sto87].

B. Classes of Image-Processing Problems

This section presents the major results known about the limits to which various kinds of image-processing problems can be sped up by parallel processing. Problems are grouped according to the coherence of the corresponding mapping between input and output. Arranged in this way, an interesting pattern emerges from these results – the lower-bound complexities due to data transmission are the same for all problems in any coherence class (table 2.1.1). And these lower bounds prove to be the dominant factors in the complexity of many image-processing problems.

i) Zero-linkage Problems

By definition, zero-linkage problems have no interaction between a zone and a location that is sufficiently far away. These are exactly the problems best handled by local operations. The simplest of these are *local measurements*, i.e., the uniform application of a spatially-limited template across the image. These include pointwise remappings of intensity (e.g., gamma correction) and convolutions by functions of limited spatial extent. More generally, zero-linkage problems include those that can be solved using properties of *fixed support*, i.e., where the property can be extracted from a fixed set of points in each zone [Ull84]. The limit on transmission distance means that each PE can complete its operation within a fixed time independent of image size, and so these problems have $O(1)$ complexity. The limited time and spatial extent also mean that each PE need only be a finite-state automaton. A natural architecture for a zero-linkage problem is therefore a simple array in which each finite-state PE takes its input from the corresponding zone in the image.

Although communication time is minimal in an array, an extensive amount of wiring is usually required to connect pixels to their PEs, especially if the zones are large. Furthermore, such a network would be impossible to reconfigure when a different zone size is required. These drawbacks are largely eliminated by using a mesh. Here, the input is partitioned into nonoverlapping sections, with each PE taking its input from a single section. Since PEs do not generally have direct access to all information in a zone, information must be transmitted through the mesh. In essence, a mesh trades off time for space.

For zero-linkage problems, the natural mesh architecture is the *cellular automaton* [TM87, CHY90, TM90], for which the processing elements are simple finite-state automata. By limiting the number of iterations allowed for each PE, a cellular automaton can carry out zero-linkage problems such as spatial filtering [PD84]. As the PEs are given more power, they are able to combine simple measurements in interesting ways — for example, to determine the color or orientation of line segments by comparing the magnitudes among a basic set of local measurements (see, e.g., [Gra85]).

ii) Constant-linkage Problems

Constant-linkage problems are characterized by a positive linkage whose strength does not depend on the size of the zone. Two variants can be distinguished: unidirectional and bidirectional.

Unidirectional problems

One of the simplest unidirectional problems is to determine the average value of the pixels in an image. As discussed in section 2.1.1, determining this quantity requires only one parameter (the sum of pixel intensities) from any zone. Similarly, the calculation of the standard deviation is also unidirectional, requiring two parameters (the sum of pixel intensities, together with the sum of their squares) to be obtained from each zone. Other problems which can be formulated this way include finding the minimum distance between black (or white) pixels in the image [Sto87], determining the center of mass [Tan84], and detecting horizontal or vertical concavities [Sto87].

All these tasks can be carried out in $O(\log n)$ time on a pyramid architecture [Tan84, Ros86, Sto87]. In general, logarithmic complexity can be achieved for any process in which each PE reduces the data from the level below it to a constant amount, and passes this data upwards [Sto87]. The pyramid is consequently a natural architecture for the entire class of unidirectional constant-linkage problems. Hypercubes allow no additional reductions of complexity.

Bidirectional problems

Relatively little work has been done on this class of problems. Determining the median can be done in $O(\log^3 n)$ time on a pyramid; it is not known whether this quantity can be lowered [Sto87]. Another bidirectional constant-linkage problem is the determination of extreme points, i.e., those points located at the corners of the smallest convex polygon containing all points in the image. The complexity of this problem also is $O(\log^3 n)$ on a pyramid [Sto87]. It may be that this (provisional) limit applies to all such problems.

iii) Perimeter-linkage Problems

A large set of problems can be characterized by linkage proportional to the perimeter of the zone. Again, both unidirectional and bidirectional variants can be distinguished.

Unidirectional problems

This class is exemplified by connected component labelling (CCL), where each distinct component in the image is to be assigned a unique label. Note that a special case of this problem is the determination of whether all lines in the image are connected. Provided that the components passing through the perimeter of a zone are correctly labelled (as far as the zone is concerned), no other aspect of the zone's contents are needed to solve this problem.

The number of degrees of freedom is therefore equal to the number of perimeter crossings. Assuming a uniform distribution of components in the image, this is directly proportional to perimeter length. Another perimeter-linkage problem is the determination of the lengths of all lines in the image. Here, two parameters (label and total length inside the zone) are required at each perimeter crossing.

CCL can be done in $\Theta(n)$ time on a mesh [Sto88], $\Theta(n^{1/2})$ time on a pyramid [MS87], and $\Theta(\log(n))$ time on a hypercube [LAN89]. This latter limit is the same when a PRAM is used [SV82]. It may be that these limits also apply to the other problems in this class.

Bidirectional problems

This class includes problems of constraint relaxation,¹² which are characterized by local measurements that require context for their complete interpretation [HZ83, KI85]. Since these problems depend only on local constraints, the effect of a zone on the result is determined by a band of pixels along the border, the exact width depending on the range of the local process. Linkage is consequently proportional to the length of this border. But values in the local zone must also be consistent with their surroundings, making the problem bidirectional.

Two types of relaxation problem exist: continuous and discrete. For continuous relaxation, local values (as well as interaction terms) are represented as real numbers. Among other things, this allows non-zero probabilities to be assigned to different interpretations of any local feature. The problems themselves are generally formulated in terms of maximizing or minimizing some global quantity, which then allows them to be recast as finite difference equations [HZ83]. One particularly interesting set of problems involves reconstructing surfaces by finding the extremum of some global measure such as the smoothness or error of the reconstructed surface. This approach is the basis of general frameworks of visual processing such as regularization theory [PTK85] and Markov random fields [GG84].

Continuous relaxation can also be formulated in terms of linear programming [BB82, p. 420–430]. Since linear programming can be done in polynomial time [Kar84], it is likely that continuous relaxation is of this complexity. For problems that can be cast as the solution of elliptical equations (either linear or nonlinear), the number of iterations required to solve the problem to within a given accuracy is $O(n^L)$, where L is the order of the equation [Bra77, p. 281]. On a pyramid architecture, where multiresolution techniques [Bri87] can be used,

¹²These are often referred to as relaxation *processes*. They are described here as *problems*, however, since they are abstract specifications of input-output mappings that are quite independent of the particular processes used to carry them out.

this is reduced to $O(n)$ iterations [Gla84].

In contrast to continuous relaxation, discrete relaxation requires that values assigned to pixels be integers, and that only one interpretation be allowed for each object. Many of these problems are NP-complete, including the interpretation of line drawings [KP85]. It is strongly suspected (although not proven) that NP-complete problems take an exponentially large amount of time in the worst case [GJ79]. Assuming this to be true, the complexity of discrete relaxation results more from the cost of search than from bottlenecks on data transmission.

iv) Area-linkage Problems

Finally, problems exist for which linkage is proportional to the area of the zone. These *area-linkage* problems have minimal coherence between values in the input and the output at any location. An example is the rotation of a discrete image by 180° . Here, a change in one part of the input can change the output at a position arbitrarily far away. Another example is the sorting of pixel intensities. Here again, a change in the value of a single pixel can lead to changes at locations far removed from the original zone.

Area-linkage problems involve such large amounts of data transmission that pyramid architectures (and variants thereof) cannot efficiently handle the transmission of data. Bottlenecks exist at the higher-level PEs of the pyramid, and so considerable time is therefore required to move data over large distances. Image rotation, for example, is of complexity $\Omega(n)$ on a pyramid. A similar limit exists for sorting [Sto87]. This latter value may a general limit for area-linkage problems on this architecture.

When the communication bottlenecks are bypassed by the use of more completely-connected architectures, the complexity of area-linkage problems is reduced. For example, sorting on a hypercube requires $O(\log n)$ time [Sto87], the lowest complexity possible on any architecture [GR88].

2.1.2 Psychophysical Studies

Early vision consists of those operations in the human visual system that are rapid, spatially parallel, and require little attention. Since these operations are directly involved with the incoming image, they are relatively easy to study empirically. Consequently, they have long been the subject of psychophysical investigation (see, e.g., [Zuc87b]).

Since the focus here is on the limits to this kind of processing, this section surveys only the results of psychophysical studies on the descriptions used at the highest stages of early vision. As this survey shows, there is a remarkable degree of convergence to these results.

A. Basics

Psychophysical studies of rapid visual processing have largely been concerned with those activities that occur almost instantaneously and without conscious effort. For example, when a horizontal line is placed among a group of vertical lines, it invariably “pops out” of the image, no matter how many vertical lines there may be. On the other hand, detecting a T-shaped figure among L-shaped figures requires a much slower and more effortful serial scan of the display [Tre88]. This is generally taken as evidence that fast search is based on “visual primitives” formed rapidly and in parallel across the visual field, while slow search is based on constructs formed serially at higher levels.

This difference in performance (in both accuracy and response time) can be used to determine the set of properties determined rapidly and in parallel in early vision. For the most part, experiments have been based on one of three types of task: visual search, texture segmentation, or grouping.

i) Visual Search

In visual search, the task is to determine whether a displayed image contains a subset of some given collection of target patterns. Performance is generally measured by the accuracy or speed of the response. Psychophysical studies explore how this performance varies as a function of the number and type of target patterns in the collection, the number and type of items in the display, and the duration of the display itself (see, e.g. [Rab78, Rab84]).

Visual search experiments date back to the work of Green and Anderson [GA56], who demonstrated that search speed for a target was unaffected by variations in the shapes of the other items, except for those of the same color. This suggested that color is available at early levels to allow the selective processing of visual information. Further work by Neisser [Nei63] showed this to hold for simple geometrical properties as well: target letters embedded in a group of nontargets are detected more quickly when they have a distinctive shape or orientation.

More recent studies (e.g., [Tre82, Tre88, Dun89]) measure response time for a single target as a function of the number of items in the display, with response accuracy being held constant. These experiments show that if the target is sufficiently distinct from the other items, response time is effectively independent of the number of items present — subjectively, the target “pops out” of the display. Otherwise, detection time is roughly proportional to the number of items in the display, with the constant of proportionality being twice as large for target-absent displays as for target-present ones. This latter pattern is taken as evidence for a serial scanning process that terminates when the target pattern has been found [Rab78, Tre82, Tre88].

ii) Texture Segmentation

An alternative way to investigate rapid parallel processing is based on the perception of visual texture. Texture perception has several different aspects. These include obtaining surface shape from texture gradient, determining the intrinsic structure of a surface, and finding the boundaries between regions of different texture (see, e.g., [Wil90]). Much of what is known about texture perception is based mostly on studies of this latter aspect, called *texture segmentation*. In particular, studies have concentrated on finding the determinants of “effortless” segmentation, i.e., segmentation occurring within several hundred milliseconds of initial viewing and with no conscious scrutiny (e.g., [Jul81]).

Segmentation itself has several different aspects, including the detection of regions of different texture, and the determination of the shape of the possible boundaries [Wil90]. For the most part, experiments proceed either by measuring the time required to perform these tasks to within a given accuracy or by measuring performance accuracy as a function of display time.

In both cases, a pattern of results is found that is much the same as that for visual search. Textured regions can be separated effortlessly from each other when they differ sufficiently in the density of their elements, or if these elements are sufficiently distinct from each other (e.g., a region of horizontal lines against a region of vertical lines) [Jul86]. It must be kept in mind, however, that texture segmentation is a process with goals that are in many ways different from those of visual search, and so may not necessarily involve the same set of elements.

iii) Visual Grouping

The representations used in early vision can also be studied by finding the determinants of visual grouping [Bec66, Bec82]. This approach has its origins in the Gestalt laws of grouping. Disconnected elements can be grouped together into larger units (such as lines and regions) on the basis of similarity and spatial organization [Zuc87a]. Turning this around provides a way to define these properties operationally — similarity and spatial organization are exactly those properties that lead to visual grouping.

Since studies of visual grouping often are based on the conscious perception of grouping strength (e.g., [Bec82, SBG89]) and not on processing speed or accuracy, their results do not necessarily pertain to rapid parallel processing. But it has been found that “spontaneous” grouping is not based on the overall shapes of objects, but rather on the similarity of their “elementary parts” [Bec82]. To the extent that these parts are consistent with the elements of visual search or texture perception, they can provide a check on the descriptions formed at early levels of vision.

Three types of grouping are commonly studied: (i) segregation into regions, (ii) segregation into populations, and (iii) creation of intrinsic surface structures. The first of these is similar to texture segmentation. But, whereas segmentation is generally concerned with the *boundaries* of textured regions, segregation focuses on the linking of items into distinct *regions*.

Population segregation is similar in most ways except that linking based on proximity in the image is replaced by linking based on proximity in a more abstract space of intrinsic properties (e.g., color or orientation). Thus, for example, a group of yellow dots intermixed with blue dots can be separated into two distinction populations, even though no geometrical boundaries exist. Experiments are generally based on judgements of whether two or more kinds of features are scattered throughout the image. Although it is also sometimes termed “texture segmentation” [Bec82], this task is conceptually quite different, involving the pooling of visual elements based on their intrinsic properties rather than their locations.

The third type of grouping is the formation of “intrinsic” structures, such as one-dimensional contours or two-dimensional flow patterns, which can arise even in images formed only of dots [Ste78, ZSS83]. Like the segregation of elements into regions or populations, this process is generally thought to be based on simple properties computed over local zones in the image [Bec82].

B. Models of Rapid Visual Processing

The general pattern of results from experiments on visual search, texture segmentation, and grouping is much the same: performance is governed by a small set of simple image properties such as line orientation, curvature, contrast, and color (see [TG88]). The explanation of these results, however, is far from straightforward. First, the limited range of conditions over which data is collected can make it difficult to determine whether a process requires constant, logarithmic, or linear time. And there is no necessary connection between those properties that are computed quickly and those that are computed in parallel — a description may be the result of an extremely fast-acting serial mechanism, or conversely, a parallel mechanism may still require time that increases with the size of the input [Tow72].

In spite of these reservations, several theories have been proposed to explain many aspects of the results. Although differing in details, these theories agree that simple properties are computed rapidly and in parallel at an early “preattentive” stage, and that complex properties require the application of more sophisticated serial operations at a subsequent “attentive” stage of processing [Bec82, Jul86, Tre88].

i) Feature Integration Theory

Feature-integration theory [Tre82, TG88] was originally developed to explain why “pop-out” in visual search occurs when the properties of targets differed sufficiently from those of nontargets, but not when they differed only in the spatial arrangement of their parts. According to this theory, the preattentive system is composed of a set of parallel spatiotopic maps, each describing the distribution of a particular property (or “feature”) across the visual field. These features are simple properties of the two-dimensional image, including orientation, curvature, binocular disparity, color, and contrast [TG88]. Once these maps have been computed, a target containing a unique feature can be detected simply by checking for activity in the relevant map [Tre88].

The separation of the maps, however, means that spatial relations between features cannot be represented explicitly. Instead, the coherence of items is represented indirectly via a “master map” linking together the appropriate locations in the feature maps. The computation of complex structures therefore requires a *spotlight of attention* to access the master map and link up all the relevant features into a coherent whole. Since this spotlight must serially inspect each collection of features present in the image, the detection of complex features requires time proportional to the number of features present. This explains why, for

example, targets distinguished only by inside/outside relations do not pop out [TG88].

In its original form, feature-integration theory did not account for several phenomena. Among these were the finding that conjunctions of simple features at the same location can be rapidly detected when their constituents are strongly discriminable, and the finding that search rate increases smoothly with the discriminability of the stimuli [Tre88]. The first of these has since been explained by postulating an inhibition (or excitation) of the master map at locations where elements are strongly activated. This allows all items containing nontarget features to be effectively ignored, leaving a small remainder among which the target can be quickly detected [Tre88, CW90, TS90]. The second effect is accounted for by postulating that the spotlight of attention operates not on individual items but on *groups* of items, the size of the group varying with the discriminability of its members [TG88]. Both these refinements, however, maintain the assumption that only simple local operations are carried out in parallel at preattentive levels.

ii) Resemblance Theory

Resemblance theory [DH89, Dun89] is an alternative account of visual search that differs from feature-integration theory in several ways. It shares the basic premise that simple features are computed at the preattentive level but postulates that the speed of search depends entirely on the resemblance between the target and nontarget patterns in the image. It explains the relatively slow search for conjunctions as due to the similarity of target and nontarget items arising from their common features.

One of the more interesting aspects of this theory is that resemblance is based on the degree of transformation needed to map the features of one figure into those of another [DH89]. It therefore is a first step away from the idea that preattentive processes are necessarily based on simple local properties. Although some of the difficulty of conjunction search is apparently due to conjunction itself [Tre91], the possibility remains that some aspects of preattentive operation are best explained in terms of features resulting from procedures applied to simple line elements.

iii) Texton Theory

In contrast to both feature-integration and resemblance theory, texton theory was developed to account for effortless texture segmentation. Here, perceived texture is thought to depend entirely on the first-order densities of spatial patterns called *textons*. These are

localized geometric shapes with simple properties, including endpoints, elongated blobs, line crossings, and line segments of various lengths, widths, and orientations [Jul84a].

Texton theory explains texture segmentation by a model similar to those used for visual search, with processing being separated into distinct preattentive and attentive systems. The preattentive system is composed of a set of spatiotopic maps, each describing the distribution of a particular texton across the visual field. Effortless segmentation occurs when the regions differ sufficiently in the first-order densities of their constituent textons. Because only texton *densities* are involved, textures cannot be effortlessly segmented when they differ only in the relative arrangements of their textons (e.g., a region of L-shaped figures against a region of T-shaped figures). To separate such regions therefore requires conscious "scrutiny" by higher-level processes [Jul84a].

Textons have much in common with the set of features postulated for visual search. They include not only length, width, and orientation, but also color, motion, binocular disparity, and flicker [Jul84a]. Indeed, given that line-crossings are no longer considered to be true textons [Not91], the two sets appear to be almost identical. Textons have even been used to explain visual search itself, using a mechanism analogous to the spotlight of attention being postulated to account for the detection of particular texton combinations [JB83]. Like feature-integration theory, texton theory has also been revised to allow groups of items to be searched in parallel within limited regions, the size of these regions varying with the strength of the density gradient [Jul87].

But important differences also exist. Whereas feature-integration and resemblance theories are based on the absolute presence or absence of features, texton theory posits boundaries based on the local *differences* between texton densities [Jul86]. Furthermore, while most (if not all) textons have properties similar to those of preattentive features, they are quite different ontologically: textons are geometric elements containing specific properties, and are not the properties themselves. In essence, each texton contains a conjunction of simple properties. Thus, although effortless texture segmentation cannot be based on spatial relations, it *can* be based on the conjunction of simple features [Jul84b].

iv) Spatial Filtering

Recent attempts to provide an algorithmic framework for texture segmentation have shown that much of it can be explained in terms of the spatial filters postulated for edge detection, viz., localized linear filters of differing widths and orientations (e.g.,

[Cae84, BA88, VP88]). It also has been suggested that the texture boundaries themselves are determined via operations analogous to those used for edge-detection, with the array of filter outputs being smoothed and the lines of maximum change then used to mark the texture boundaries [VP88, GB89, BCG90]. Direct psychophysical evidence has been obtained in favor of this view [Not91]. As a consequence, there is now some doubt about the need to maintain textons as a separate set of texture primitives (see, e.g., [Not91]). But consensus remains that texture primitives — whatever these ultimately may be — are based only on simple local properties computed rapidly and reliably from the image.

The spatial-filter model also helps to explain the grouping of image elements. Such filters respond not only to actual lines of a given orientation and length but also to simple structural groups having the same general outlines, such as the “virtual lines” formed by a row of dots of similar contrasts [Zuc86]. However, although filters are thought to be necessary for the grouping process, they are not usually believed to be sufficient. Apart from exceptions such as the rapid detection of “locally parallel” structure in Glass patterns [Ste78]), grouping processes are generally thought to require nonlocal integration of information across the image (e.g., [Zuc87b]).

It also appears unlikely that the properties determined at the preattentive level can be explained entirely in terms of a single set of filter-based elements. For example, population grouping is based on the lightness *differences* of the elements, rather than by the contrast ratios that govern region segregation. Furthermore, conjunctions of these properties do *not* support population segregation, whereas they do support region segregation [BGS91]. The two sets of processes cannot therefore involve the same set of basic elements. Further support for this view comes from studies that show texture identification and texture segmentation to be based on different sets of primitive elements [Not91]. Thus, given the possible existence of several different sets of preattentive elements, it is likely that at least some of them are not directly related to spatial filters.

2.1.3 Computational versus Psychophysical Studies

From a computational viewpoint, there are good theoretical grounds for the assumption of a distinct stage of early visual processing. To begin with, almost all the properties at this level have two important characteristics: (i) they are zero-linkage (section 2.1.1),¹³ and

¹³As a convenient way of speaking, the linkage of a property is identified with the linkage of the corresponding image-processing problem.

(ii) they have a fixed support, i.e., the relevant property can be extracted from a fixed set of points in the zone. A spatially-bounded template can therefore determine the relevant property at each point and the corresponding map can be computed rapidly and in parallel. Although recent experiments have shown that conjunctions of preattentive features can pop out when sufficiently distinct [TS90], this has little effect on the general argument since, as several models have shown (e.g., [CW90, TS90]), this can be accounted for entirely by a more sophisticated search mechanism that selectively suppresses (or excites) the outputs of the simple feature maps.

In contrast to these “template” properties, others are neither zero-linkage nor have a fixed support. For example, determining whether a given object is inside or outside a neighboring object cannot be done within some fixed zone, since there are no limits to the extent of the neighbor’s boundaries. Even if limits were imposed, there would still be no fixed points which could always be used. Thus, a different template is required for each of the exponentially-increasing number of possible shapes.¹⁴ It has therefore been suggested that “nonlocal” properties, including virtually all types of grouping and spatial relations, are determined *procedurally* via specialized *visual routines* applied to earlier “base” descriptions [Ull84]. Many of these routines are serial, spatially inhomogeneous, and are thought to be controlled by higher-level processes. As such, their application is sometimes linked (to greater or lesser extent) with the spotlight of attention required at attentive levels [Ull84, TG88]. This point of view receives some confirmation from the finding that spatial relations such as parallelism and inside-outside cannot be detected preattentively [TG88].

However, this grouping of visual processes into distinct early and later stages is not without its difficulties. Consider first the property of length. This is generally regarded as a primitive quantity, both in empirical studies on visual search (e.g., [TG88]) and in computational models of early vision (e.g., [Mar82]). But length is not a zero-linkage property — a gap arbitrarily far away can change the value assigned to a line. It also is not easily determined by a template, or even a set of templates along the line, since the value from any individual template depends on the overlap between it and the line being measured. At best, length might be determined from competition among the set of templates along the given line (cf. [Zuc87a]) but this begins to introduce a nonlocal element into the computation.

It also has been found that binocular disparity (and possibly depth) can be determined

¹⁴For example, consider a surface patch divided into n intervals. If k possible values (e.g., color, height) exist for each interval, then k^n different combinations are possible.

preattentively [NS86]. It is possible to call disparity a zero-linkage property, in the sense that the value at any point depends only on some finite surrounding zone in the image. But there is no way in which it can be given a compact fixed support — to ascertain disparity requires the matching of patches in the left and right images, and the contents of these patches can be quite arbitrary. Matching must therefore be done procedurally.

Recent results have also shown that the preattentive system can determine properties such as direction of lighting and three-dimensional orientation — properties not of the image, but of the *scene* to which it corresponds [Ram88, ER90a, ER90b, ER91]. Such recovery appears to have a nonlocal procedural component, since its success does not depend completely on the presence or absence of any particular local property, but instead depends on the entire *system* of line relations present in the item [ER90b, ER91, ER92]. These findings call into question the basic assumptions behind the conventional assignment of visual processes to early and later levels. In particular, they call into question the reasons for believing that three-dimensional structure cannot be rapidly determined by a spatiotopic array of parallel processors.

2.2 The Interpretation of Line Drawings

The problem of line interpretation is one that is simple enough to allow easy formulation and experimental manipulation, yet complex enough that its solution requires at least some degree of intelligence. As such, it provides an interesting arena in which to study processes of a type generally thought to be restricted to higher levels of cognition.

This section surveys some of the main results of the computational and psychophysical studies that have been carried out in this area. In particular, it examines the case where the drawings correspond to two-dimensional projections of opaque polyhedra. Some of the more important theoretical results are first surveyed. This is followed by an overview of what is known about the ability of humans to interpret such drawings.

2.2.1 Computational Studies

The problem of determining the three-dimensional structure of an object from its corresponding line drawing has been the focus of a considerable amount of work in the field of computational vision (see, e.g., [CF82]). This section reviews several important results that have been obtained. These results have for the most part been developed within a single

framework – the *blocks world*. When the basic assumptions of this framework are met, a great deal can be said about what can and cannot be recovered from a line drawing.

A. Basics

Early work on the machine interpretation of line drawings (e.g., [Rob65]) attempted to analyze scenes composed of a small set of known polyhedra. The goal was to identify lines in the image with edges of particular instances of these objects. Recognition proceeded by using *a priori* knowledge of the polyhedral shapes to determine which image regions corresponded to which surfaces.

Although research in “model-based” vision (e.g., [Low85]) continues to use such global constraints, attention also turned to the use of “local models”, i.e., constraints on the local structure of the objects in the scene. Guzman [GA68] showed that the structural relations among the lines of the junctions were often sufficient for the extraction of three-dimensional structure. Subsequent work (e.g., [Clo71, Huf71, Wal72, Mac76]) gave this approach a more solid theoretical framework in which to formulate and discuss issues of line interpretation.

This theoretical framework was based on the so-called “blocks world”, a scene domain comprised of polyhedral objects with trihedral corners, i.e., corners formed from the intersection of three polygonal faces (see, e.g., [CF82]). The corresponding image domain is formed by the orthographic projection of these objects onto the image plane. Given that corners are trihedral in the narrow sense (section 1.1), this projection consists of straight-line segments connected by junctions of either two or three lines. By using line drawings alone, all effects of surface coloration (e.g., reflectance and texture) and shading are discounted. Viewing direction and direction of lighting are held constant, with the two directions often being made coincident in order to avoid shadows. The result is a “miniworld” in which attention can be focused entirely on the recovery of surface geometry.

The most comprehensive, and difficult, problem concerning line interpretation in this miniworld is that of *realizability*: Given a line drawing, does it correspond to an actual arrangement of polyhedral objects in some three-dimensional scene? If so, what are the three-dimensional shapes and positions of these objects? Virtually all the work done on the blocks world has proceeded by splitting this problem into two parts: a qualitative aspect concerned with the structural relations between edges and surfaces in the scene, and a quantitative aspect concerned with the slants of the lines and the depths of the vertices [Sug86, KP88].

Although both aspects can be approached independently, the results of qualitative analysis have usually been used as the starting point for quantitative analysis (see, e.g., [Sug86]).

B. Qualitative Interpretation

Since the faces of a polyhedral object are planar, its structure is completely determined by the locations of its edges, i.e., locations where the orientation of adjoining faces suddenly changes. Qualitative analysis is based on the structural relations between these edges (see, e.g., [Mal87]). Edges can be subdivided into convex and concave forms according to whether or not the edge folds outward, i.e., whether or not an external plane can be placed into contact with the edge. A further subdivision results from the relation between object and viewer. Edges can be grouped according to whether both or just one face is visible.¹⁵ This latter kind of edge is referred to as a *boundary edge*. These correspond exactly to places in the viewer-centered description where depth changes discontinuously. Two types of boundary edge are often distinguished, according to which side of the line corresponds to the visible face. The remaining edges are referred to as *interior edges*. These correspond to locations where the depth gradient changes discontinuously. These are divided into two types, according to whether the corresponding edge is convex or concave. Between them, boundary and interior edges describe not only an object's shape but also the segmentation of the image, i.e., which regions in the image do or do not correspond to connected surfaces in the scene (see, e.g. [Sug86, Kan90]).

To interpret a line drawing, each line must be labelled as a particular kind of edge (convex, concave, or boundary). The interpretation is guided by a set of explicit constraints on the various edge labellings. These constraints can be provided largely by restrictions on the labelling of junctions [Huf71, Clo71]. Four types of junction can be distinguished (figure 2.2). The first three are trilinear, formed from the joining of three lines: (i) arrow-junctions, for which the greatest angle between two lines is greater than 180° , (ii) Y-junctions, for which it is less than 180° , and (iii) T-junctions, for which it is exactly 180° (see figure 2.2). There also exist L-junctions, formed from the joining of two noncollinear lines. Each type of junction leads to a particular set of constraints. These constraints, first given by Huffman [Huf71] and Clowes [Clo71], are shown in figure 2.3. These correspondences fail to hold when there is an accidental alignment of viewing direction with particular arrangements of surface edges. Since accidental alignments are exceedingly rare, the assumption usually is made that they

¹⁵It is evident that this distinction applies only to convex edges.

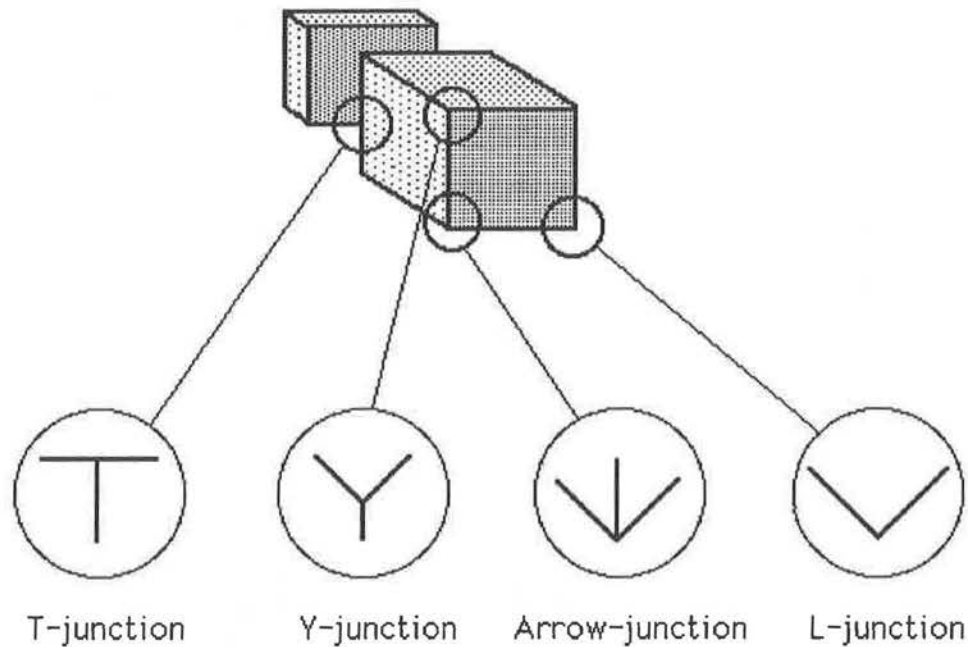


Figure 2.2: Types of junctions.

do not occur; under this *general viewpoint constraint*, the correspondences between junction and edge types will always hold.

The problem of finding a consistent set of labels for a given drawing is known as the *line labelling* problem. Every polyhedral scene gives rise to a unique set of labels [Ric88], and if a drawing is realizable, it can be consistently labelled [Huf71, Sug86]. But the converse is not true. The separation into independent qualitative and quantitative components means that the metric structure of the scene is not available to the qualitative labelling process. Consequently, line drawings can be consistently labelled, but have no correspondence with any polyhedral object [Huf71, Kan90].

The labelling of a given drawing can be carried out by a relatively straightforward procedure. As figure 2.3 shows, each type of junction can be labelled in several different ways. To reduce the number of local candidates, interpretation often begins with the application of “Waltz filtering” to eliminate labels that are locally inconsistent [Wal72, Mac77]. This kind of consistency check is a relatively simple procedure that can be carried out in polynomial time [MF85].

Waltz filtering finds a correct interpretation if the locally consistent labels are globally consistent as well. But this does not always occur. Consequently, it is sometime necessary to

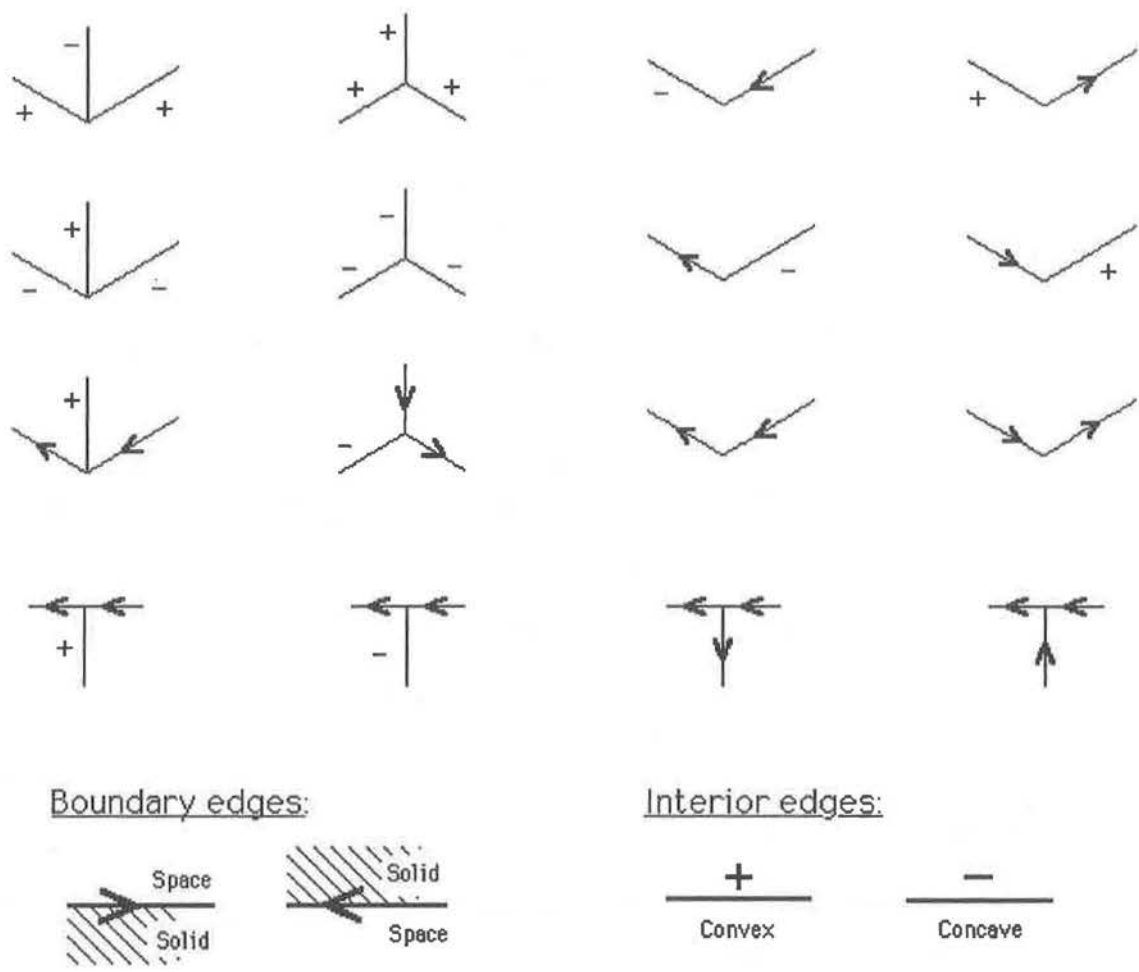


Figure 2.3: Huffman-Clowes labelling set.

explore all possible combinations of the remaining labels, each combination then tested for global consistency. Since the labelling problem is NP-complete [KP85], it is highly unlikely that a globally consistent solution can always be found in polynomial time. Instead, the worst-case time is likely to be an exponential function of the number of junctions (and lines) in the image.

Although the Huffman-Clowes constraints never lead to inconsistency in a drawing that corresponds to a physically realizable object, they sometimes consistently label an object that cannot be realized. Two types of error occur: inconsistencies in the global topological structure, and inconsistencies in the depths of the surfaces (see, e.g., [Dra81, Kan90]). Topological inconsistencies can be eliminated when all corners are rectangular [Kan90]. Inconsistencies in depth, however, must be handled via more powerful constraints based on the metric structure of the scene.

The use of metric constraints was pioneered by Mackworth [Mac73], who developed an approach based on the observation that regions in the image must correspond to flat planes in the scene. Each plane is represented by its gradient, a two-dimensional measure of its orientation in space. Since all faces of a polyhedral object are planar, its coherence can be captured by constraints in the gradient space, which eliminate many inconsistent interpretations.

Although constraints on gradient space are useful, they do not eliminate all inconsistent interpretations [Dra81]. This is because only partial use is made of three-dimensional information — gradients ignore the fact that planes are also specified by their depth along the line of sight. This latter quantity forms the basis of sidedness reasoning [Dra81], in which constraints are based on the condition that one plane must always be in front of the other on a given side of their intersection line. The resulting set of constraints then ensures that all consistent interpretations correspond to physically realizable objects [Dra81].

C. Quantitative Interpretation

An alternative to qualitative line interpretation is to work directly with the quantitative structure to obtain the depths and the three-dimensional orientations of the objects in the scene. This technique, first suggested in [Fal72], is based on the observation that the junctions around a common region correspond to points and edges around a common planar face. This plane can be described by a linear equation, with the unknowns being the depths of the corners in contact with the face. Collecting the equations for each region in the drawing yields a system of linear equations, which can be solved by straightforward means [Sug86, Kan90].

In general, these systems of equations are underdetermined. Thus, even when an absolute depth is attached to one point, several degrees of freedom still remain [Sug86]. Additional constraint on the solutions is therefore required. One such constraint is an *a priori* specification of the three-dimensional orientation of particular faces. Since each of these specifications is independent, the number of degrees of freedom is reduced by the number of orientation specifications that can be given.

Other kinds of local constraint also are possible. If a junction corresponds to a rectangular corner, the slant and tilt of the corresponding faces and edges are completely determined by the angles of the lines about its vertex, the values depending only on whether the junction is concave or convex [Per68, Mac76, Kan90]. Furthermore, there exists a set of necessary (but not sufficient) conditions on a junction that corresponds to a rectangular corner [Per68]: an

arrow-junction must have one angle greater than 90° and the other two less than 90° , while a Y-junction must have all its angles greater than 90° [Per68, Kan90]. Three-dimensional orientations can also be recovered when only two of the angles are 90° [Kan90]. In this case, the lines must be correctly identified with the corresponding edges in the scene. Recovery of slant is possible for both orthogonal and perspective projection of the object onto the image plane [Kan90, ch. 8].

Global constraint also is used. One approach is to specify the recovered surface as the smoothest of all possible candidates [Kan90, ch. 10]; this loosely corresponds to the regularization technique suggested for several aspects of early vision [PTK85]. More generally, there is an interplay between local and global constraints. For example, Mulder & Dawson [MD90] have shown that for some objects, a complete quantitative interpretation requires that only a subset of corners be rectangular. This essentially is a special case of maximizing the rectangularity in the recovered figure.

2.2.2 Psychophysical Studies

In contrast with the work on computational aspects of line interpretation, work in psychophysics has been rather heterogeneous. It encompasses a wide variety of experimental methodologies and stimuli, as well as different theoretical frameworks. However, there is wide agreement in the general pattern of experimental results, and these patterns also are consistent with many of the results from computational studies.

A. Basics

Investigations into the perception of line drawings extend back to the very beginnings of experimental psychology. The first comprehensive explanation of how drawings could be perceived as three-dimensional objects was given at the turn of the century by Mach, who proposed that the visual system operates on a “principle of economy” (see [Att82]). This gave way to the Gestalt principle of figural “goodness”, which selected those interpretations that required minimal “energy” for their representation (see e.g., [Hoc78, pp. 131-155]). According to this principle, a line drawing of a cube is perceived as a three-dimensional object rather than as a collection of two-dimensional lines because this requires less energy for its representation. Similar reasons also explained the tendency to perceive its sides and angles as equal whenever possible.

The vagueness of Gestalt laws eventually led to their abandonment by many workers in the field. However, the central insight remained that interpretation must involve constraints on the interpreted object. This provided the starting point for later investigations (e.g., [HM53, Att54]) which attempted to provide a more rigorous study of these constraints. As in the case of machine vision, these later studies can be categorized into two groups: those concerned with the qualitative aspects of line interpretation, and those concerned with its quantitative aspects. Studies in the first group focus on the factors that determine whether a line drawing is perceived as a set of lines or as a three-dimensional structure. Studies in the second group are concerned with the perception of the metric properties of the structure itself. Reflecting a bias toward viewing line interpretation as a “high-level” activity, both kinds of studies tend to rely on verbal reports of consciously perceived structure.

B. Qualitative Interpretation

In contrast with computational studies of qualitative structure, psychophysical studies have tended to focus on global aspects of the interpretation rather than local properties such as the convexity or concavity of individual edges. At least some of this emphasis likely is due to the legacy of the Gestalt school, with its emphasis on the minimum energy of the entire interpretation. One of the first attempts to put this approach on a more rigorous footing was the work of Attneave [Att54], who recast the principle of minimum energy into one of “maximal simplicity”, where simplicity was based upon the “information” contained in the percept. By identifying this information with that used in information theory, it was hoped to have a more objective basis for the rules of the interpretation process.

Since the absolute amount of information depends upon the coding scheme, such rules cannot be entirely objective. Nevertheless, a few general principles can be derived. For example, maximally simple structures have lines of equal length, symmetry about the origin, corners of equal angle, etc. In the case of a cube, this approach correctly predicts that its line drawing is interpreted as a symmetric structure with edges of equal length rather than as an asymmetrical set of lines of unequal length.

Although this approach could explain the perception of simple line drawings, it could not do so for more complex ones without imposing *ad hoc* rules on how various regularities could be traded off against each other [HM53, Att54]. In turn, this could not be done without the specification of a particular coding scheme.

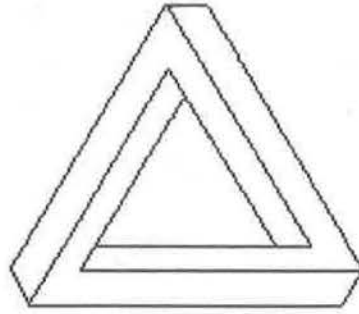


Figure 2.4: Penrose triangle.

Such schemes have been proposed (e.g. [Lee71]). If a large enough set of rules is imposed, it can indeed explain the perception of many kinds of line drawings [Res82, BL86]. But such “minimal description” approaches suffer from serious drawbacks. First of all, the emphasis on global measures means that a drawing that cannot be consistently interpreted must be represented as a two-dimensional structure. This is at odds with the finding that globally inconsistent figures such as the Penrose triangle (figure 2.4) are perceived as three-dimensional objects. (Also see, e.g. [Hoc78, pp 152-155].) It also is difficult to provide a plausible mechanism for finding minimal encodings. Even if this could be done, the search for the minimal description would still take considerable time [Att82, Res82]. Most importantly, perhaps, it is difficult to justify why the size of the description itself should be the main determinant of the process, rather than some property of the structure being described.

A rather different approach was taken by Weisstein [WM78], who investigated how various line drawings influenced the adaptation of the visual system to sinusoidal gratings. A simple blank hexagon placed on a grating extending over the entire visual field resulted in a complete lack of adaptation at the locations it covered. At these locations, relatively low-contrast gratings could be easily detected, although this was not possible in the rest of the visual field. But when a Y-junction was added to the hexagon, adaptation suddenly appeared in the blank field, as if that area had been “filled in” by the surrounding gratings. These results were explained by a tendency for the early visual system to perceive this figure as a cube, which was then separated from the flat background.

More generally, it was found that line segments can be more accurately identified when they are part of a drawing of a coherent three-dimensional object than when they are among a set of unstructured lines [WH74]. This was found to hold for junctions as well [BWH75], indicating that local properties govern this process. A similar set of results was obtained

by Walters, who found lines to be perceptually brightened when interpretable as edges of a coherent three-dimensional object. This brightening was found to be unaffected by global properties, depending only on junction type and line length (over distances of less than 1°) [Wal87].

C. Quantitative Interpretation

As computational studies show, line interpretation can be achieved via constraints on the quantitative structure of the recovered object. Interestingly, psychophysical studies suggest that several quantitative constraints are indeed involved in the human perception of line drawings.

One of these constraints is that of *rectangularity*, i.e., the requirement that polyhedral objects have sides at right angles to each other. The projection of rectangular corners yields junctions having a particular set of constraints on the angles between their lines (section 2.2.1). Empirical tests [Per72, She81] have shown that subjects are highly sensitive to these constraints, being able to determine accurately whether a line drawing does or does not correspond to a rectangular cube. This can be done even when some of the lines are removed from the figure, provided that at least one line is kept from each of the three orientations [Per82]. In contrast, subjects are far worse at recognizing which structures contain corners with angles of 60° or 120° [She81]. Consequently, it is likely that the critical factor is rectangularity rather than simple equality among the angles.

Rectangularity also makes it possible to determine the orientation of an object in three-dimensional space (see section 2.2.1). A very high correlation has been shown between actual slant and judgements obtained from line drawings of rectangular figures [AF69]. Although the perceived slants are less steep than the actual slants, this "flattening" can be lessened when contributions from other cues are reduced [Att72].

Another useful structural property is that of bilateral symmetry, i.e., symmetry about a plane through the center of an object. This property is generally perceived in a line drawing whenever it is consistent with the laws of projective geometry (e.g., [Per76, PC80, Per82]). Even when the task itself makes no use of it, subjects spontaneously perceive symmetry about half the time. Indeed, it is even possible to alternate between interpretations based on symmetry and rectangularity [Per76]. Not all kinds of symmetry can be detected, since equal angles of 60° or 120° are not generally perceived as symmetrical [She81]. The preference for

bilateral symmetry may have ecological origins — most animals are bilaterally symmetric [Per76].

Subjects can also detect the coplanarity of two planes in a line drawing, and although the accuracy for this is somewhat lower than for the detection of rectangularity, it is still quite good [Per82]. This shows that the human visual system can recover at least some quantitative spatial relations from line drawings.

Little is known about the mechanisms that carry out line interpretation in human vision. The two extremes have been proposed: “convergence” and “direct computation” mechanisms [PC80, Per82]. The former is essentially a general-purpose relaxation process (section 2.1.1) that can incorporate various constraints into its operation. Recovered properties are obtained as the computation settles into an equilibrium state. The latter is a special-purpose device that computes properties directly (i.e., in a non-iterative way), obtaining its speed at the price of decreased flexibility. There is insufficient evidence to determine which of these two processes (if either) is responsible for line interpretation, but the sensitivity of the visual system to several kinds of geometrical properties has been taken to support the existence of the general-purpose “indirect” process [Per82].

2.2.3 Computational versus Psychophysical Studies

As is evident from sections 2.2.1 and 2.2.2, there is considerable agreement between the results of computational and psychophysical studies in the areas where they overlap. According to computational models, there is enough information in the junctions to allow the recovery of almost all qualitative structure from a line drawing. This result is echoed in the finding that the perceived three-dimensionality of a line drawing depends on the types of the junction involved. The similarities extend to the quantitative aspect of interpretation as well, where the importance of structural constraints such as rectangularity has been established in both areas of study.

Agreement, however, is not the same as completeness — many aspects of line interpretation have not yet been investigated by either kind of study. For example, most computational and psychophysical studies have been based on perfect or near-perfect line drawings, so that interpretation in the presence of noise is a relatively unexplored domain. Another largely unexplored area is the complexity involved with interpreting various kinds of line drawings. In its most general form, the realizability problem (section 2.2.1) is NP-complete [KP85],

and so the interpretation of some drawings must sometimes be difficult and time-consuming, regardless of whether the system is artificial or biological. No studies, however, have explored the way in which complexity issues are handled by the human visual system. The fact that psychophysical studies are usually based on reports of relatively high-level percepts shows a tacit agreement that line interpretation requires sophisticated processing. But how then to account for rapid line interpretation in early vision?

Evidently, rapid interpretation must be possible for only a subset of the scene domain, one for which the time complexity is very low. A few subdomains of this kind are known to exist. One of these is the *orthohedral* world, where all objects are constrained to have surfaces parallel to the x, y , and z planes. Here, the labelling of n lines can be done in $O(n)$ time on a serial processor, and in $O(\log^2 n)$ time when n^3 processors are available [KP88]. But this result does not necessarily pertain to rapid recovery in human vision, since n^3 processors may not always be available, and $O(\log^2 n)$ time may not always be allowed. More generally, it is not clear which aspects of scene structure can be rapidly determined, or even which aspects should be. These questions can only be examined in the context of a computational theory.

2.3 High-level versus Low-level Vision

In order to develop a computational theory of rapid parallel recovery, it is necessary to know what role this process could play. This section re-examines the reasons for separating vision into high and low levels, and for the particular assignment of various processes to these levels. It then examines what can be expected of a rapid recovery process, and shows how it can help bridge the gap between the two levels.

2.3.1 The Structure of Low-level Vision

Information-processing tasks generally have aspects common to all inputs and aspects applicable only to special cases. In vision, these two aspects take the form of distinct levels: a “low” level based on the general constraints of geometry, physics, and information theory, and a “high” level based on the more specific relations between individual objects in the scene (e.g., [Mar82, Fel85]). The boundary between low- and high-level vision therefore reflects the limits of a “common core” believed to be derivable (usually in a bottom-up fashion) from general considerations alone.

Owing to the inverse relation between the generality of a constraint and the structural

complexity of the objects it applies to (see, e.g., [Sal85, p. 49]), this common core must involve structures of a relatively simple structure. In particular, the common core is usually taken to be a viewer-centered map (or set of maps) of various scalar properties of the image or scene, with possibly some explicit representation of structural grouping as well [BT78, Mar82].

This characterization of low-level vision differs from that of *early* vision, in that emphasis is placed on properties of the input-output mapping itself rather than on properties of the process that generates it.¹⁶ But processing speed is used (often implicitly) to decompose low-level vision into a sequence of “horizontal” modules. Each of these is concerned with a distinct stage of the computation, and is applied to the entire image (see, e.g., [Mar82, Ull84, Uhr87]). While there is no consensus on the exact structure of these stages, there is considerable agreement on their existence and general operation.

A. Early Stage

The first stage of low-level vision is generally identified with early vision, i.e., based on operations carried out rapidly and in parallel across the visual field. This is sometimes described as the “image processing” stage, since the representations for both input and output are generally arrays of pixels, usually with the same spatial dimensions [Ree84]. Early vision is believed to provide a quick initial analysis of the image, making explicit those properties useful for subsequent stages of processing (e.g., the locations and orientations of lines in the image) [Mar79, Mar82].¹⁷ Its primitive elements therefore describe properties that can be reliably determined in this way. Typically, this is done by the concurrent application of fixed templates to each point in the image (e.g., spatial filtering [Gra85]).

This early stage is common to virtually all computational models of low-level vision, taking on forms such as the “raw primal sketch” of Marr [Mar82], the “MIRAGE model” of Watt and Morgan [WM85, Wat88], and the “cortex transform” of Watson [Wat87]. Although the primitives used in these models differ in detail, they generally describe simple properties of the image, such as color, orientation, and spatial frequency. It has been recognized (e.g. [Mar82]) that primitives should describe properties of the scene whenever possible (e.g., using

¹⁶This distinction between early and low-level vision is not one that is usually drawn. However, it helps to illustrate one of the points being made here, viz., that computational models must incorporate issues of resource use.

¹⁷Interestingly, in his earlier work, (e.g. [Mar79]), Marr emphasized that “there seems to be a clear need for being able to do early visual processing roughly and fast as well as more slowly and accurately” [Mar79, p. 31]. This idea became less prominent in later work.

contrast to obtain changes in surface reflectance). But scene-based properties that can be reliably determined with templates are few and far between. For the most part, a complete determination of scene properties requires subsequent stages of processing that employ more sophisticated and time-consuming operations.

B. Later Stages

There are many ways to associate properties of the scene with primitive image elements (see [dYvE88]). Because of this, and because of the shortage of relevant information from psychophysical and neurophysiological studies, there is no general consensus as to how subsequent processing is carried out.

One possibility is that reconstruction is based directly on the image elements, using constraints derived from the nature of the scene and the way it is projected to the image plane. This is sometimes assumed to be done via separate streams for each kind of visual medium (e.g., information obtained via luminance, motion, or texture) or for different scene and image properties (see, e.g., [CAT90]). But although some recovery processes can be carried out almost immediately when parallel processing is available (e.g. the recovery of three-dimensional surface orientation via photometric stereo [Woo81]), much more time is generally required. For example, the interpretation of line drawings is an NP-complete problem [KP85], which effectively rules out the possibility of always speeding it up sufficiently by parallel processing alone (section 2.1.1). Recovery processes described by the frameworks of regularization theory [PTK85] or Markov random fields [GG84] also are relatively time-intensive, typically requiring several thousand iterations for images of moderate size (e.g., [Bla89]). Furthermore, their close association to relaxation problems makes it likely that the time required increases at least linearly with the size of the input.

An alternative approach is to build up the scene descriptions more gradually, via an intermediate stage containing “non-template” properties that can be determined quickly. For example, the primitive elements of Marr’s raw primal sketch [Mar82] are grouped together on the basis of local properties (e.g., common orientation) to form higher-level symbolic structures. This grouping is done recursively, so that highly complex elements can be built up. The result is a “full” primal sketch that is available to subsequent processes, such as those involved with texture segregation [Mar82].

Ullman [Ull84] has suggested that many spatial relations (including those described in

the full primal sketch) are obtained via the application of visual routines to the elements of early vision (section 2.1.3). These routines are based on a small set of simple operations such as marking and propagation, which are then concatenated together to form the desired procedure. This allows many “non-template” properties to be extracted from the image in time proportional to the size of the input. But although the complexity of these strategies is relatively low, the linear complexity bounds are still insufficient for many purposes, especially if images are large and complex. Furthermore, many of these operations are spatially inhomogeneous, suggesting that they may be based on a higher-level serial control [Ull84].

Another alternative is to choose a more modest common core, e.g., the image itself, with perhaps a few of its properties (such as the orientations of line fragments) made explicit. Essentially, this identifies low-level vision with some variant of early vision, perhaps augmented by high-speed grouping processes. This approach is found in many model-based recognition schemes (e.g., [Bro81, Bie85, Low85]), where recognition proceeds via the matching of image features to projections of a predefined model onto the image plane. It also is found in techniques that use image features to index directly into a large set of predefined models (e.g., [PE90]). But models are not always available, especially for unknown environments. Even when they are, occlusion often removes many of the relevant features, raising the possibility of confusion with other objects that share the same subset of visible features. Furthermore, this approach must be able to handle all possible views of all possible objects at all possible orientations in the scene. This makes the system unwieldy as the number of objects to be represented increases: memory requirements can become substantial if all possibilities are to be stored; if procedures are used to reduce the memory requirements, computation time increases. Thus, a “minimal core” based on simple image properties is often too minimal for low-level vision. A more complete intermediate description of the scene is therefore required.

2.3.2 The Role of Rapid Parallel Recovery

It would appear that low-level vision faces a dilemma of sorts, since a common core based on properties of the scene cannot be computed quickly, while simple image-based properties are insufficient for general purposes. But there is a way around this dilemma: instead of demanding that interpretations make optimal use of available information, demand only that they be “reasonably correct”. In particular, instead of demanding that interpretations be consistent over the entire image, demand only that they be consistent over spatially limited zones.

Relaxing consistency in this way allows the recovery of scene properties to have a complexity far below that of “optimal” recovery: not only is maximal use made of parallelism, but the interaction of each processor with its neighbors can be considerably simplified (cf. section 2.1.1). Since nonlocal context provides much of the information for interpretation, the outcome is usually suboptimal; in fact, interpretations may exist only over a sparse set of locations in the visual field. Thus, a rapid recovery process cannot be expected to produce a description that is complete, or even globally coherent. What *can* be expected, however, is that some of this description will be accurate enough for tasks further along the processing stream.

Such “quasi-valid” estimates could be useful in several ways. For example, they could help guide processes that cannot afford to wait for a complete analysis of the scene (e.g., active visual processes such as gaze or focus of attention [Bal91]). They could also act as precursors to serve as the initial estimates for slower processes that restore some degree of global consistency [ER92]. They might also be used as (invariant) indexes into higher-level object models, thereby increasing the efficiency of model-based recognition. In any event, this view of early vision suggests that parallel processes may play a greater role than previously suspected — in essence, the “horizontal” stages of the conventional theories may be complemented with “vertical” islands of locally-consistent interpretations.

Given the plausibility of this viewpoint, the problem now is to develop it into a rigorous theory of early visual processing. It is essential to find a way to describe a rapid recovery process precisely and to justify its operation. What is required for this is a framework that allows it to be given a computational analysis in the sense of Marr [Mar82].

2.4 The Analysis of Resource-Limited Processes

If rapid recovery is to be given a rigorous computational analysis, a general framework must exist that allows a clear formulation of the problem and sets the ground rules for its explanation. The framework proposed by Marr [Mar82] goes a long way towards this end. However, it can only be used to analyze processes for which the limited resource is the information available in the image [RP91]. A few studies (e.g., [FB82, Ros87, Tso87]) have grappled with the issue of how time and space limitations influence the structure of a visual process, but a general framework for the incorporation of resource limitations has not yet appeared. Such a framework is therefore developed here, based on a direct extension of Marr’s framework.

2.4.1 Marr's Framework

According to Marr [Mar82], the complete analysis of a visual process involves three different levels of explanation:

1. **Computational level.** This is concerned with the functional aspects of the task. It consists of two parts: (i) a description of the constraints between the input and output of a visual process, and (ii) a justification of *why* these particular constraints were chosen.
2. **Algorithmic level.** Analysis at this level describes and justifies the representations and algorithms used. It is essentially a constructive demonstration that an algorithm exists capable of generating the required mapping.
3. **Implementational level.** This level is concerned with the description and justification of the physical substrate on which the algorithms are implemented. An "implementational" explanation provides a constructive demonstration that there exists a physical system that can carry out the required computations.

One of the strengths of this framework is its recognition of a separate "computational" level of explanation focusing on both the *what* and the *why* of the input-output mapping. The *what* is concerned with the explicit *description* of the constraints on the form of the mapping. This aspect of analysis is complete when the constraints are shown to determine a unique mapping. The *why* is concerned with the justification of these constraints, showing that the resulting set of associations between input and output is suitable for the purposes at hand. To use an example taken from Marr [Mar82, pp. 22-24], the *what* of a cash register's function is explained by describing its output as the sum of its inputs. The *why* of this function is explained by the need for a pricing mechanism that has a zero value, is commutative and associative, and that allows inverse operations.

In this approach, constraints on the input-output mapping of a visual process are assumed to be machine-indifferent, originating from the laws of optics or from the structure of the objects under consideration. As such, it implicitly assumes that the mappings are shaped only by the information available in the image, and not by limits on the computational resources.¹⁸ This allows analysis to be completely general, with no dependence on the structure of the processor carrying out the computations. When the process is limited primarily by the

¹⁸ Marr [Mar82] does consider efficiency to be important, but only once the task itself has been laid out. Efficiency itself is therefore addressed at the algorithmic rather than the computational level of analysis.

available information, it can be completely explained by this kind of analysis. But when it is limited by other factors, something more is needed.

2.4.2 Extensions

A. External and Internal Constraints

If resource limitations are to be incorporated into a computational framework, several important distinctions must first be made. The first is that between *external* and *internal* constraints. External constraints are those on the “static” aspect of the mapping, i.e., those definable without regard to the way the output is generated. These are essentially the constraints that apply when the processor is viewed as a “black box”. In the case of the cash register, for example, the requirements of commutativity and associativity are external constraints, applicable only to the final form of the output function. These constraints can operate either directly via relations between input and output elements, or more indirectly via relations between the elements of the input or output domains (see, e.g., [RM89]).

When a process can be analyzed entirely in terms of external constraints, the internal details of the processor are irrelevant. But when limited computational resources enter into the picture, it becomes important to consider exactly *how* the process is carried out. This is specified by the internal constraints, which apply to the way the output is *generated* [RP91]. More precisely, these are invariants of the information flow that occurs during the course of the computation. These constraints include limits on the communication bandwidth and architectural constraints on the set of basic operations to be used. Internal constraints can therefore influence the complexity of a given operation on various kinds of processors.

B. Resources and Resource Limitations

It is important to recognize that when an information-processing task is analyzed, a subset of constraints is usually specified that is fixed and not subject to further discussion. For example, when analyzing a process to recover shape from shading, the available information is determined by the viewing conditions and sensor array specified in the problem formulation. Explanation then centers around the constraints used to recover shape from this information, but the available information itself remains as a given throughout this analysis, and does not need to be explained. As such, the available information is effectively a “boundary condition”

for the analysis, limiting the set of input-output mappings that can be considered. Such quantities are referred to here as *resources*, and the corresponding constraints as *resource limitations*.

Resources can involve either the external or the internal aspects of processor operation. External resources are quantities that can be defined independently of processor structure. These include not only the available information, but also such things as the total amount of time or energy used. The corresponding constraints are referred to as *external limitations*. These generally result from higher-level factors in the surrounding environment. As such, they can be considered to be constant over the course of processing (cf. [Sal85, ch.4]).

Internal resources can be similarly defined as those quantities relevant to the internal structure of the processor. Examples of these include communication bandwidth, the distribution of memory buffers within the architecture, and the proportion of matter taking the form of processing elements. The corresponding constraints on these quantities are referred to as *internal limitations*. Considering again the example of the cash register, the explanation of a particular design may involve an internal limitation such as the requirement that metal gears be used for all operations.

Note that a similar complementarity exists for both kinds of constraint – as a given analysis requires fewer resources in its “boundary conditions”, it applies to a wider range of processes. If an analysis requires the existence of five identifiable points in an image, it also is applicable to processes based on six identifiable points. If only four points are required in the analysis, it can be applied to an even larger set of processes. In the same way, an analysis that explains the operation of a cash register containing twenty gears also applies to a wider range of processes than one based on a limitation of forty gears.

C. Abstractness

A quantity is said to be *abstract* to the degree that its physical composition is relevant to the analysis. The most abstract quantities are purely formal ones, i.e., those that are independent of the properties of the underlying substrate. Such formal quantities include information and computational measures of time (section 2.1.1). The corresponding constraints and limitations are as abstract as the least abstract quantity involved. For example, the requirement of commutativity is a purely abstract constraint on the operation of a cash register, being completely independent of its material composition. Similarly, the requirement that a base

10 representation be used is also independent of physical structure. As these examples show, both internal and external constraints can be completely abstract.

More concrete quantities contain intrinsic constraints due to the physical properties of the underlying substrate, such as its density or thermal conductivity. These properties can affect both external and internal aspects of the processor's operation. For example, setting an upper limit on the weight of a cash register limits the total value that can be represented. This results in an "approximation" of the addition operation in which the output is given a definite upper bound. This upper bound may not necessarily be important for practical purposes if the cash register is an electronic device, but it may well have a serious effect if addition is required to be done mechanically.

D. Completeness

An analysis is said to be *complete* to the extent that the constraints determine the uniqueness of the mapping, algorithm, or implementation being analyzed. For example, requiring addition to be based on a positional numeric representation provides only a partial specification of its algorithmic structure, since — among other things — the particular base has not been specified. The choice of base has no impact on functional properties such as commutativity and associativity. It may, however, influence the efficiencies possible for various operations.

Note that the initial set of limitations assumed in the formulation of a problem already sets limits to the kinds of mappings, algorithms, or implementations that are possible. The set of constraints obtained from a computational analysis therefore serves to complete this original set of specifications.

2.4.3 A Revised Framework

The above considerations can be incorporated into a coherent system by a straightforward extension of Marr's framework. The resulting system is summarized in figure 2.5.

As in the original framework, analysis is carried out at three different levels of explanation. The most general of these is the computational level, where analysis is centered around the description and justification of the mapping between image and reconstructed scene. Analysis at this level is complete when it is shown that the mapping described by the constraints is (i) unique, and (ii) is consistent with the given limitations.

1. Computational Level

Constraints sufficient to determine **input-output mapping** that is (i) unique, and (ii) exists within given limitations

	External	Internal
Abstract	All	Possibly some
Concrete	Possibly some	Possibly some

2. Algorithmic Level

Constraints sufficient to determine **procedural decomposition** that is (i) unique, and (ii) exists within given limitations

	External	Internal
Abstract		All remaining
Concrete	Possibly some	Possibly some

3. Implementational Level

Constraints sufficient to determine **physical instantiation** that is (i) unique, and (ii) exists within given limitations

	External	Internal
Abstract		
Concrete	All remaining	All remaining

Figure 2.5: Extended computational framework.

If the analysis is to be general, these limitations must be abstract (i.e., involve no physical properties) and external (i.e., have no dependence on the internal structure of the processor). The constraints derived under these conditions (shown in the upper left quadrant of figure 2.5) are therefore independent of any assumptions about the processor itself. This essentially corresponds to an analysis carried out at the computational level in Marr's framework, except that constraints may now be justified by an appeal to abstract resources other than available information (e.g., time or space).

It may not be possible to explain a mapping in such a general way if it has been shaped by the physical properties or internal structure of the processor. Analysis must then be completed by invoking limitations that are less general. These may be less abstract or may involve the internal structure of the processor to some degree. The corresponding constraints are located in the remaining quadrants of figure 2.5. Note that if limitations pertain to only a few aspects of the process, they can give rise to only a partial set of constraints on its physical substrate or architecture. If so, this still allows the analysis to be applicable to a relatively large set of processes.

Similar considerations apply to the algorithmic level of analysis, where the goal is to decompose the given process into a system of more elementary data structures and operations. Explanation at this level describes and justifies the constraints that make this decomposition unique. If internal constraints exist at the computational level, the two levels of analysis will not be completely independent — the algorithmic analysis must not only obtain a set of abstract internal constraints, but also ensure that they are consistent with those obtained from the computational level (upper right quadrants in figure 2.5). An algorithmic analysis is complete to the extent that it specifies a decomposition that is both unique and consistent with all other constraints. It is general to the extent that nothing is assumed about the physical composition of the processor itself.

The final level of analysis is that of implementation. As in Marr's framework, the goal is to specify a set of constraints that determine a unique physical instantiation of the processor. However, there are now two sources of constraint to contend with: external constraints on the total amount of material, and internal constraints on its distribution within the processor. Note that the implementational constraints do not determine the physical implementation precisely, but only to the "granularity" of the algorithmic analysis. Once a process is understood at the three levels, analysis can be recursively applied to each of the components of this decomposition.

2.5 Rapid Line Interpretation

Computational models have been most successful when (i) the parameters of the problem (such as input, output, and resource use) can be clearly specified, (ii) the problem can be solved by a modular process, and (iii) the constraints obtained by the analysis lead to testable predictions (see, e.g., [PTK85]). It is evident from section 2.3.2 that the rapid recovery process is highly modular, requiring virtually no interaction with other aspects of visual processing. It is also evident that knowledge of the constraints on this process can lead to predictions about the kinds of line drawings that can and cannot be rapidly detected at early levels of human vision. This section shows that the problem itself is well defined, with all relevant parameters clearly specified.

2.5.1 Basic Terms

A. Time

In order to keep the analysis as general as possible, the basic unit of time is taken to be that required to combine two independent quantities, or to transmit across some unit distance. By describing time in terms of O-notation, this basic unit does not need to be specified in greater detail (see section 2.1.1).

It is also important here to distinguish between serial and parallel measures of time. Serial time refers to that required on a serial machine; essentially, this describes the total amount of “work” needed. Parallel time is the minimum time required on a given parallel architecture, and is often less than serial time.¹⁹ Unless otherwise specified, time is identified here with parallel time.

B. Rapid processing

For many visual processes, it is assumed (often implicitly) that optimal or near-optimal use is made of the information available in the image. This effectively places a fixed lower bound on the information to be used for a process, the exact bound depending on the input image. Since every problem has an intrinsic complexity, any such “information-limited” problem must have a lower bound on the time it requires (see section 2.1.1).

¹⁹The solution of some problems cannot be sped up by using a parallel architecture — see section 2.1.1.

Similar considerations hold for other resource limitations. In particular, a “time-limited” process can be defined by placing upper bounds on the available processing time, these bounds depending on the input image. No upper bound is explicitly given for the information used by such a process, but complexity considerations imply that such an bound must exist. Given the complementary nature of their upper and lower limits, it is seen that — at least in a very broad sense — information-limited and time-limited processes are duals of each other.²⁰

Intuitively, a rapid process is a time-limited process for which the upper bounds on time are relatively low. In the interests of precision, the term ‘rapid’ refers here to any process for which the complexity is a sublinear function of the number of lines in the image.²¹ This choice is motivated by two considerations. First, processes that can be carried out in polynomial time form a natural complexity class, with all polynomial processes retaining polynomial complexity even when carried out on various machines (section 2.1.1). As such, linear-time processes cannot be readily isolated. Given that processes of high-degree polynomial complexity cannot be considered as rapid, the sublinear criterion must be imposed if an awkward theoretical boundary is to be avoided.

The choice of the sublinear criterion also is motivated by practical reasons: it is generally impossible to distinguish a linear-time parallel process from a constant-time process applied sequentially to each location in the visual field [Tow72]. Consequently, only sublinear processes can be readily identified as being carried out in parallel.

2.5.2 Formulation of the Problem

In what follows, the expression ‘rapid recovery’ refers to the rapid interpretation of line drawings. The scene domain is a restriction of the blocks world (section 2.2.1) in which only three edges can be in contact about the vertex of any corner (section 1.1). The scene is assumed to be projected onto the image plane via a monocular orthographic projection. The inputs are therefore drawings composed of straight line segments with no dangling ends and which meet in junctions composed of either two or three lines. The outputs are viewer-centered dense descriptions (i.e., maps) of the structure of the corresponding polyhedra in

²⁰Note that this is completely separate from considerations of efficiency. The efficiency of an information-limited process is a measure obtained by comparing the time it requires against the absolute lower bound imposed by complexity considerations. Similarly, the efficiency of a time-limited process is measured by comparing the amount of information it extracts from the image against the maximum that could be achieved. In both cases, efficiency is described in the same terms.

²¹The term ‘real-time’ has been suggested for processes requiring at most linear time [Vol82].

the scene.²² An estimate of the relevant properties is assumed to exist at every point along these lines. As for rapid processing generally, the available time is limited to a sublinear function of the size of the problem, i.e., the number of the lines and vertices in the drawing. The problem is to recover as much of the scene structure as possible within the allocated time.

In what follows, a rather severe limitation is imposed: the recovery process must use only a constant amount of time, i.e., the amount of time must be independent of the size or content of the input. This is motivated by several considerations. First, if the output of the rapid recovery process were the basis for more complex operations at higher levels (section 2.3), control of this interface would be greatly simplified if it could be assured that recovery was always completed within a fixed amount of time.

Second, constant-time line interpretation is an extreme case of rapid recovery, and therefore an interesting problem in its own right. Among other things, any structure recovered under these conditions sets a lower bound on what can be expected of any rapid recovery process. And given that extremely low limitations are involved, the results obtained would be applicable to the widest variety of processes (section 2.4.2).

Finally, constant-time interpretation leads in a very natural way to the locally-consistent estimates assumed to be provided by rapid recovery (section 2.3). Since transmission speeds are finite, a constant-time limitation translates into a constant-distance limit on the transmission of information in the output. As such, inconsistencies resulting from violations of the underlying assumptions are not propagated throughout the image, but are restricted to relatively small regions, or "patches". This consequently avoids the destruction of interpretations in areas where these assumptions do hold.

To make the analysis relevant for the greatest range of processors, relatively severe limitations are also placed on the available processing resources (cf. section 2.4.2). Since the rapid interpretation is likely to be done in-place by a spatiotopic array of processing elements (section 2.3), the number of processors must be proportional to the number of locations upon which the line drawing falls; accordingly, $O(n)$ processors are assumed to be available for an input of size n . The simplest way to coordinate these elements is as a two-dimensional array of independent processors. But although this architecture is in some sense a minimal

²²The term 'structure' refers here to properties of the scene. These are chosen to be the (positive) convexities, slant signs and slant magnitudes of the edges, as well as the contiguity relations between edges and surfaces (section 4.1.1).

one, it requires a considerable amount of wiring for each processing element (section 2.1.1). Processors are therefore assumed to be arranged in a mesh, with each element connected only to its nearest neighbor. It also is assumed that each processor is simple enough that its operation requires only a fixed amount of space and time.²³

Although the particular space and time limitations that apply to rapid recovery are not known, most aspects of this process can be analyzed without knowing their exact values. This can be done by assuming that the time required for local processing is less than that required for transmission across some small fraction of the image. This amounts to an assumption that the complexity of rapid recovery is dominated by transmission time, a point of view largely in accord with known limits on biological and artificial processors (section 2.1.1). Among other things, this assumption removes the need to distinguish between time as defined by signal propagation and time as defined by the number of switches along the path (section 2.1.1), since these two measures are directly proportional to each other for a mesh architecture.

The interpretation process can therefore be described in terms of the percolation of information through a mesh network at some constant speed. The absolute size of the image, the size, speed and spacing of the processors, and the speed of transmission do not need to be known — all that is relevant is the ratio of transmission speed to the length of the lines in the drawing. Even this can be eliminated by a rescaling of the image (e.g., setting the average line length to some constant). Consequently, the computational analysis is largely independent of the details of any particular representation or architecture used.

²³These assumptions, of course, do not rule out the use of a more complex architecture such as a pyramid. Rather, they merely avoid assuming the extra processing power, allowing the analysis to apply to a larger set of processes.

Chapter 3

Low-Complexity Recovery

The success of a rapid recovery system rests upon its ability to recover a large amount of scene structure within a small amount of time. Since the interpretation of a line drawing is an NP-complete problem (section 2.2.1), a mapping that recovers all possible three-dimensional structure is not generally suitable for this purpose. Instead, a low-complexity “approximation” must be used that captures only part of the relevant structure.

An approximation can differ from a more complete mapping in three ways: (i) fewer degrees of freedom in the input, (ii) fewer degrees of freedom in the output, and (iii) fewer transformations of the given data. The first way (see, e.g., [Tso87]) essentially reduces the input resolution, while the second (see, e.g., [Lev86]) reduces the expressiveness of the output. But rapid recovery is assumed to have the same kind of mapping as for optimal interpretation, viz., an association of scene properties to each (high-resolution) line in the image (section 2.5.2). Approximation is therefore based on the third way — fewer transformations of the data.

Because fewer transformations are involved, scene properties cannot always be recovered successfully at each zone in the image. If constraints are chosen carefully, however, the likelihood of this recovery can remain high. Given the limitations on time and transmission distance, this likelihood is highest for those aspects that (i) are easy to compute, and (ii) require minimal “nonlocal” input, i.e., minimal input from areas outside the zone.

This chapter examines the extent to which low-complexity recovery can be carried out along concurrent streams, each concerned with a single dimension of scene analysis. Four particular dimensions are considered: the contiguity and convexity of edges and the sign and magnitude of edge slants. Complexity bounds are derived that show the extent to which each

of these properties can be computed in sublinear time and with minimal nonlocal information. It also is shown that these streams can be combined to completely recover both qualitative and quantitative structure in sublinear time for several subdomains of polyhedral objects, including convex polyhedra and polyhedra with rectangular corners.

3.1 General Issues

Several general issues are involved in the specification of a mapping for a rapid recovery process. This section discusses three of the more important ones: the degree to which processing power can be increased by concurrent processing streams, the complexity of solving the constraints within each stream, and the tradeoffs that exist when approximating a given mapping by one of lower complexity.

3.1.1 Concurrent Streams

A decomposition into separate processing streams is found in many computational models of early vision (see, e.g., [PTK85]). This decomposition often has its origins in the processing of different media (e.g., contours defined by luminance, motion, or texture [CAT90]), or in the processing of different aspects of the output (e.g., motion, color, and binocular disparity). Each of these streams essentially contains a bundle of highly-correlated information (a “dimension”) that describes some particular aspect of structure in the image or scene.

The existence of separate streams is believed to facilitate the development of perceptual processes, since natural selection can act independently on each one [Sim81, Mar82]. But there also is another reason for their existence — they maximize the sheer amount of data transformation that can be done within a given amount of time. If a set of operations are independent of each other, they can be carried out faster in parallel rather than in sequence. Also, the complexity of each operation is often lower when fewer and less complex variables are involved. If the constraints can be reformulated such that each dimension involves only a few variables, then a maximum amount of data transformation is possible.

Such a dimension is readily obtained by coalescing the original variables into a few groups, which are then treated as coarse-grained variables governed by a smaller set of “collapsed” constraints [MMH85, Mal87]. By grouping the original set of variables in several different ways, the original problem can be largely decomposed into several simpler subproblems, each of which can be solved by a concurrent processing stream. Note that the sets of *properties*

handled by these streams do not need to be independent of each other — only the *systems of constraints* need to be this way.

Decomposing a process into concurrent streams can lead to a considerable reduction of processing time, but the price of this reduction is a loss of coherence: the solutions obtained in each stream are not necessarily compatible with those obtained in the other streams. Thus, cross-dimensional constraints must be incorporated if all (or even much) of the power of the original set of constraints is to be retained. An important aspect of developing a successful approximation is therefore to maximize the use of cross-dimensional constraints without increasing the complexity of the problem.

One such strategy is based on a hierarchical decomposition of variables [MMH85]. Here, the original variables are grouped together into a few sets of coarser-grained variables that obey a simple set of collapsed constraints. Once the set of collapsed constraints is solved, the result is used as the basis for a new problem involving finer-grained variables. This can in turn be applied to yet another set of constraints on even finer-grained variables. In essence, the problem has been decomposed into a sequence of simple streams in which the outputs of the coarser-grained systems help with the solution of the finer-grained ones.

More generally, low-complexity interaction is possible if information from an unambiguous set of results in one stream can be transmitted to help constrain possible solutions in another. Since the transmission is based on unambiguous (local) results, the backward flow of information from the second stream to the first one has no further effect on the original result. This essentially corresponds to a unidirectional linkage (section 2.1.1) between streams, with linkage now generalized to apply not only to interactions across geometrical space, but across more abstract dimensions as well. If the amount of information to be transmitted is small, the cross-dimensional constraints will not add to the complexity of the problem.

3.1.2 Reduction to Canonical Forms

If an approximation is to capture much of the structure of the original mapping, it must focus on those aspects of the scene that are (i) easy to compute and (ii) need a minimal amount of information from outside the local zone. One way to help ensure that these conditions are met is to select dimensions such that their determination can be reduced to the solution of some low-complexity problem. Two problems are of particular importance in this regard: 2-Satisfiability (2-SAT) and connected components labelling (CCL).

In order to simplify the reduction to these problems, only the constraints on arrow-, Y-, and L-junctions are considered explicitly. The constraints on T-junctions are handled by a preprocessing step where virtual gaps are introduced between the stems and the crossbars of each T-junction, the two lines afterwards treated as unconnected. The crossbar of the T-junction then corresponds to an occluding edge, while the stem becomes an unconstrained line that has at least one “dangling” end. This reformulation has the advantage that the remaining linkages automatically split the n lines in the image into separate *partitions*, each of which can be treated separately.¹

A. Reduction to 2-SAT

One way to ensure that a dimension is easy to compute is to restrict the set of “intra-dimensional” constraints so that they correspond to an instance of the 2-Satisfiability (2-SAT) problem. This can be defined in terms of a set of boolean variables² $V = (v_1, v_2, \dots, v_n)$ and a set of clauses $C = (c_1, c_2, \dots, c_m)$, with each c_i containing either one or two variables. The problem is to assign truth values to the v_i such that all clauses in C have at least one ‘true’ literal (see, e.g. [GJ79]). Since the clause $c_k = \{v_i, v_j\}$ is a disjunction of variables, it has the equivalent form $c_k = \sim (\bar{v}_i \wedge \bar{v}_j)$. Consequently, any problem involving binary constraints on two-valued variables can be treated as an instance of 2-SAT [Mac91].

For the line labelling problem, the variables are the possible edge labels, with the set of Huffman-Clowes constraints (section 2.2.1) determining their allowable combinations. Since these variables are four-valued, reduction to 2-SAT can only be done by decomposing the set of variables into sets of simpler elements.

For the most part, such a reduction occurs via the direct transcription of edge labels into two-valued variables and the recasting of the remaining junction constraints into binary form (i.e., into a form involving only two variables). For example, edge convexity could be expressed in 2-SAT by taking the ‘+’ and ‘-’ labels as the complementary values to be attached to the (interior) edges, and — as far as possible — implementing the constraints on the junction interpretations via binary constraints on these variables. Because of the

¹Structures such as holes sometimes lead to separate sets of labels being used for different parts of the same object. But this occurs even in Huffman-Clowes labelling.

²More precisely, the 2-SAT problem is defined in terms of a set of literals $U = (u_1, \bar{u}_1, u_2, \bar{u}_2, \dots, u_n, \bar{u}_n)$. These literals are constrained such that only one of the pair u_i or \bar{u}_i can be used, allowing them to be treated as two-valued boolean variables, the value of variable i being ‘true’ if u_i is selected and ‘false’ if \bar{u}_i is selected.

independence of the partitions, variables in the image can take on more than two values, provided that this does not occur within any single partition.

Bringing together the above considerations yields

Theorem 3.1 *If a line drawing of n lines has one or more partitions, each such that*

1. *All variables take on 2 values, and*
2. *All constraints on more than one variable are binary,*

then the relevant labelling problem can be reduced to an instance of 2-SAT.

All 2-SAT problems of size n can be solved in $\Theta(n)$ time on a serial processor [EIS76], and in $O(\log^2 n)$ time when $O(n^3)$ processors are available [KP88].

B. Reduction to CCL

The reduction to 2-SAT makes little appeal to the geometrical organization of the constraints *per se*, using them only in regards to the ease of computation. But spatial coherence can also be used, both to obtain an approximation of lower complexity, and to help minimize the amount of nonlocal input needed for the local interpretations.

In particular, note that there sometimes exists a coordination among the sets of edge labels possible for a junction, or more generally, for some connected subset of lines in the drawing. For example, if boundary edges are ignored, all lines in a Y-junction must have the same label, either '+' or '-'. To capture this notion of coordination, define a *bijective* constraint on a set of variables u_i as one where the number of possible values for each variable is the same, and with a 1:1 linking between the allowable values (figure 3.1). For a bijective constraint, therefore, the value of one variable determines the values of the others.³ The variables are essentially "locked together", and can be treated as a single quantity, with appropriately reformulated constraints being applied to neighboring variables.

When two bijective constraints apply to a common variable, the resultant set of constraints is also bijective. This can simplify analysis considerably, allowing sets of junctions with bijective constraints on m variables to be treated as a single m -valued complex when these junctions are connected to each other by lines in the drawing (figure 3.1). An example

³More precisely, the value of any variable is related to that of any other by a bijective function.

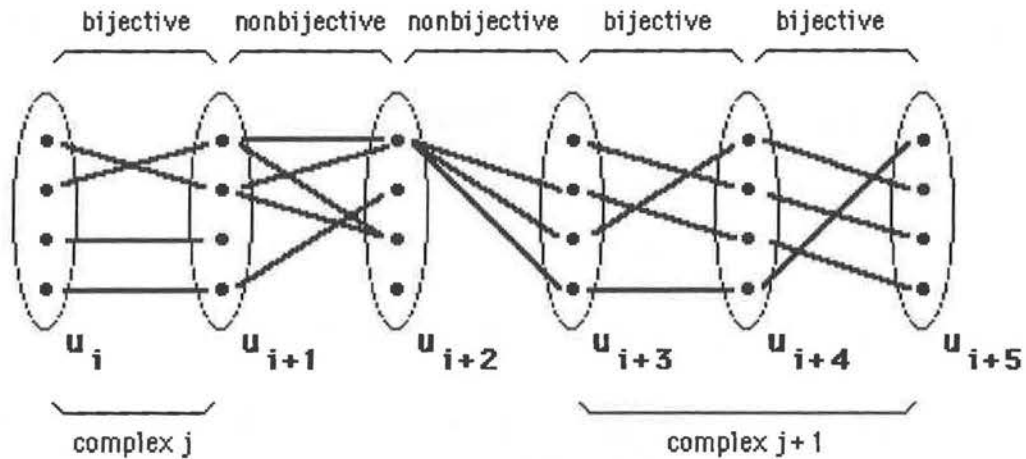


Figure 3.1: Linking of local constraints. Lines connect values compatible with each other.

of such a co-ordinated complex is the Necker cube. Here, two globally-consistent interpretations are possible, each of which has no local interpretations in common with the other. Because the junctions are linked via bijective constraints, the interpretation of any one junction immediately determines those of all the others.

For a bijective complex, globally inconsistent labellings can be removed by sending a signal from locations where legal values are missing and propagating it along the lines of the complex, the signal causing the withdrawal of the relevant value at each location along the way. This propagation can be stopped at locations where the value has already been removed, and so when all propagated signals have stopped, only the consistent labellings will remain. This process is essentially a variant of connected component labelling (CCL), with connections made on the basis of the bijective constraints found at each junction.⁴ If only one particular interpretation is required, all but one value can be deleted from one of the locations, and the interpretations associated with the deleted variables removed.

The interpretation process for a complex of bijective constraints can therefore be reduced to CCL. Since a line drawing may contain several complexes separated or surrounded by “free” variables not in a complex, this is not necessarily true of the interpretation of the drawing itself. A lower-complexity approximation will only be possible when minimal effort is used in assigning values to the free variables and co-ordinating the interpretations of the various complexes. At least two such conditions exist: when variables can have only one

⁴More abstractly, this is a unidirectional perimeter-linkage problem (section 2.1.1), since all that is required is knowing which of the m labels to attach to each of the lines crossing the boundary of the zone. The result of joining together two complexes across adjacent zones is always a single complex, since the constraints across the boundary are also bijective.

value, or when they are not subject to any constraints at all. This consequently yields

Theorem 3.2 *If a drawing of n lines has one or more partitions, each such that*

1. *All variables take on m values, and*
2. *All constraints on more than one variable are bijective*

then the problem can be reduced to an instance of CCL.

The complexity of CCL is $\Theta(n)$ time for a serial processor, and $O(\log n)$ time when n processors are available⁵ [SV82, LAN89].

Bijjective constraints can also simplify the analysis of cross-dimensional interactions. If the interpretation in one stream corresponds to a single complex that covers the entire drawing, the number of possibilities is fixed, being at most the number of values possible for any local variable. And if the interpretation in a different stream also corresponds to such a complex, it too will have a fixed number of possible interpretations. Since only a fixed number of possible combinations needs to be examined, the interaction between the two streams will increase complexity by at most a constant factor. This remains true even if the complexes do not cover the entire drawing, or if complexes of different streams are not aligned with each other — the important factor is only that at each location in the image only a fixed number of combinations is possible.

3.1.3 Approximation Strategies

It is often the case that a set of constraints must be altered if a problem is to be reduced to a low-complexity form. Although there are a large number of ways that this can be done, two general strategies — each diametrically opposed to the other — can be discerned.

The first of these is a *conservative* strategy, which increases the number of constraints until all can be re-expressed in the appropriate (e.g., binary or bijective) form. This approach effectively rejects a subset of legal labellings, avoiding those that require greater time (cf., “unsound reasoning” [Lev86]). Loosely speaking, speed is gained by increasing the number of “Type I” errors, i.e., increasing the number of realizable drawings (i.e., those that correspond

⁵The number of processors is actually linear in the number of edges *and* the number of vertices. But since all vertices in the line drawings considered here have at least two and at most three edges, only the number of edges is used here.

to a polyhedral scene) that are not detected as such. The result is a “quick and dirty” estimate as to what *can* exist in the scene.

The opposite of this is a *liberal* strategy, which removes constraints until the remainder can be put into the appropriate form. Here, low complexity becomes possible by increasing the “Type II” error rate, i.e., increasing the number of unrealizable drawings deemed to be realizable. Such a strategy can be used as the basis for a quick “preprocessor” that provides limits as to what *cannot* exist in the scene.

In general, elements of both strategies may be used to develop an approximation, the Type I and Type II error rates being traded off against each other.

3.2 Individual Dimensions

Given that line interpretation is to be carried out along separate dimensions, which dimensions should these be? Several sets of considerations must be taken into account. If the determination of a dimension is to have a low complexity, it must involve as few values as possible; indeed, if the associated problem is to be reduced to 2-SAT, the variables must have only two possible values. Similarly, if use of nonlocal information is to be kept low, constraints should be bijective. And if interactions between dimensions is to be minimized, each dimension must involve constraints that interact with the others only in a unidirectional way.

It also is assumed that the dimensions involve quantities that are viewer-centered, a condition generally assumed for all of early visual processing [Mar82]. Among other things, this ensures that the recovery process obeys the more general *viewpoint consistency constraint* [Low87], which assumes that the scene is viewed from a single direction. It also entails that three-dimensional orientation must always be defined with respect to the direction of viewing.

Each dimension must also obey a second constraint used by virtually all theories of line interpretation: *the general viewpoint constraint* (section 2.2.1). This requires any interpretation to be stable under small changes in viewing direction. One of the consequences of this constraint is that no two edges in the scene can be contained in a plane at right angles to the image plane. This allows accidental alignments to be ruled out — arrow- and Y-junctions will always correspond to coherent corners in the scene, and since corners are assumed to involve no more than three edges (section 1.1), T-junctions will always correspond to occlusions of

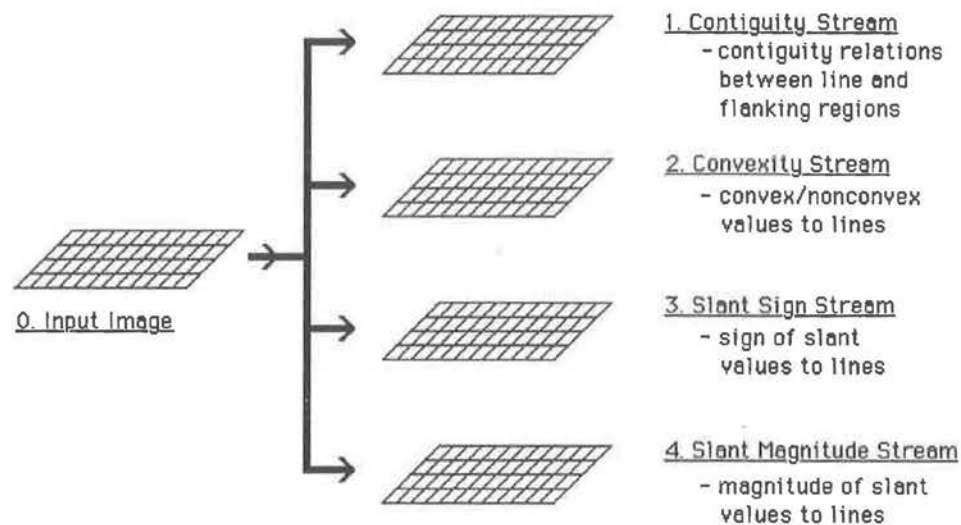


Figure 3.2: Separation into individual dimensions.

one edge by a noncontiguous surface. In general, there are usually only a few viewing directions that give rise to unstable interpretations, and so only a small penalty in interpretative power is given up in return for a large gain in performance (see, e.g., [Sug86]).

In what follows, attention is given to both the qualitative and the quantitative aspects of “optimal” line interpretation (section 2.2.1). To further increase the amount of concurrent processing (section 3.1.1), each of these is further split, yielding four largely independent dimensions: edge contiguity, edge convexity, slant sign, and slant magnitude (figure 3.2).

3.2.1 Contiguity Labelling

Much of the effectiveness of processing at early levels depends on knowing whether neighboring regions in the image correspond to contiguous or noncontiguous surfaces in the scene [Hor86, pp. 354-355]. Consequently, a reasonable candidate for an independent processing stream is one concerned with the determination of contiguity.

To be as independent as possible, the corresponding dimension must avoid quantities that describe the internal structure of the objects (e.g., convexity and concavity). Furthermore, it would also help reduce complexity if labels can have only two possible values. Thus, Huffman-Clowes (HC) labelling cannot be used. A somewhat different scheme is therefore proposed — labels indicate only whether the sides flanking a line correspond to surfaces that

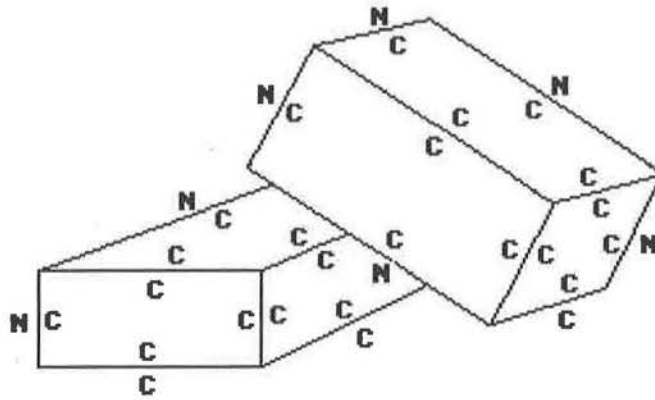


Figure 3.3: Contiguity labelling.

are contiguous (C) or noncontiguous (N) with the corresponding edge in the scene⁶ (figure 3.3). In contrast with HC labelling, each line therefore has *two* labels, one for each side.

A. Constraints on Contiguity Labelling

In order to exclude doubly discontinuous edges (i.e., wires), contiguity constraints are required for the lines. These are subject to the constraint that both sides cannot be labelled with 'N', since the polyhedral world contains no wires; however, all other combinations of 'C' and 'N' are possible (figure 3.4).

Constraints on junctions are taken from the Huffman-Clowes scheme by identifying convex and concave edges with doubly-contiguous lines, and boundary edges with singly-contiguous lines (figure 3.4). This collapses the HC constraints into the set shown in figure 3.4. It is apparent that any coherent scene will give rise to a consistent set of labels, and that this can be done by a process similar to that used for HC labelling. The result is a segmentation of the image into sets of regions corresponding to noncontiguous surfaces in the scene.

Because these constraints have been derived from the Huffman-Clowes set, any drawing which can be given a consistent HC labelling can also be given a consistent contiguity labelling. The converse situation, however, does not necessarily hold: a consistent contiguity labelling may not correspond to a consistent HC labelling (e.g., the drawing in figure 3.5). The increased susceptibility of the contiguity system to false labellings stems from the loss

⁶Mackworth [Mac74, MMH85] describes a somewhat similar scheme of "connect" and "nonconnect" edges, based on the distinction between interior and boundary edges. However, it differs from the present scheme in using one rather than two labels per line.

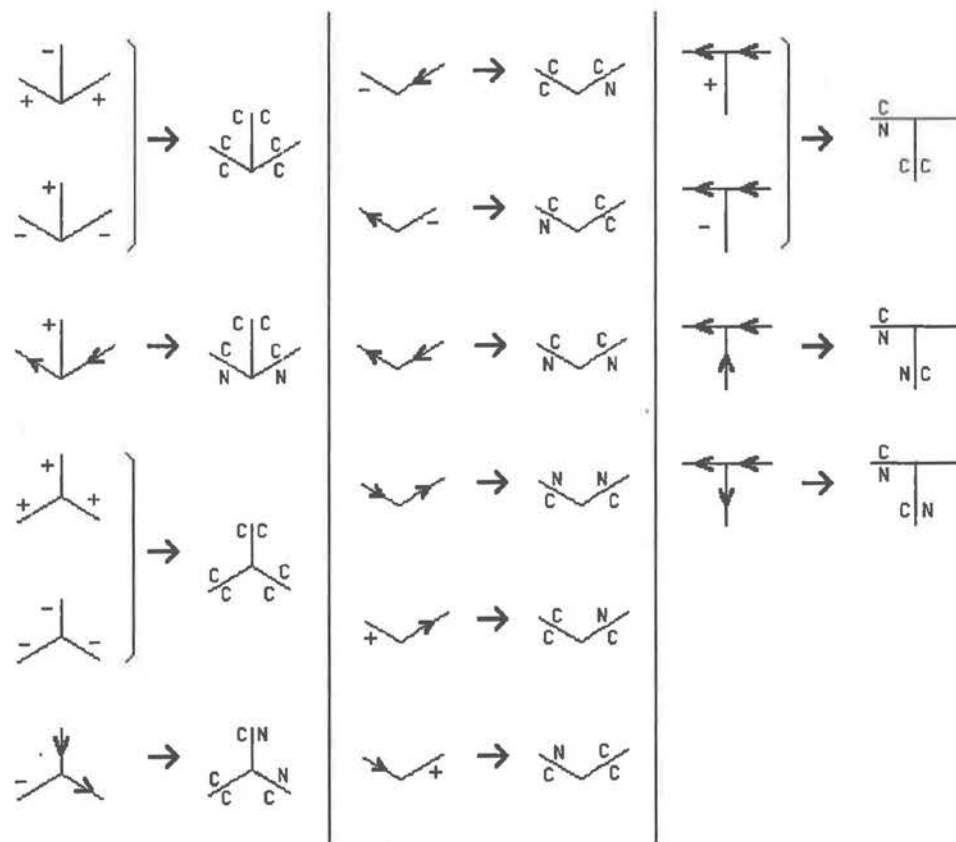


Figure 3.4: Set of contiguity constraints.

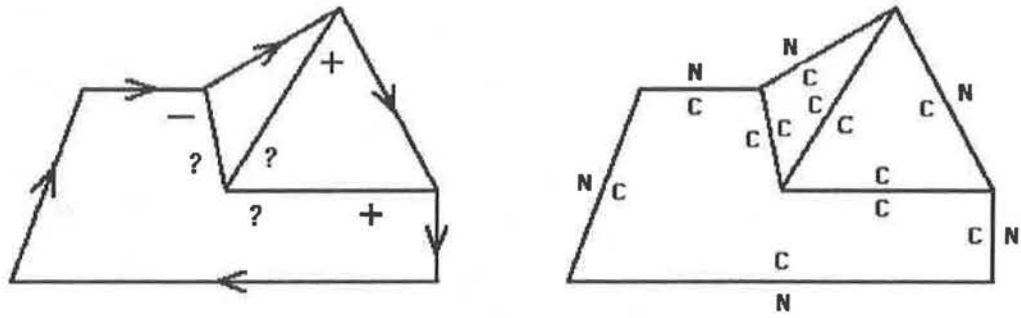


Figure 3.5: Inconsistent drawing with consistent contiguity labelling.

of the correlations between contiguity and convexity, which are not taken into account when contiguity alone is considered. Thus, a consistent HC solution that has been “weakened” by collapsing the convex and concave labels is only one of perhaps several possible solutions to the contiguity labelling problem.

B. Complexity of Contiguity Labelling

Since contiguity labelling involves only two values, its reduction to 2-SAT depends entirely on the extent to which it can be described by a set of binary constraints. As shown in figure 3.6, almost all of these constraints can be converted into binary form. Constraints on lines are quite simple, since only a prohibition against double discontinuity is needed. For Y-junctions, an additional bijective constraint is imposed: the “inside edges” of a region (i.e., lines sharing a common region) must have the same contiguity labelling. Among other things, this yields the constraint that at most one of the three faces bordering a Y-junction can be noncontiguous. For arrow-junctions, all lines except for those on the “outside” of the arrowhead must be marked as contiguous, and the outer sides of these junctions are subject to the bijective constraint that both must have the same value.

There are 16 possible combinations of N and C labels on L-junctions, of which 6 are allowed. A constraint against doubly-discontiguous lines leaves $3 \times 3 = 9$ possibilities. A constraint against diagonal N labels removes another two. This leaves only one more constraint — that against 4-way contiguity (figure 3.6) — to be enforced. This constraint, however, cannot be enforced using binary constraints. Low complexity can therefore be guaranteed only for approximations in which this constraint has somehow been replaced.

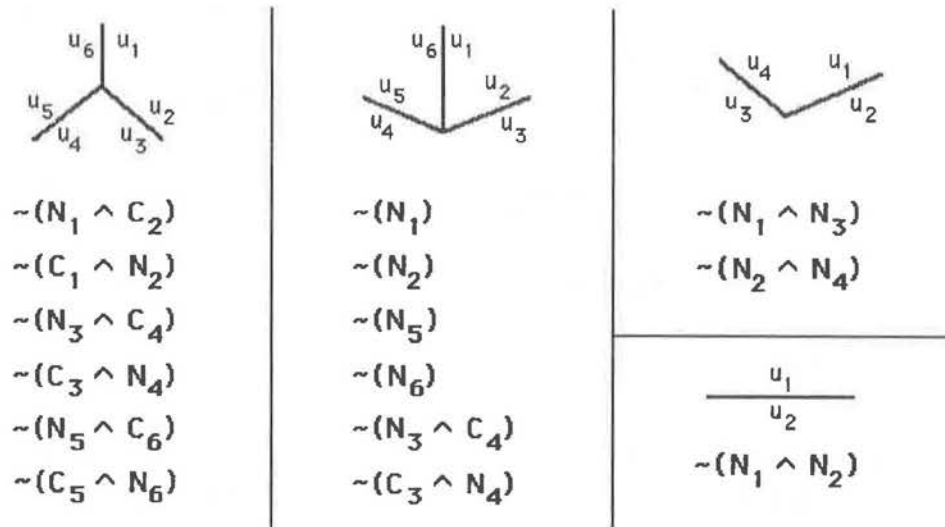


Figure 3.6: Reformulation of contiguity constraints.

Conservative approximation

One way to remove the need for an explicit constraint against 4-way contiguity is to require the inside and outside edges of an L-junction to have identical contiguity values; alternatively, one of the inside edges can be constrained to be discontinuous. Via theorem 3.1, this results in

Proposition 3.1 *When binary constraints are added that prohibit the 4-way contiguity of L-junctions, contiguity labelling can be reduced to 2-SAT.*

Liberal approximation

A low-complexity approximation can also result by omitting the need to exclude 4-way contiguity on L-junctions. This leads to

Proposition 3.2 *When the constraint against 4-way contiguity on L-junctions is omitted, contiguity labelling can be reduced to 2-SAT.*

Note that similar reductions to CCL are not possible unless extremely severe alterations are made to the constraints.

3.2.2 Convexity Labelling

Given that contiguity is concerned with inter-object relations, its natural complement is intra-object structure, viz., edge convexity. As for the case of contiguity, the standard HC labels are not suitable for present purposes, and must be replaced. A two-valued system is used here, based on that of [Mal87]: '+' for edges of positive convexity (this has the same meaning as in the HC system), and 'o' for all others. Note that the label 'o' does not necessarily correspond to negative convexity, but rather, serves as the complement required in a two-valued system. In what follows, the term 'convexity' refers to positive convexity, in the sense defined here.

A. Constraints on Convexity Labelling

The constraints on convexity labelling can be determined from those of the Huffman-Clowes set by collapsing the labels in a manner similar to that done for contiguity. The resultant set is shown in figure 3.7. Any coherent scene will give rise to a consistent set of labels, which can be found by a "standard" labelling process (section 2.2.1).

Because the convexity constraints are a subset of the HC constraints, any drawing that can be given a consistent HC labelling can also be given a consistent convexity labelling. As for the case of contiguity, however, the converse situation does not necessarily hold. An example of this is shown in figure 3.8, which can be given a consistent convexity labelling even though a consistent HC labelling is impossible.

The results of the contiguity and convexity streams can be combined if the edges marked as '+' in the convexity stream match a subset of the doubly-contiguous edges in the contiguity stream. The remaining 'o' edges can then be assigned HC labels on the basis of contiguity alone. It is evident that combining the results in this way is possible exactly when a solution of the HC constraints can be found. But such a co-ordination requires the results in both streams to be weakened versions of the HC solution and, since the streams are separated, this does not generally occur.

B. Complexity of Convexity Labelling

Since convexity labelling involves only two values, its reduction to 2-SAT depends on the extent to which the constraints can be put into bijective or binary form. As shown in figure

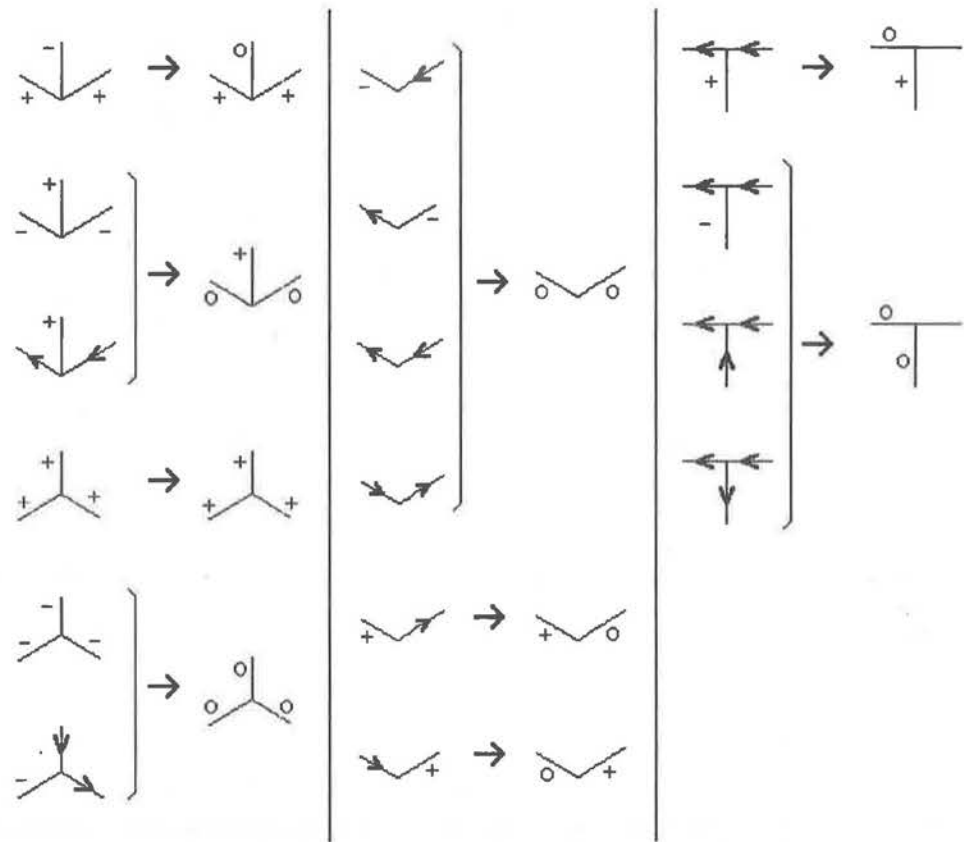


Figure 3.7: Set of convexity constraints.

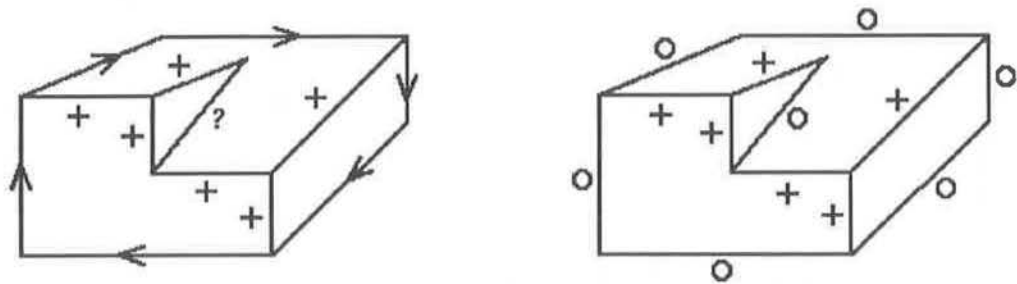


Figure 3.8: Inconsistent drawing with consistent convexity labelling.

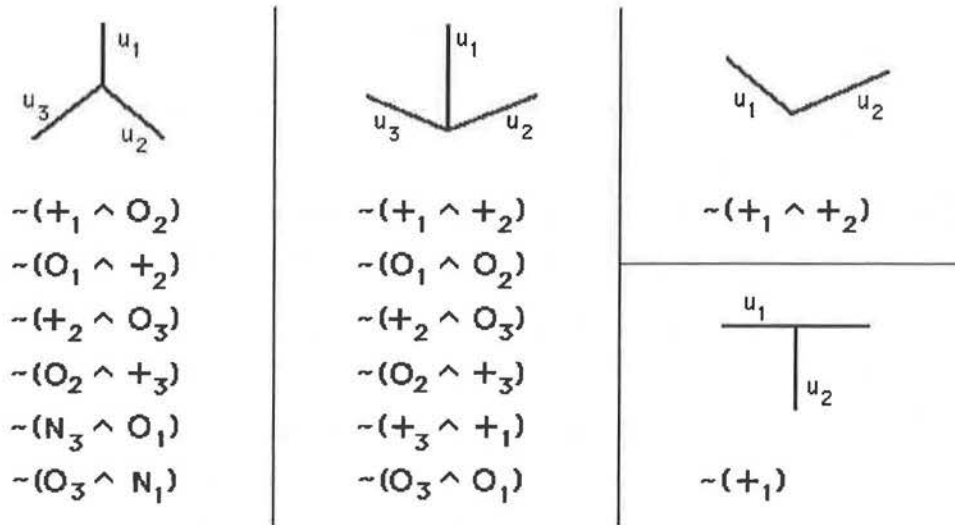


Figure 3.9: Reformulation of convexity constraints.

3.9, all of these can be put into this form. Theorem 3.1 therefore yields:

Proposition 3.3 *Convexity labelling of line drawings can be reduced to 2-SAT.*

As is evident from figure 3.9, all of these constraints are also bijective, except for the prohibition against the double convexity of L-junctions. This suggests that approximations can be derived without a great alteration of the set of constraints.

Conservative approximation

A low-complexity approximation to convexity labelling can be obtained by requiring all L-junctions to either have both sides labelled with 'o', or else to have only one side labelled with 'o'. Theorem 3.2 then leads to:

Proposition 3.4 *If all L-junctions are constrained to have both sides labelled 'o', or to have only one side labelled 'o', convexity labelling can be reduced to CCL.*

Liberal approximation

A liberal approach would be simply to allow interpretations to contain doubly-convex L-junctions. Theorem 3.2 then yields:

Proposition 3.5 *If both sides of L-junctions are allowed to be convex, convexity labelling can be reduced to CCL.*

3.2.3 Slant Sign Labelling

The quantitative aspect of line interpretation considered here is the three-dimensional orientation of the edges of each polyhedron. This property has two aspects: *tilt*, the two-dimensional orientation in the image plane, and *slant*, the deviation away from this plane. Since tilt is already available in the image, processing can focus entirely on the recovery of slant.

The determination of slant can itself be split into two components, concerned with its sign and magnitude respectively. Slant sign (see, e.g., [Kan90]) describes whether the depth of the edge increases or decreases as it travels along some direction. It remains invariant under any positive rescaling of the depth, i.e., it can represent the “z-affine” structure, which may be the most important aspect of the recovered scene [TB90]. In this sense it is similar to convexity. But slant sign is viewer-centered rather than object-centered, and so is more typical of the properties thought to be handled by early vision (section 2.1.2).

Slant sign is represented here by a double arrow⁷ (\gg), the direction of the arrow indicating the direction to follow to increase distance from the viewer (figure 3.10).⁸ The only consequence of using this representation is that under the general viewpoint constraint (section 3.2), the slant sign must remain the same under small changes of viewing position. Zero slant is therefore not allowed. This can be stated as a constraint that no edges in the scene can be at right angles to the line of sight. It is evident that any polyhedral scene obeying this general constraint will give rise to a consistent labelling of the line drawing.

A. Constraints on Slant Sign Labelling

Although many approaches (e.g., [Sug86]) require the qualitative aspects to be solved *before* the quantitative aspects, the demands of rapid processing (section 3.1.1) require that the two types of aspects be determined largely concurrently. But if this is to be done, some

⁷It may be useful to view the arrowheads as parallel lines receding into the distance.

⁸In contrast to the other quantities, slant sign can only be defined with respect to a particular direction of travel. If slant sign is to be treated as a pure scalar, a canonical direction must therefore be defined. A natural choice for a coordinate system is one based on the lines surrounding each vertex, the reference direction being that in which the vertex is approached. Represented in this way, slant sign is subject only to an additional constraint that the labelling of lines be split, with opposite ends of the lines having opposite values. A directional component also exists in the labelling of lines by arrows in the HC system, and the splitting required to put it into scalar form has become the basis of the contiguity system developed here. But because constraints on the slant system are binary and bijective, using a “split” representation will affect neither the power nor the complexity of slant sign labelling. In the interests of clarity, the “directional” form is used.

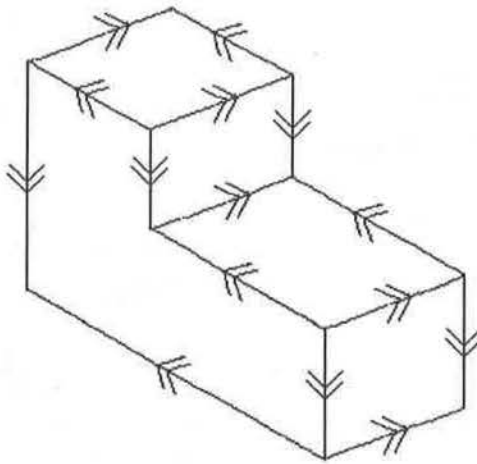


Figure 3.10: Slant sign labelling.

additional *a priori* assumptions are needed about the structure of the polyhedra in the scene — otherwise, any combination of slant signs can be attached to the lines about a junction.⁹ Such structural assumptions do indeed seem to be used by the human visual system to determine three-dimensional structure (section 2.2.2).

Convex polyhedra

A very general structural assumption is that the polyhedra are convex. This prohibits Y-junctions from having all lines slanted towards the viewer,¹⁰ since this would correspond to a dent in the surface. Similarly, an arrow-junction could not have its stem slanted away from the viewer while its two outer edges had to opposite slant. Constraints also come into play via the planarity of the faces: If the face is convex, two “chains” of arrow labels exist, which diverge from the junction at greatest distance and converge on the junction nearest the viewer.

Directangular corners

A more specific assumption is that polyhedra have *directangular* corners, i.e., corners for which two edges are at right angles to a third about which they can “swivel”.¹¹ Constraints can be based on the observation that two edges meeting at a right angle in the scene will

⁹For example, a junction can always be interpreted as a very shallow corner, and this can be tilted or flipped to achieve any combination of signs.

¹⁰More precisely, the distance to the viewer cannot be decreased as the distance from the vertex is increased.

¹¹For example, a book partway open has directangular angles at points where the spine meets the top and bottom of the covers.

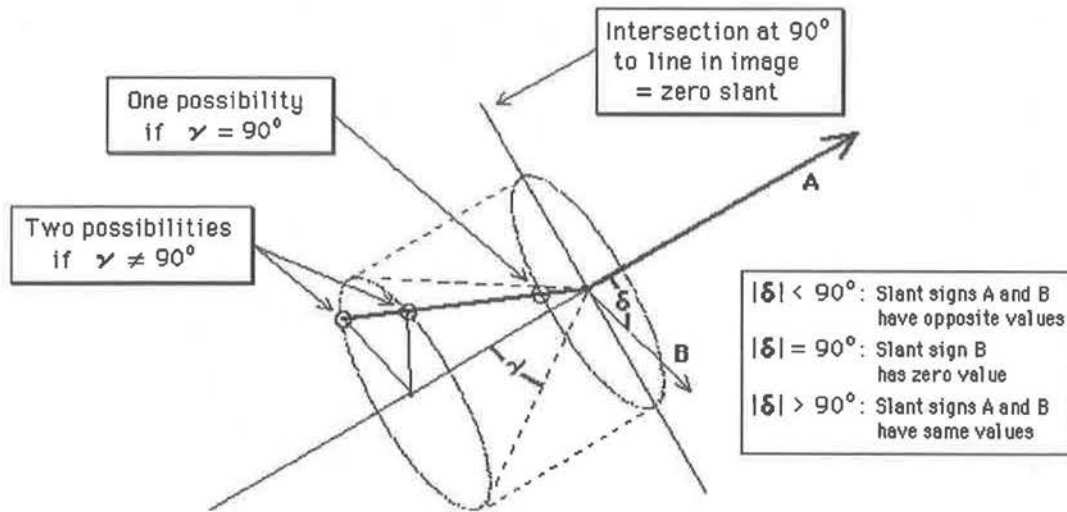


Figure 3.11: Constraints on isolated L-junctions.

always give rise to lines of opposite sign when the angle in the image plane is less than 90° , and to lines of the same sign when the image angle is greater than 90° (figure 3.11).

Given the line corresponding to the swivel edge, then, the slant signs of the other two lines can be immediately determined (figure 3.12). It follows that the constraints on the slant signs of arrow- and Y-junctions are bijective (section 3.1.2), the exact constraints depending on whether the angles between the line pairs are greater or less than 90° . However, constraints on L-junctions cannot be determined unless the orientation of the swivel axis in the image can be identified, since otherwise the angle in the image may not correspond to a 90° angle in the scene. Note that although the orientation must be given, it does not matter on which side of the junction the hidden swivel lies — a change of 180° will result in the same set of bijective constraints.

Rectangular corners

A powerful constraint apparently used by the human visual system is that of rectangularity, the assumption that all edges in each corner are at right angles to each other (section 2.2.2). As in the more general case of directangular corners, knowing the label attached to one of the lines on an arrow- or Y-junction immediately determines those of the others. But now it is not necessary to know in advance which of the lines corresponds to the swivel edge, since all edges are equivalent. The constraints themselves take on a simple form — for arrow-junctions, the slant signs of the wings must be opposite that of the stem, whereas all

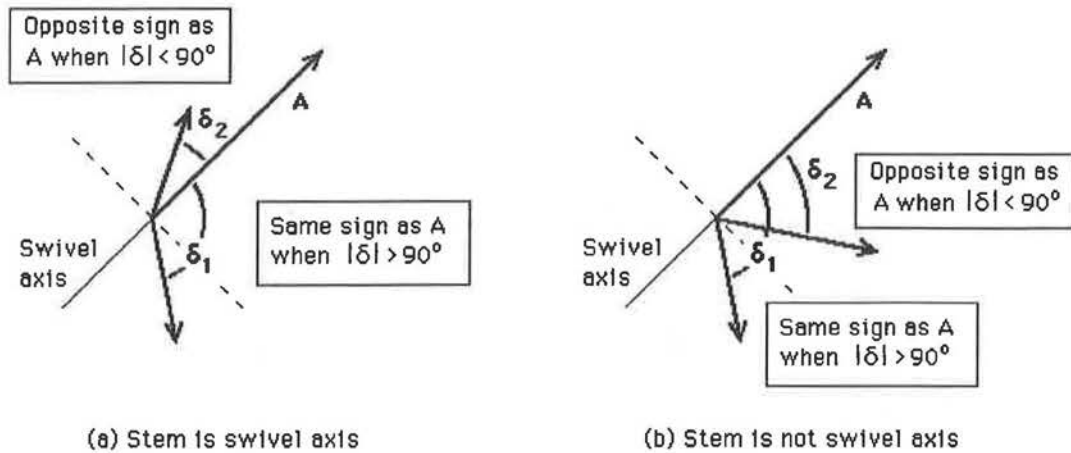


Figure 3.12: Slant sign constraints for arrow- and Y-junctions.

lines in Y-junctions must be given the same slant signs [Kan90]. Requiring a consistent set of slant signs for these junctions leads to Perkins' laws (section 2.2.1): for Y-junctions, all angles must be greater than 90° , while for arrow-junctions, the largest angle must be less than 270° , and the second-greatest less than 90° .

The set of constraints on slant sign labels for rectangular corners are shown in figure 3.13. Note that the slant sign labels on arrow- and Y-junctions become closely matched to the convexity labellings: for Y-junctions, edges are convex exactly when they are slanted away from the viewer, and are nonconvex when they are slanted towards the viewer. Similarly, an arrow-junction will have its stem slanting towards the viewer when it is nonconvex, and away when convex, the other lines taking on complementary values.

The homogeneity of angles also means that there is no ambiguity about the angle between the edges of the L-junctions. And since this angle is 90° , the slant sign of one line automatically determines that of the other (figure 3.11). Thus, L-junctions can be described entirely in terms of bijective constraints, without any need for *a priori* knowledge about the direction of the hidden swivel axis.

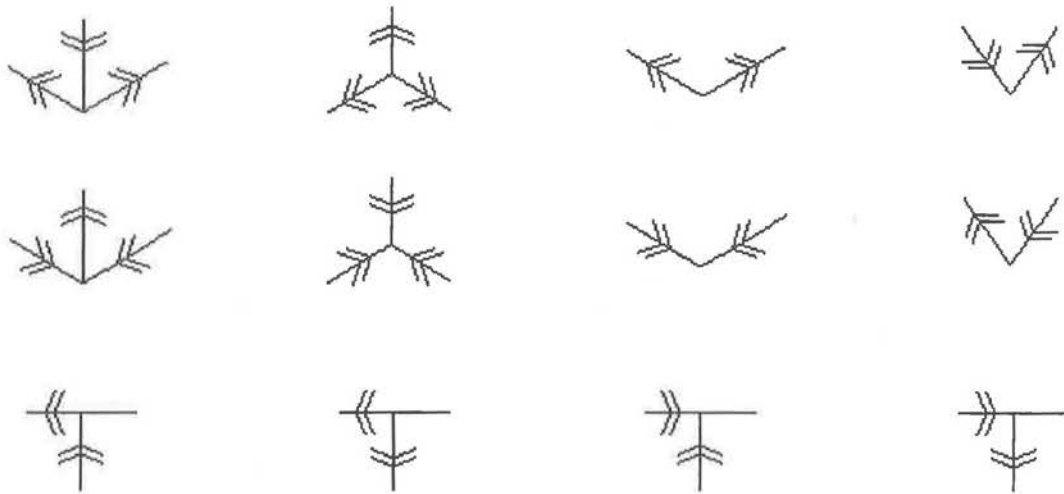


Figure 3.13: Slant sign labellings for rectangular corners.

B. Complexity of Slant Sign Labelling

Directangular corners

When corners are directangular, they give rise to bijective constraints on arrow- and Y-junctions. And when the directions of the hidden swivel axes are known, L-junctions have a similar set of constraints. Thus, from theorem 3.2,

Proposition 3.6 *When all corners are directangular and the directions of the swivel axis at all junctions are known, slant sign labelling can be reduced to CCL.*

Rectangular Corners

When corners are rectangular, a special swivel axis need not be singled out. And since L-junctions always have bijective constraints under this condition, this yields

Proposition 3.7 *When all corners are rectangular, slant sign labelling can be reduced to CCL.*

Note that the differences between directangular and rectangular corners do not lead to a significant difference in the complexity of slant sign labelling. Rather, the main differences are in the amount of *a priori* information needed from nonlocal sources.

3.2.4 Slant Magnitude Labelling

Slant magnitude is an absolute value which represents the “steepness” of an edge with respect to the image plane. This quantity is completely independent of slant sign, being invariant under inversion about the image plane, but sensitive to the rescaling of depth. It is also a quantity that takes on a continuous value. Among other things, this latter property means that the particular representation used (e.g., angle, gradient) is not important from a computational point of view, since these quantities can be transformed into each other via information-preserving operations.

A. Constraints on Slant Magnitude Labelling

As for slant sign, assumptions must be made about the structure of the polyhedra in the scene if this dimension is to be determined independently of the others.

Known corners

If the three-dimensional structure of a corner is known and its edges have been identified with the corresponding lines in the image, a system of equations can be set up between the slants of these edges and the angles between the lines of the junction [Kan90, p.288]

$$\begin{aligned}\sin \phi_1 \sin \phi_2 \cos(\theta_1 - \theta_2) + \cos \phi_1 \cos \phi_2 &= \cos \gamma_{12}, \\ \sin \phi_2 \sin \phi_3 \cos(\theta_2 - \theta_3) + \cos \phi_2 \cos \phi_3 &= \cos \gamma_{23}, \\ \sin \phi_3 \sin \phi_1 \cos(\theta_3 - \theta_1) + \cos \phi_3 \cos \phi_1 &= \cos \gamma_{31},\end{aligned}\tag{3.1}$$

where ϕ_i is the slant of edge i (with zero being along the line of sight towards the viewer), θ_i the angle of edge i in the plane, and γ_{ij} the angle between edge i and j . A solution can be found for any value of angles chosen. However, this solution requires an iterative scheme (e.g., Newton-Raphson) unless additional constraints are introduced [Kan90]. In order to keep the measure of slant symmetrical about the image plane, the angle $\alpha = \pi/2 - \phi$ is used for the slant magnitude itself, with α always in the interval $(-\pi/2, \pi/2)$.

Slant magnitudes cannot be determined for L-junctions in isolation, even when the angle between their edges is known. But equation 3.1 shows that if the slant magnitude of one of the edges is known, that of the other can be determined. And because this is an equation linear in $\sin \phi$ and $\cos \phi$, ϕ (and therefore α) can be solved for analytically. If γ is not 90° , two values are possible, corresponding to edges of greater or lesser slant (figure 3.11). These

can take on different slant signs, depending on the particular value of γ ; if this occurs, slant magnitudes must become signed in order to maintain the correct binding between the signs and the magnitudes assigned to the edge. Otherwise, the slant sign of the known edge need not be given, since the solutions are symmetrical about the image plane.

Directangular corners

If the corner is known to be directangular and if the line corresponding to the swivel edge can be identified in the image, the set of equations 3.1 takes on the form [Kan90, p.289]:

$$\begin{aligned}
 A &= \frac{\cos^2(\theta_1 - \theta_3)}{\cos^2(\theta_2 - \theta_3)} [\cos^2(\theta_1 - \theta_2) - \cos^2\beta] \\
 B &= 2 \frac{\cos(\theta_1 - \theta_2) \cos(\theta_1 - \theta_3)}{\cos(\theta_2 - \theta_3)} - \left(1 + \frac{\cos^2(\theta_1 - \theta_3)}{\cos^2(\theta_2 - \theta_3)} \right) \cos^2\beta \\
 C &= \sin^2\beta \\
 X &= \frac{-B + \sqrt{B^2 - 4AC}}{2A} \\
 \alpha_1 &= \pi/2 - \tan^{-1} \sqrt{X} \tag{3.2}
 \end{aligned}$$

$$\alpha_2 = \pi/2 - \tan^{-1} \left(\frac{\cos(\theta_3 - \theta_1)}{\cos(\theta_3 - \theta_2)} \cot \alpha_1 \right), \tag{3.3}$$

$$\alpha_3 = \pi/2 - \tan^{-1} \frac{-1}{\cos(\theta_3 - \theta_1) \cot \alpha_1}, \tag{3.4}$$

where β denotes the angle about the swivel axis, taken here to be edge 3. These values are coordinated sets, and so allow magnitude and sign to be completely separated. Since the two solutions of these equations are reflections of each other about the image plane [Kan90], arrow- and Y-junctions have unique magnitude estimates for each edge.

Although slant magnitudes cannot be determined for L-junctions, one constraint still applies — if the angle between corresponding edges is 90° , the magnitude of one edge uniquely determines that of the other. If the angle is not 90° , two values are possible (figure 3.11). Thus, if the swivel angle cannot be identified, three values are possible for the slant of the second edge.

Rectangular corners

For a rectangular corner, all edges are orthogonal to each other, and the relation between slant and junction angle can be expressed in the much simpler form [Per68, Kan90]

$$\begin{aligned}\alpha_1 &= \pi/2 - \tan^{-1} \sqrt{\frac{\cos(\theta_2 - \theta_3)}{\cos(\theta_1 - \theta_2) \cos(\theta_3 - \theta_1)}}, \\ \alpha_2 &= \pi/2 - \tan^{-1} \sqrt{\frac{\cos(\theta_3 - \theta_1)}{\cos(\theta_2 - \theta_3) \cos(\theta_1 - \theta_2)}}, \\ \alpha_3 &= \pi/2 - \tan^{-1} \sqrt{\frac{\cos(\theta_1 - \theta_2)}{\cos(\theta_3 - \theta_1) \cos(\theta_2 - \theta_3)}}.\end{aligned}\tag{3.5}$$

Note that the equality of all angles between edges also eliminates the need to know which line is the projection of the swivel axis.

The rectangularity of the corner also means that there is no ambiguity in identifying the angle γ between the edges of any corner corresponding to an L-junction. And since γ is 90° , the slant magnitude of one edge is uniquely determined by the magnitude of the other. For rectangular corners, therefore, L-junctions are completely described by bijective constraints.

B. Complexity of Slant Magnitude Labelling

Directangular Corners

When corners are directangular, there are three possible sets of magnitudes for a junction, corresponding to the three possible choices of swivel axis. If the direction of the swivel axis and the swivel angle β are known, unique magnitude estimates can be assigned to edges contacting arrow- and Y-junctions (equations 3.3 – 3.4). Furthermore, this condition also leads to binary constraints on the magnitudes possible for L-junctions. But a chain of such L-junctions could cause the number of possible values to increase exponentially with its length, these values being impossible to resolve except by sequentially proceeding along the chain. In the worst case, therefore, the determination of slant magnitude for directangular corners could require at least linear time, even on a parallel architecture.

Rectangular Corners

If all corners are rectangular, the magnitudes for the edges of arrow- and Y-junctions can be obtained directly from equation 3.5. Values for L-junctions can be determined from the fact that slant magnitude remains invariant under a reflection of one edge by 180° ;

consequently, only the angle of the hidden edge is needed, and not its direction in respect to the vertex. By determining and rebroadcasting the values of all orientations in the partition to all junctions, the direction of the hidden edge can be made available to all L-junctions.¹² Once the local estimates of slant magnitude have been obtained, it only remains to check their consistency. As discussed in section 3.1.2, such a consistency check can be carried out via CCL. Since all other operations can be done in constant time, this yields

Proposition 3.8 *When all corners are rectangular, the determination of slant magnitude has a complexity no greater than that of CCL.*

Note that although the computation of the magnitude as given by equation 3.5 can be done in constant time, it does involve several trigonometric functions. But this calculation can be done quite simply if the *slope* of the slant rather than its *angle* is the relevant quantity. In particular if the square of the slope (essentially, a “slope energy”) is used, this removes the need for both an inverse tangent and a square root function. The only remaining quantities then become cosine functions of the angles between junction lines, which can be determined quite simply via the dot product (cf. section 5.2.2). Since slope energy and slope angle are related by a monotonic function, the particular quantity chosen is of no great importance for most purposes. In the interests of maintaining a parallel between two-dimensional and three-dimensional orientations, slope is represented here by its angle.

3.3 Integration of Dimensions

As shown in section 3.2.2, completely separated dimensions are often unable to capture large parts of the mapping structure contained in the original set of constraints. For example, a drawing may have several different contiguity and convexity labellings, and *if* these are chosen such that the edges with positive convexity correspond to lines that are doubly contiguous, the two can be combined into a complete HC labelling. The separation of streams, however, means that it will generally be impossible to pick out the appropriate contiguity and convexity interpretations from among the alternatives. Instead of yielding a completely coherent interpretation, the process will be more likely to yield two partial interpretations that are incompatible with each other.

¹²If only two directions exist in the drawing, any magnitudes compatible with equation 3.5 can be assigned to the edges.

This loss of interpretative power, however, can be lessened by a controlled amount of interaction between streams. As discussed in section 3.1.1, this can be done without raising the complexity of the process provided that it is based on the unidirectional transmission of unambiguous results. In order to quickly achieve unambiguous results, a conservative strategy must be employed, based on additional structural constraints which rule out many legal interpretations (section 3.1.3). It is shown here that such a strategy can succeed for several subdomains of polyhedral objects.

3.3.1 Convex Objects

A particularly simple domain in which to begin is that of convex objects. These are polyhedral objects in which all edges of the object are convex; consequently, "material" always exists along the shortest path connecting any two points on two contacting edges (i.e., two edges that meet at a corner).

A. Constraints on Labelling of Convex Objects

By definition, the interior edges of convex objects are convex. As such, all arrow- and Y-junctions have a unique interpretation in both the contiguity and the convexity streams. Convexity also forces the inner edges of any L-junction to be contiguous; this in turn forces a unique convexity labelling of all L-junctions, i.e., all edges nonconvex. The resulting set of constraints, shown in figure 3.14, leads to a unique set of convexity labels. The only indeterminate quantities are the contiguity labels on the outer edges of arrow-junctions and L-junctions. Constraints on the outer edges of arrow-junctions are binary and bijective, requiring both edges to have the same value, whereas those on L-junctions are simple binary constraints that prohibit more than one side from being contiguous.

B. Complexity of Labelling Convex Objects

All convexity labellings are unique, and so the contiguity labels in any consistent interpretation must necessarily be compatible with the convexity labels. Consequently, the determination of a complete qualitative interpretation reduces to the determination of a consistent set of contiguity labels.

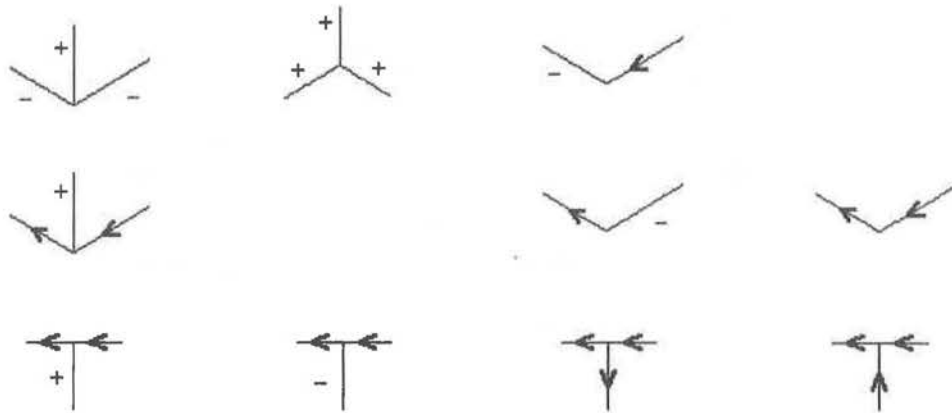


Figure 3.14: Huffman-Clowes labellings for convex objects.

i) Reduction to 2-SAT

Since all relevant constraints are binary, and since only two labels apply, proposition 3.1 leads directly to

Theorem 3.3 *The line labelling of convex objects has a complexity no greater than that of 2-SAT.*

Note that if the HC labels are required, they can be recovered simply by assigning ‘-’ to any doubly-contiguous non-convex lines, and assigning boundary lines to singly-contiguous lines. Similarly, any consistent interpretation based on separate convexity and contiguity labels can be put into HC form.

ii) Reduction to CCL

For convex objects, the contiguity of the outer edges of the L-junctions results only from the contact of adjacent blocks, and not on any intrinsic structural property. It is therefore evident that a legal labelling for a drawing exists if and only if it can be assigned an interpretation in which all blocks have been moved to a position in which they are “free floating”. This latter condition can be obtained by setting all outer edges of L-junctions to be discontinuous; the result is a set of unique contiguity labels on all outer edges.

Since all constraints are unique, no additional work is required to coordinate the results in the contiguity and the convexity streams. The complexity of the interpretation process is therefore exactly that needed to check for the presence of inconsistencies in each (section

3.1.2). This depends upon the conditions assumed for the image and scene domains. If only one set of connected lines exists in the image, the preprocessing to remove T-junctions can be omitted, and the need to maintain partitions eliminated. Under these conditions, the complexity of the interpretation process is exactly that of detection.

But the blocks world generally allows several blocks to exist simultaneously in the scene, making it necessary to distinguish between various groups of lines in the image. Since the basic HC labellings are unique, so are those of the contiguity and convexity streams. And since both these streams involve the same partitions, the integration of values is straightforward, leading to

Theorem 3.4 *If objects are assumed to not contact each other, the line labelling of convex objects has a complexity no greater than that of CCL.*

In essence, then, a principle of “minimal exterior contiguity” has been invoked to obtain a problem of lower complexity by reducing the set of preferred solutions. As opposed to a purely conservative strategy (section 3.1.3), however, this strategy does not affect the labelling problem in its narrowest sense, viz., a determination if *at least one* solution exists.

Note also that if an interpretation of a partition exists, it is necessarily the only “free floating” interpretation possible. Thus, a complete determination of scene structure (i.e., solving the realizability problem) can be reduced to finding a solution to a system of linear equations and inequalities [Sug86]. This can be solved via linear programming, which can be carried out in polynomial time [Kha79]. Linear programming, however, is a P-complete problem [Joh90, p.80], and as such is unlikely to be solvable by a sub-linear algorithm even when parallel processing is available (section 2.1.1).

3.3.2 Compound Convex Objects

Consider now a slightly less restricted domain in which it is still assumed that material always exists along the shortest path connecting any points along two contacting edges (as for convex objects), but for which the edges themselves are no longer required to be convex. These objects are referred to here as *compound convex objects*, since they can be readily realized by the attachment of convex objects to each other, this attachment being subject to the general constraint that only three edges can make contact at any vertex. Examples of such objects are shown in figure 3.15.



Figure 3.15: Examples of compound convex objects.

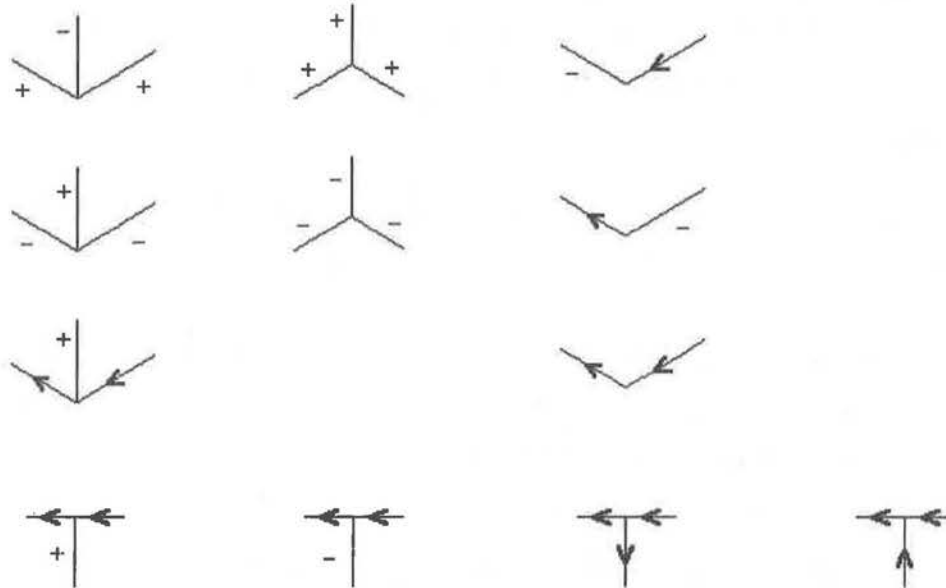


Figure 3.16: Huffman-Clowes labellings for compound convex objects.

A. Constraints on Labelling of Compound Convex Objects

Compound convex objects give rise to almost the same set of arrow-, Y-, and T-junction labellings as found in the “standard” HC set. But because of the shortest-path requirement, there must always be a common surface on the side formed by the interior angle of any L-junction, and on the surfaces between edges of an arrow-junction. The interpretation process can therefore be based on the set of junction labellings shown in figure 3.16. (The conversion into separate contiguity and convexity constraints is straightforward.) Note that only four constraints have been removed from the original Huffman-Clowes set.¹³

¹³The interpretation of an arrow-junction with a concave stem should not be allowed if consideration is focused on compound convex objects *per se*, since it can reduce the ability of the system to detect line drawings not obeying the constraints assumed. However, this interpretation can easily be removed, with all arguments going through unaffected. It is left in to show that the set of constraints in figure 3.16 potentially applies to a slightly larger domain of polyhedral objects.

B. Complexity of Labelling Compound Convex Objects

To establish bounds on the complexity of line labelling, consider first the convexity system. From figure 3.16, it is seen that no L-junctions can have a convex edge; consequently, all must have a 'o' label attached to both edges. Via proposition 3.4, it follows that convexity labelling for this domain can be reduced to CCL, the computation proceeding independently for each partition.

Compatibility between the convexity and contiguity streams can be guaranteed by transmitting the identities of any edge marked as '+' and constraining the relevant edges to be doubly contiguous. Unique contiguity values can also be assigned to the inside edges of Y-junctions, and to the crossbars of T-junctions. Since the partitions are the same for both streams, contiguity labelling needs to be done at most two times for each partition — once for each of the two possible convexity interpretations.

i) Reduction to 2-SAT

All contiguity constraints on lines and L-junctions in figure 3.16 can be put into the binary form described in section 3.2.1. Application of proposition 3.1 then yields

Theorem 3.5 *For compound convex objects, line labelling has a complexity no greater than the maximum of that of 2-SAT and CCL.*

ii) Reduction to CCL

Since the labelling of arrow-junctions, Y-junctions, and T-junctions can all be based on bijective constraints, the possibility is raised that the line labelling of compound convex objects can be reduced to CCL. This can be done by showing that the bijective constraints on L-junctions and lines are unnecessary.

Notice that each complex of bijective constraints beginning on the outside of an L-junction can be considered a "chain" that travels along the sides of arrow-junctions, terminating either when it contacts an edge with a unique value (e.g., the stem of an arrow-junction), another outer L-junction edge, or a dangling edge (figure 3.17). These chains can be readily determined via CCL. Because the junction constraints preserve contiguity, all values along a chain must have the same value. Thus, if a chain terminates at a junction which forces it to have a unique value, or contains an edge which is similarly constrained, all of its elements

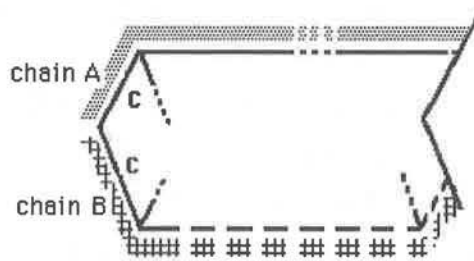


Figure 3.17: Free chain complexes.

must be set to that value, an operation which can be carried out by CCL. Otherwise, the chain is free to take on either contiguity value. As long as it ensures that the basic contiguity constraints at its ends are obeyed, the chain can be considered to be essentially decoupled from the rest of the interpretation, its values then unaffected by subsequent assignments in the rest of the drawing.

If a free chain has at least one end in contact with an L-junction, interpret its constituent variables as discontinuous. This assignment is always compatible with the constraints of figure 3.16. An interpretation constrained in these ways is therefore possible if and only if it is possible to interpret the drawing as a set of compound convex objects. The use of this restriction is essentially a generalized application of the principle of “minimal exterior contiguity” used in the analysis of convex objects. Invoking this principle essentially causes these objects to be dismantled into separate convex components whenever possible.

Somewhat similar considerations apply to a chain that has dangling edges at both its ends, except that here the chain will be interpreted as contiguous. In a direct parallel with the previous principle, this can be seen as a principle of “maximum interior contiguity”. Note that the two contiguity principles have been invoked to obtain a problem of lower complexity by reducing the set of preferred solutions. As opposed to a purely conservative strategy (section 3.1.3), however, this strategy does not affect the labelling problem in its narrowest sense, since a restricted solution will be found if *at least one* more “general” solution exists.

Having dealt with L-junctions, it must now be shown that an explicit binary constraint is not needed against doubly-discontiguous lines. If no junctions are present, a line can immediately be given any legal labelling. Otherwise, as figure 3.16 shows, the prohibition against double discontinuity is automatically imposed for all junctions. This proves

Theorem 3.6 *For compound convex objects that are assumed to not contact each other, line labelling has a complexity no greater than that of CCL.*



Figure 3.18: Examples of rectangular objects.

3.3.3 Rectangular Objects

Low-complexity interpretation is also possible for *rectangular* objects, i.e., polyhedral objects for which all corners have edges that meet at right angles. These constitute a large domain of objects, examples of which are shown in figure 3.18.

A. Constraints on Labelling of Rectangular Objects

Rectangular objects impose no additional explicit constraints on the HC labellings of arrow-, Y-, and T-junctions. Constraints only apply to L-junctions, the particular choice of constraints depending on the angle in the image.¹⁴ There are two cases to consider here. The first is when this angle is acute (i.e., less than 90°). Because the angles between the corresponding edges in the scene are 90° , the lines of an acute L-junction must have opposite slant signs (section 3.2.3). And since the hidden edge of a rectangular corner is always slanted away from the viewer [Kan90], the consistency of the slant signs leads to a bijective constraint on the convexity labelling (figure 3.19(a)).

A parallel situation exists for obtuse L-junctions (i.e., those for which the angle is greater than 90°). Rectangularity now forces both sides to take on the same slant signs. When both edges are slanted away from the viewer, they must be interpreted as a pair of singly-contiguous lines; if they are slanted towards the viewer, three interpretations are possible (figure 3.19(b)). The resultant set of junctions labellings, shown in figure 3.20, is much the same as that of Huffman-Clowes, except that the constraints shown in figure 3.19 have been added.

A more quantitative constraint that can also be used is that of *planarity*: if the planarity of the faces is to be maintained, any chain of three connected lines having three different

¹⁴To avoid possible confusion, this angle is taken to be the smaller of the two possibilities.

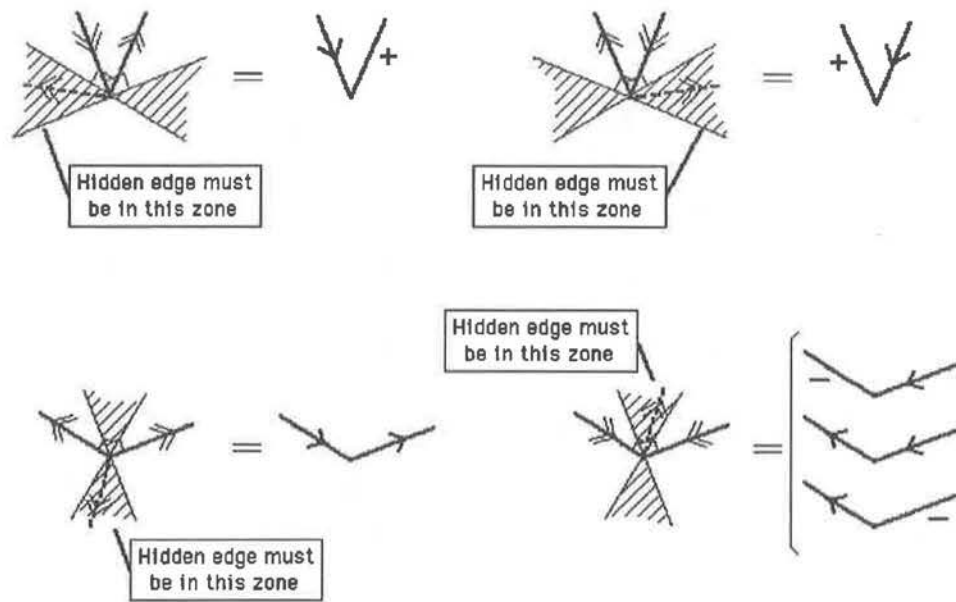


Figure 3.19: Constraints on L-junctions.

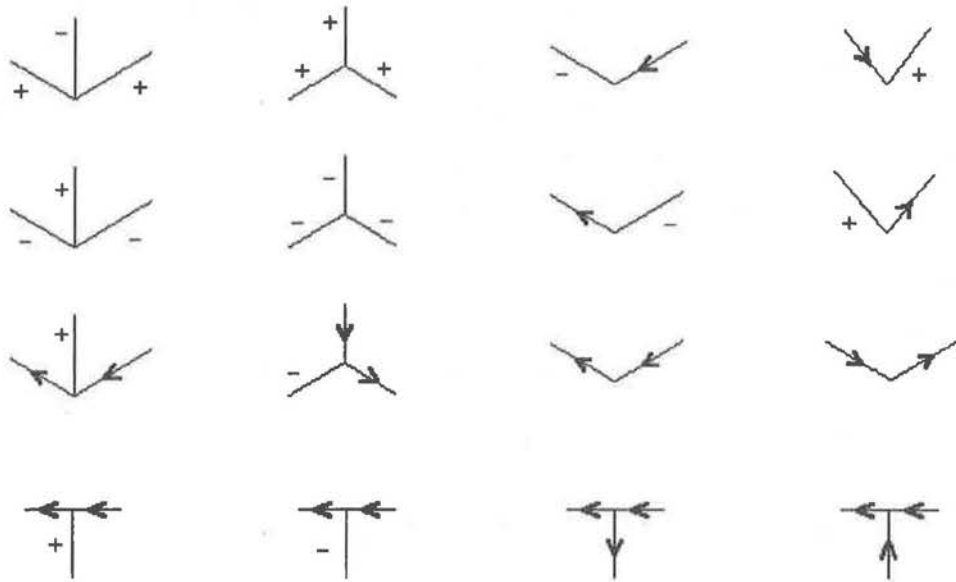


Figure 3.20: Huffman-Clowes labellings for rectangular objects.

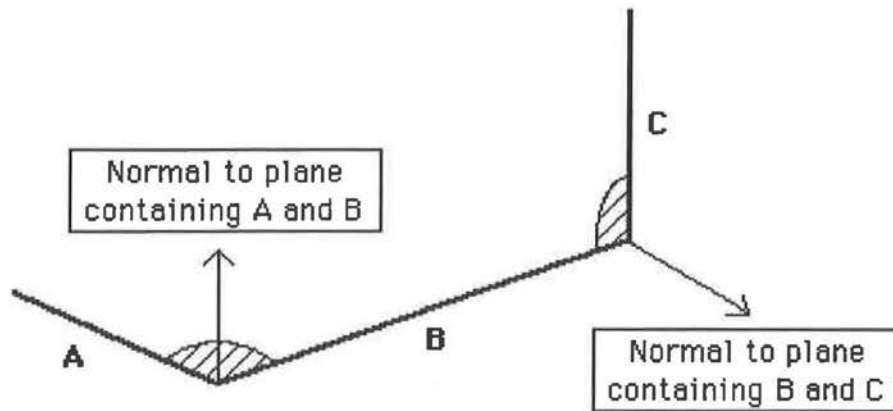


Figure 3.21: Planarity constraint. Surface normals for a common surface must be the same.

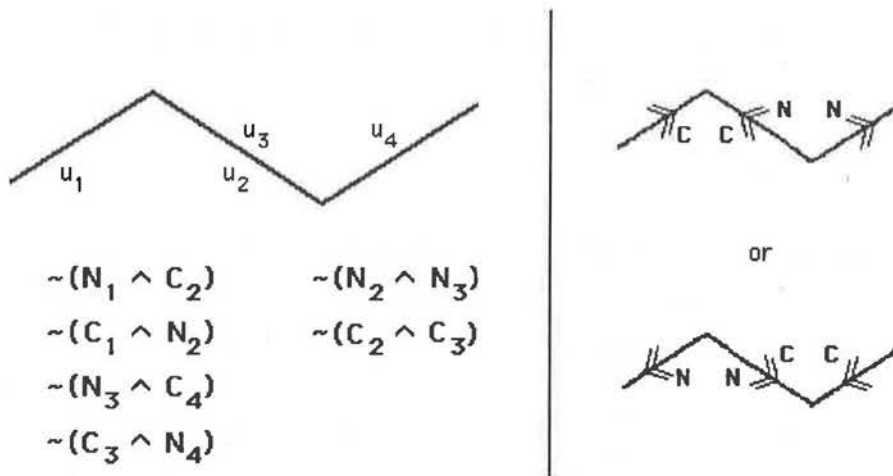


Figure 3.22: Interior angle constraint.

directions in the image cannot all be labelled as contiguous (figure 3.21). This constraint applies to both sides of the chain. As figure 3.21 shows, this constraint stems from a prohibition against having different surface normals defined from each of the line pairs.

Another quantitative constraint is the *interior angle constraint*: if a slant sign is to have local consistency, any chain of two connected obtuse L-junctions must alternate in the consistency labels attached to the edges on their interior angles (figure 3.22). This arises from the close connection between slant sign and contiguity for these L-junctions (figure 3.19), together with the requirement that slant signs on these junctions must alternate.

B. Complexity of Labelling Rectangular Objects

Since the convexity labelling of edges involves only binary bijective constraints, proposition 3.4 ensures that this can be reduced to CCL. As for the other domains discussed here, only two convexity interpretations are possible for each partition. And as before, compatibility can be ensured by first solving for convexity and then requiring lines labelled as '+' to be doubly contiguous.

In order to reduce contiguity interpretation to a low-complexity problem, constraints must be put into an appropriate form. The constraints on arrow- and T-junctions are already binary and bijective, as are the constraints on acute L-junctions. Since the contiguity constraints on Y-junctions can also be described by a set of binary bijective constraints (section 3.2.1), the reduction of the contiguity labelling problem centers on the constraints for obtuse L-junctions and lines,

i) Reduction to 2-SAT

Lemma 3.1 *For rectangular objects, the planarity and interior angle constraints allow the constraint against the 4-way contiguity of obtuse L-junctions to take on binary form, this reformulation having complexity no greater than that of CCL.*

Proof: If an obtuse L-junction is isolated, simply assign it one of the legal labellings of figure 3.19. Via an exhaustive enumeration of all possibilities, it can be seen that the planarity constraint rules out a joining of acute and obtuse junctions. Unique values can also be assigned when the shared edge is an "outer" edge of an arrow-junction, with the stem pointing away from the interior angle (figure 3.23). When the stem points towards the interior angle, a binary (although not bijective) constraint can be imposed on the possible values. A unique set of contiguity labels can also be made possible for Y-junctions by invoking the planarity constraint (figure 3.23).

Otherwise, all shared edges are with others of the same type, and so the junction is part of a chain of obtuse L-junctions. These chains can be detected in a preprocessing step based on CCL. In such a chain, each interior side of a junction is an exterior side of its neighbor. And since the interior angle constraint forces the interior labellings of neighbors to be different, it is impossible that such a junction can have both of its interior and exterior sides labelled as contiguous. Since only a direct assignment of values and constraints to local configurations are involved, the complexity of these

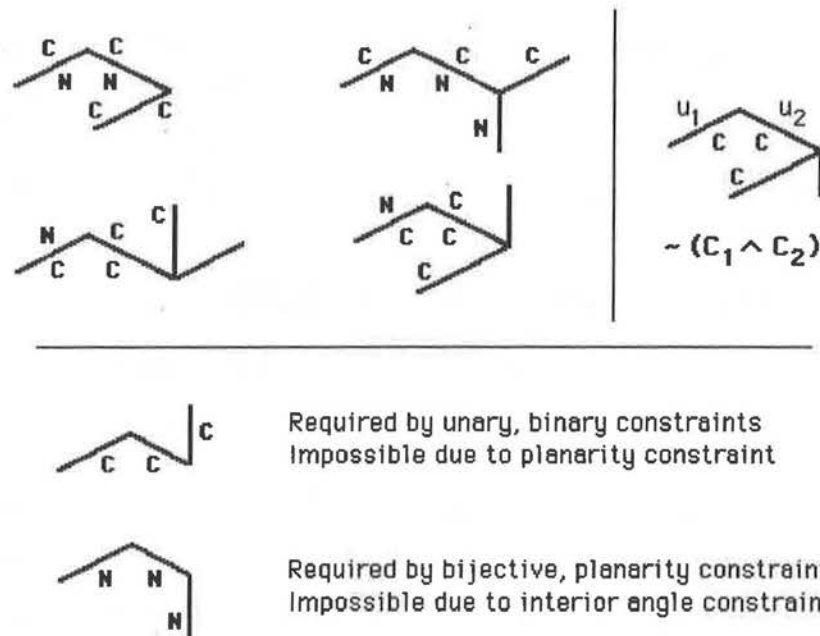


Figure 3.23: Contiguity constraints on obtuse L-junction combinations.

operations are no greater than that of CCL. The proof then follows from the observation that the interior angle constraint is both binary and bijective. ■

Theorem 3.7 *For rectangular objects, the planarity and interior angle constraints allow line labelling to have a complexity no greater than the maximum of that of 2-SAT and CCL.*

Proof: Since lines can be handled by a binary constraint (section 3.2.1), it is only necessary to express the constraint against the 4-way contiguity of obtuse L-junctions in binary form. From lemma 3.1, it follows that these can be cast into the appropriate form via a preprocessing step that assigns unique values to many of the obtuse L-junction labels, and binary constraints to the rest. And it follows from the lemma that this step has a complexity no greater than CCL. ■

These are the same complexity bounds found by Kirousis and Papadimitriou [KP88] for the somewhat more restricted case of the *orthohedral world*, in which all edges are required to be parallel to one of the three main axes in the scene. The addition of explicit planarity and interior angle constraints, therefore, makes similar low-complexity recovery possible for the more general domain of rectangular objects.

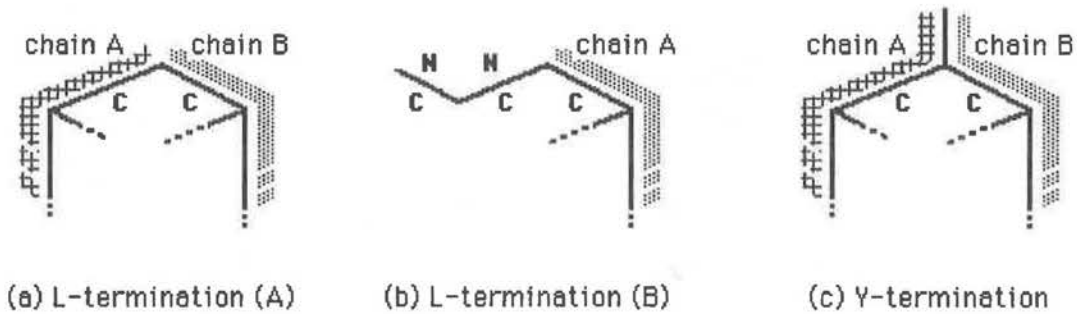


Figure 3.24: Termination configurations.

ii) Reduction to CCL

As for the case of convex and connected convex objects, it is also possible to show that line labelling for rectangular objects is of complexity no greater than that of CCL. This requires a careful isolation of the remaining binary constraints on obtuse L-junctions and on lines.

Lemma 3.2 *Bijjective constraints can determine the correct contiguity labelling of obtuse L-junctions, except for the case of “L-terminations”, in which an L-junction contacts an arrow-junction with its stem oriented toward the interior angle (figure 3.24).*

Proof: As lemma 3.1 shows, most values on obtuse L-junctions can either be uniquely assigned or given bijjective constraints, except for the case where it contacts an arrow-junction with its stem pointing toward from the interior angle.

If the L-termination is of type A (i.e., free variables for both exterior edges), the exterior edges require some additional constraint to ensure that they cannot both be contiguous. If the L-termination is of type B (i.e., free variables on only one of the exterior edges), the side contacting the L-junction is uniquely determined, and so no constraints are needed for the other side. ■

It must now be shown that there is no need for an explicit binary constraint against doubly-discontiguous lines.

Lemma 3.3 *Bijjective constraints alone can enforce the prohibition against double discontinuity, except for the case of a “Y-termination”, i.e., a configuration in which a dangling edge contacts two arrow-junctions with stems oriented away from that edge (figure 3.24(c)),*

Proof: If no junctions are present, a line can immediately be given any legal labelling. Otherwise, as figures 3.19 and 3.20 show, the prohibition against double discontinuity is automatically imposed for all junctions except obtuse L-junctions and Y-junctions. As shown in figure 3.23, the constraints on obtuse L-junctions ensure that each line has at least one contiguous side. This leaves only Y-junctions to be considered.

The only constraints on Y-junctions are the set of bijective constraints shown in figure 3.6, which require that the same value be assigned to lines contacting a common region. If a drawing does correspond to a scene containing rectangular objects, however, two Y-junctions will never contact each other, for to do so would immediately violate the planarity constraint. The constraint against double contiguity can therefore be inherited directly from the junctions that the Y-junction contacts.

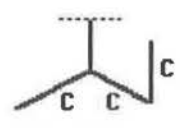
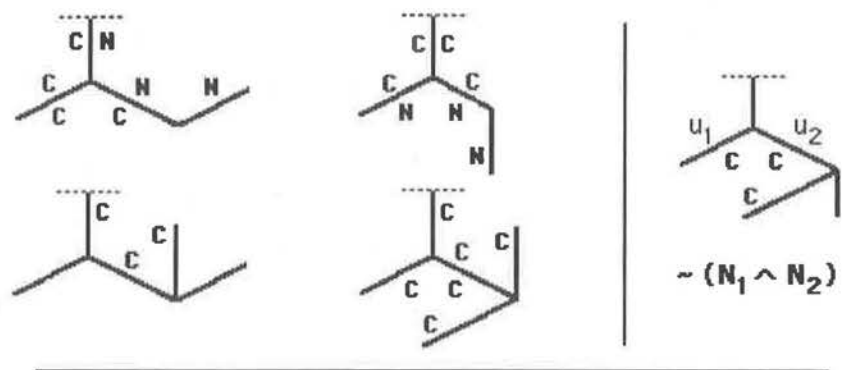
Complications arise from the dangling edges arising from occlusion (i.e., the stems of T-junctions), but these can be handled by a preprocessing stage:

If all three edges are dangling: the junction can simply take on any legal interpretation.

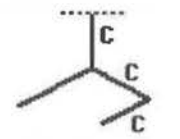
If two edges are dangling: the remaining edge necessarily contacts another junction, and so obeys the constraint; the inner edges of the dangling lines are undetermined, but without loss of generality the appropriate constraints can be enforced by requiring these to be contiguous.

If only one edge is dangling: An exhaustive examination of all cases (figure 3.25) shows that the planarity constraint ensures the appropriate contiguity condition for all configurations except the Y-termination. ■

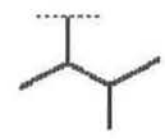
Given lemmas 3.2 and 3.3, it must now be shown that the remaining contiguity constraints on L- and Y-terminations can be handled appropriately. Any complex beginning at an L- or Y-termination can be seen as a "chain" that travels along the sides of Y-junctions and the outer sides of arrow-junctions, and terminates at another L- or Y-termination. These chains are similar to those used to analyze the interpretation of compound convex objects (section 3.3.2), and are handled in much the same way. Since all other aspects of the line interpretation can be handled by bijective constraints, it is only necessarily to show that the chains can also be labelled in a consistent way via a process of complexity no greater than that of CCL.



Required by unary, binary constraints
Impossible due to planarity constraint



Required by unary, binary constraints
Impossible due to planarity constraint



Cannot exist for a rectangular object
Impossible due to planarity constraint

Figure 3.25: Contiguity constraints on Y-junction combinations.

Contiguity labelling can be carried out by first assigning labels to variables constrained to take on unique values, and to those chains that contain such a variable. Next, any chain with more than two orientations to its set of edges must necessarily be discontinuous if it is to obey the planarity constraint; consequently, each free chain that remains cannot bend, but must travel in one general direction only.

The remaining free chains can now be labelled. Note that since these chains are decoupled from the rest of the interpretation (section 3.3.2) it does not matter whether this is done before or after determining labels for the rest of the drawing. Indeed, it follows from lemma 3.2 and the bijective nature of the constraints that all variables outside the chains will have only one possible value.

In order to reformulate the constraints on the remaining free chains, the range of possible interpretations is restricted in a manner similar to that employed in the other two domains, viz., a restriction such that a solution for the restricted variant exists if and only if a solution exists for the more general case. Consider first the chains that are connected together cyclically in a group, i.e., connected by common L- or Y-terminations¹⁵ If the number in such a group is even, let all termination configurations be contiguous on one side only. If the number is odd, pick a termination configuration and set both of its sides to be contiguous if it is a Y-termination or discontinuous if it is an L-termination, and then constrain the remaining configuration to be contiguous only on one side. This results in a set of bijective constraints that allow all chains in the cycle to be interpreted while continuing to prohibit double discontinuity. It is evident that this process can be carried out in parallel for all cyclic chains in the partition.

Those chains that are not cyclic must contact a termination configuration for which one side of the central L- or Y-junction has already been assigned a definite contiguity value. If there is only a single chain between such junctions, it can be determined in a fixed amount of time whether or not it can be given a value consistent with those already assigned; this is possible exactly when a legal labelling exists for the drawing. A similar situation holds for two connected chains.

¹⁵The common L-terminations are necessarily of type A.

If three or more chains are connected together, a slightly more complex procedure can be used:

1. Determine the values for the endpoints if they are to have a legal labelling of their termination configurations.
2. Constrain all terminations between the chains along this path to have sides of opposite contiguity.
3. If the number of chains is even and the endpoints have different contiguity labels, the alternation of contiguity at the terminations will suffice for a legal labelling. A similar situation holds when the number of chains is odd and the endpoints have the same contiguity labels. Otherwise, pick one of the inner termination configurations:

If it is an L-termination: it must be of type A; constrain both sides of its central junction to be discontinuous.

If it is a Y-termination: constrain both sides of its central junction to be contiguous.

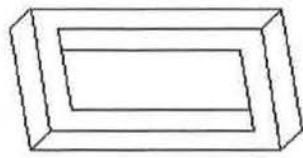
The result of this process is a sequence of chains that must alternate in value at each termination condition, except at the configurations described above. A solution for this set of constraints is possible exactly when a legal labelling can be obtained for the drawing.

The detection of the chains and the propagation of values along their extent can be carried out entirely by CCL. Since the constraints along these chains are bijective, their solution can also be obtained via this procedure. And since both convexity and contiguity labelling can be reduced to CCL, this yields

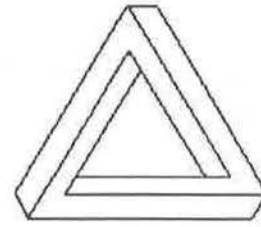
Theorem 3.8 *For rectangular objects, the planarity and interior angle constraints allow line labelling to have a complexity no greater than that of CCL.*

C. Slant Sign Constraints

For rectangular objects, constraints exist on the slant sign of each edge in the drawing (sections 3.2.3 and 3.2.4). The resultant set of constraints are shown in figure 3.13. Since all constraints are bijective, the entire drawing is implicitly linked together in one entire complex with only two possible interpretations. From proposition 3.7, it then follows that the determination of slant signs can be reduced to CCL under these conditions.



(a) Drawing detected as impossible rectangular object
- cannot be given consistent set of slant sign labels



(b) Drawing not detected as impossible rectangular object
- can be given consistent set of slant sign labels

Figure 3.26: Slant sign constraints applied to impossible figures.

Described in this way, the determination of slant sign does not rely on any other system, and so can serve as an independent source of constraint on the final interpretation. Indeed, the use of slant sign yields a process of higher accuracy than that obtained from qualitative labelling alone [Kan90], since it permits a greater rejection of drawings that cannot correspond to a rectangular object (figure 3.26(a)). However, because only local orientations about the junctions are involved, these additional constraints are not sufficient to eliminate all impossible figures (figure 3.26(b)).

Constraints on slant sign not only provide more information about the corresponding polyhedron, but can also speed up the interpretation process itself. For example, if a Y-junction has all three edges slanted away from the viewer, it must correspond to a convex corner. Similarly, if the stem of an arrow-junction slants away from the viewer, it must be concave and the other edges convex. As shown in figure 3.19, the slant sign determines the labelling of acute L-junctions and of obtuse L-junctions for which the edges slant away, allowing all contiguity constraints to be put immediately into binary form. Note also that if the slant sign stream is used (together with the convexity stream) as the basis for contiguity interpretation, it eliminates the need for explicit planarity and interior-angle constraints.

D. Slant Magnitude Constraints

Slant magnitudes are constrained via equation 3.5. It follows from this equation that the magnitude of one edge immediately determines that the other (section 3.2.4). When only two line directions are present in a partition, slant magnitude is underconstrained; the slant magnitudes of the edges can then be fixed simply by assigning some arbitrary value to one of

the directions. Since the parallel lines in each partition represent parallel edges in the scene, once the particular magnitudes have been chosen, CCL can be used to propagate them to all lines in the partition.

A similar situation exists when three different line directions exist in the image, except now the slant magnitudes are uniquely determined. As discussed in section 3.2.4, the determination of the slant magnitudes for this situation can also be reduced to CCL.

E. Robustness

i) Image perturbations

The qualitative and quantitative interpretation of rectangular objects relies on a special form of the general viewpoint assumption, viz., the assumption that slants in the scene are never zero, and consequently, that lines in the image are never perpendicular to each other. Although this assumption is sufficient for theoretical purposes, any practical system must be able to compensate for errors that arise from the measurement of image properties. As such, an additional set of techniques is required to ensure that the interpretation process remains robust against small perturbations of the input image.

For the interpretation of rectangular objects, perturbations have their greatest effect when one or two edges have a slant differing only slightly from zero. When only one of the edges is very shallow, the other two are in a plane closely aligned with the line of sight; as a consequence, their projections onto the image are nearly at right angles to the projection of the shallow edge (figure 3.27(a)); among other things, this makes it difficult to distinguish arrow- and Y-junctions from T-junctions. If two of the edges are shallow, their projections are nearly at right angles to each other (figure 3.27(b)), making it difficult to distinguish between acute and obtuse L-junctions. As such, shallow edges can cause a potential instability in the labelling of convexity, contiguity, and slant sign. Furthermore, from equation 3.5 it also follows that estimates of slant magnitude are also sensitive to small errors in line orientation angle θ ; under these conditions.

One way to obtain robustness against such perturbations is to alter slightly the angles of the lines in the junctions, setting them to values that are all the same. This helps both to remove the effect of local perturbations, and to reduce the effects of perturbations introduced at any later stage of processing. The exact procedure depends on which of the two situations

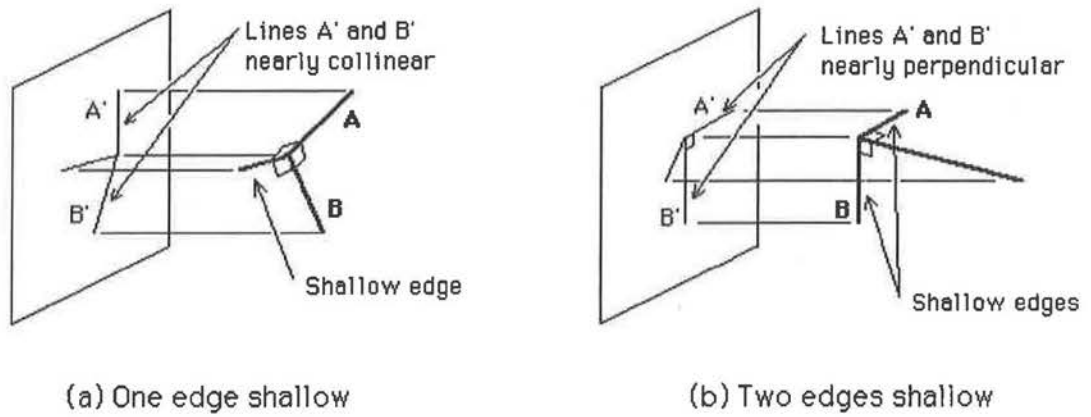


Figure 3.27: Conditions of shallow slant.

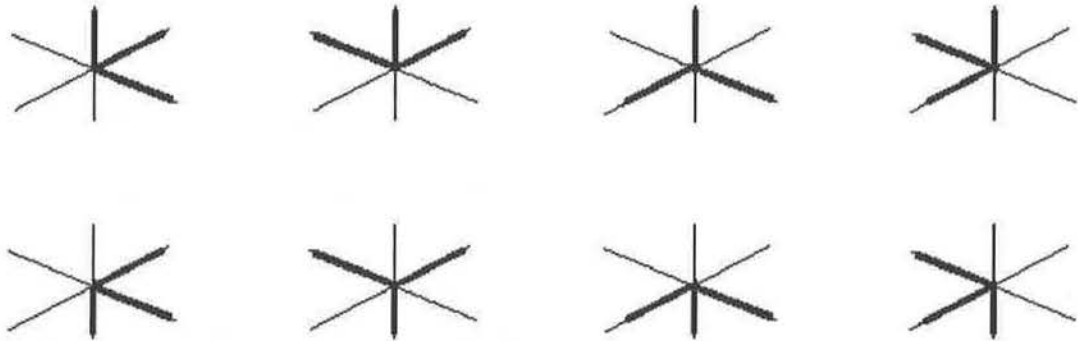


Figure 3.28: Combinations of angles into corners.

is encountered.

In both cases, the procedure begins by obtaining the distribution of line directions in the partition. Ideally, only three directions would exist, corresponding to the three directions of the edges of the corresponding object. If more than three exist, a procedure (e.g., taking the mean of each of the three distributions) can be used to remap the lines onto a smaller set of angles. Once determined, these new values can be broadcast to all junctions. This remapping applies to all possible corners (figure 3.28), since it follows from equation 3.5 that slant magnitude is indifferent to the particular combination chosen. Since a similar reassignment can also be used for the final set of directions obtained, alteration of line direction can be based entirely on a pair of canonical junctions, obtained from the appropriate rearrangement of lines in the image (figure 3.29).

Consider first the case where one edge has a shallow slant. As figure 3.27(a) shows, this is signalled by the existence of two nearly-parallel line directions in the image. Using the

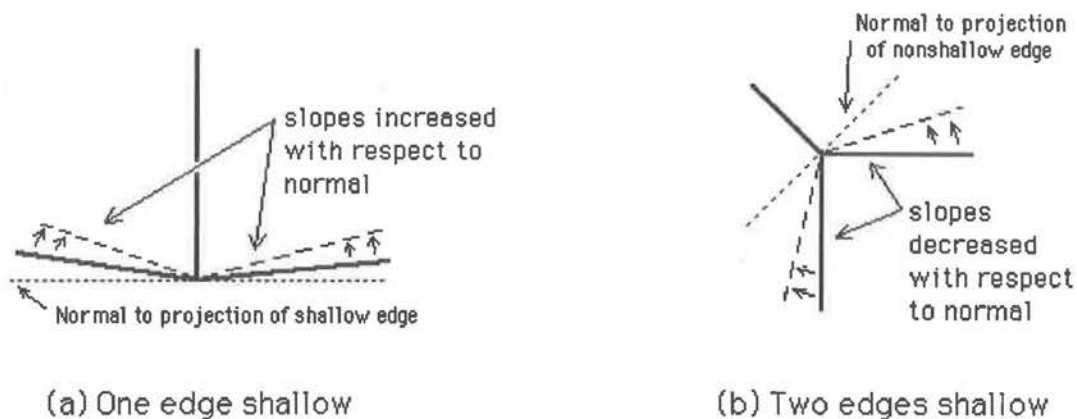


Figure 3.29: Rescaling of image angles.

normal to the shallow edge to complete a local coordinate frame, the slopes of the other two lines can be rescaled until at least one has a value δ_{min} (figure 3.29(a)).

Such a rescaling of orientations corresponds to a rotation of each corner of the object about an axis perpendicular to the shallow edge, with the slant of the shallow edge being increased. Note that this is not a true rotation of the object as a whole, since this requires a change in the distances between junctions as well. But if the rotation is small, the change in foreshortening is negligible, and the transformation can be interpreted as a shift in viewing position that results in more robust interpretation.

A similar technique can be used when two shallow edges exist. From figure 3.27(b), it is seen that this condition is signalled by the existence of two line orientations that are nearly perpendicular to each other. As for the case of one shallow edge, slopes can be rescaled, except that now the rescaling is done with respect to an axis perpendicular to the edge which is *not* shallow (figure 3.29(b)). The resulting transformation corresponds to a rotation about an axis at right angles to the nonshallow edge, the rotation serving to increase the slant of the two shallow edges.

If it is necessary to apply both transformations to the lines of an image, this can be done simply by applying the required corrections in some fixed way. Thus, provided that three different line directions can be distinguished in a partition, they can always be remapped into a new set of orientations that can disambiguate any local ambiguities caused by small perturbations in the input image, and that minimize the effects of any other perturbations that might be introduced by subsequent processing stages.

ii) Perspective distortion

The results obtained here assume the scene-to-image projection to be orthographic, i.e., rays from the scene contact the image plane at right angles. Since the scene-to-image mapping is the same at all points in the image, a spatially-uniform set of rules can therefore be used to recover the scene structure. This greatly simplifies the development and analysis of the recovery process.

Orthographic projection is almost always a good approximation of the perspective projection that actually governs the mapping of objects onto the image plane. However, it breaks down when an object extends over a large fraction of the visual field. In such a case, only one point corresponds to a perpendicular projection from the object to the image plane, and a “radial” distortion arises that is centered about this point. Although this distortion complicates the recovery process, it does not affect its interpretative power — a global transformation of the image can always be found that maps each junction to its equivalent under orthographic projection (see, e.g., [Kan90, ch.8]). Consequently, both qualitative and quantitative structure can always be recovered.

Since the emphasis of this work is primarily on rapid recovery and not on robustness *per se*, special corrections for perspective distortion are not developed here. Note that perspective distortion alters only the angles and lengths of lines in the image, and so the basic qualitative aspects of the recovery process are largely unaffected. Furthermore, if these distortions are small, the angles can be realigned by the broadcast mechanism used to handle small perturbations in the input. Thus, the only situation not encompassed by this approach is the relatively rare case where the projection of an object extends over a considerable fraction of the visual field.

Chapter 4

Computational Analysis

A computational analysis describes and justifies an image-to-scene mapping that is (i) unique, and (ii) possible within the given external and internal limitations (section 2.4). For the mapping considered here, the external limitations are that the information comes from a single orthographic projection of a blocks world scene of the type described in section 1.1, and that a constant amount of time is available for its operation (section 2.5.2). Internal limitations are that a mesh architecture is used, and that the processors have a fixed number of states. This chapter develops a set of constraints that defines a process capable of recovering a large amount of the scene structure within these limitations.

To ensure that local computation is relatively simple, a set of external constraints is chosen that limit the range of possible mappings to those that can be determined in sublinear time.¹ A set of internal constraints is developed to control the search through the space of possible solutions. These constraints are chosen so that a reasonable chance exists of finding a plausible interpretation within the allotted time.

The constraints developed here, of course, are not necessarily those used by the human early vision system. Many factors are potentially involved in the rapid recovery of three-dimensional structure, not all of which are known or fully appreciated at the present time. As such, this analysis is not intended primarily as a definitive treatment of the rapid recovery process, but rather as an illustration of how a computational analysis of this process *can* be carried out.

¹More precisely, the complexity of the mapping must be a sublinear function of the number of lines in the image (see section 2.5.1.)

4.1 External Constraints

As discussed in section 2.5.2, the output of a rapid recovery process is a dense set of estimates assigned to each spatially-limited patch (or “zone”) in the image.² These estimates must be both locally consistent and computable in a constant amount of time. If they are to have a good chance of corresponding to the actual structure of the scene, the corresponding property must be easy to compute and use minimal information from outside the zone.

Given the correlation between the amount of nonlocal information that must be transmitted and the complexity of an operation (section 2.1.1), it follows that recovered quantities must be computable by a low-complexity process, ideally one of no more than logarithmic complexity (cf. chapter 3.1.2). This can be ensured by an appropriate choice of external constraints (section 2.4) on the final form of the mapping. These constraints serve to eliminate those mappings that cannot be computed in sublinear time.

4.1.1 Image-to-Scene mapping

For a rapid recovery process, the goal is to reconstruct as much of the three-dimensional scene as possible, with interpretations required to be consistent only over spatially-limited zones (section 2.3.2). This goal is somewhat different from that of the “classical” problem of line interpretation, which assigns unambiguous interpretations to each line, and completely rejects drawings that cannot be given a globally consistent interpretation. Consequently, the image-to-scene mappings need not be the same for the two types of tasks.

Possible differences in the image-to-scene mapping include not only differences in the particular associations between input and output quantities, but also differences in the quantities themselves. The first step to find a mapping suitable for rapid recovery is therefore to determine an appropriate set of inputs and outputs.

A. Basic Quantities

A fixed limit on time translates into a fixed limit on the distance over which information can be transmitted. If recovery is to be robust with regards to this limit, it cannot be based on global properties (e.g., the number of features present in the image), or on extensive

²The exact size of these zones is not critical, the main constraint being that they are small enough that each contains no more than a few line segments (section 4.3.1).

properties (e.g., the lengths of lines and edges). Instead, it involves only those properties that can be determined *locally*, i.e., over arbitrarily small areas of the visual field (cf. section 2.1.1). Locality applies to properties of both the input and the output.

In what follows, the basic quantities in the image domain are taken to be the two-dimensional orientations of the lines and the locations of their endpoints.³ The quantities in the scene domain are taken to be the (positive) convexities, slant signs and slant magnitudes of the edges, as well as the contiguity relations between edges and surfaces. In particular, the form of the output is taken to be exactly that used in chapter 3.1.2 — a set four spatiotopic maps, one for each property, in which the variables describe the absence or presence of the corresponding quantity. All properties are assumed to be “dense”, being attached to all points along the lines in each zone. Note that this differs from the “sparse” form used in the “classical” problem of line interpretation (section 2.2.1), where properties are attached to individual *lines*, rather than points (see, e.g., [Mal87]).

This choice of properties is motivated not only by the requirement that the properties be local, but also by the results of chapter 3.1.2, which show that these quantities can indeed be rapidly recovered for several sub-domains of polyhedral objects. Note that these are not “template” properties, which can be calculated reliably on the basis of local information (section 2.1.1), but instead require at least some nonlocal information for their complete determination. However, the low complexity indicates that relatively little nonlocal information is needed. This is the key to the effectiveness of a rapid recovery process — *even though nonlocal information is generally needed for a complete local interpretation, at least some of this structure can be rapidly recovered if the amount of information needed is small.*

Of course, other quantities (such as the slants of the surfaces) could also be used, and conversely, some of the quantities used here may not actually be recovered by the human early visual system. But the quantities chosen here encompass both the qualitative and quantitative aspects of line interpretation (section 2.2.1), and are therefore adequate for present purposes, viz., illustrating how rapid parallel recovery *can* be done.

B. Isolation of Indeterminate Values

Since the interpretation provided by a rapid-recovery process does not need to be consistent over the entire image (section 2.3.2), the surrounding interpretations do not need to be

³Locations are always relative to a particular zone, so that absolute coordinates are not needed.

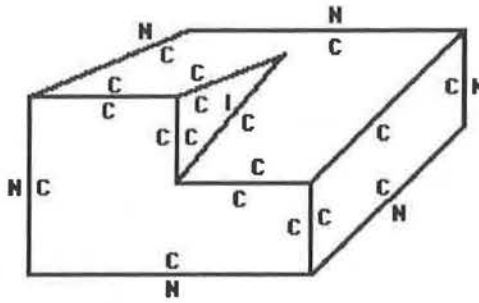


Figure 4.1: Isolation of inconsistency in contiguity labelling

removed when a local inconsistency is found. Indeed, this is not even desirable, for the propagation required for this removal can take considerable time (cf. section 3.1.2), and also results in information being lost from those areas which do obey the assumptions. Similarly, the limited time available may be insufficient to completely determine all interpretations, so that ambiguities also must be explicitly handled in some way.

In order to allow these possible outcomes to be explicitly signalled, the set of output states is expanded to include an 'I' label for inconsistencies, and an 'A' label for ambiguities. These can be applied to any variable in any dimension.

By careful application of these labels, indeterminate values can be isolated so as to allow a definite interpretation to be given to the other parts of the drawing. An example of this is shown in figure 4.1. If the contiguity constraints of section 3.3.3 are used on this figure, a globally consistent labelling is not possible. Attaching an 'I' label to one of the lines on the central notch, however, allows the remaining edges to be given a definite interpretation.

In order that these labels can be used wherever needed, no explicit constraints are placed on their application; if necessary, they can be assigned to all lines in drawing. However, the use of these labels must be minimized if the greatest amount of information is to be obtained about the three-dimensional structure of the scene. Since this cannot be done by constraints on the static assignment of the labels, it must be done via constraints on the process that generates these assignments (section 4.2).

4.1.2 General Principles

Chapter 3.1.2 shows that low-complexity approximations can be formed in many different ways, depending on the constraints selected. It is now necessary to select one particular set of constraints, and to justify this selection. Three general principles are relevant here.

A. Separation of Dimensions

As shown in chapter 3.1.2, a low-complexity mapping can be obtained when external constraints apply primarily to simple variables within separate dimensions, with only a limited amount of interaction between the corresponding streams. This strategy is taken as a basic principle here. In particular, all constraints that are nonbijective (i.e., do not have a 1:1 mapping between allowable values — see section 3.1.2) must involve only two variables, each with two possible values.⁴ This ensures that the problem is easy to solve (section 3.1.2).

Since it has been shown to lead to low-complexity mappings for many subdomains (section 3.3), the set of dimensions used here is exactly that of chapter 3.1.2: contiguity, positive convexity, slant sign, and slant magnitude.

B. Locality of Constraints

Most constraints used in line interpretation (sections 2.2.1 and 3.3) are local, pertaining to individual lines and junctions. An example of this is the requirement that all edges in a Y-junction must be labelled as '+' or 'o' (section 3.2.2). Constraints such as the interior angle constraint (section 3.3.3), however, involve relations *between* junctions, and so are not of this form. Since nonlocal constraints require image-based as well as scene-based information to be transmitted, they increase the demands on computational resources. They are also awkward to enforce on a mesh processor, where information travels at a constant speed across the image (section 2.5.2). Only local constraints must therefore be used.

As a special case of this principle, note that it is impossible to ensure that a single label can be attached to any line (section 2.2.1), since a line can extend over a considerable distance in the image. Consequently, constraints on junction labellings are still allowed, but they now apply to line *segments* of fixed length rather than lines of arbitrary length. An auxiliary set of constraints is required to constrain neighboring segments to take on the same values.

⁴Note that more values are possible for these variables, but that only two must enter into the nonbijective constraints. This allows 'I' and 'A' labels to be used in addition to the two definite values, since they do not enter into any explicit constraint.

C. Local Coordination of Dimensions

Much of the power of the original mapping can often be captured by a low-complexity approximation only if there is an interaction between the interpretations in the separate dimensions (section 3.3). Ideally, this would be carried out by a globally-coordinated sequencing between the different streams. Since global coordination is not feasible here, however, the interactions between dimensions must be reformulated to occur at a local level.

As discussed in section 3.3, the key to the successful integration of dimensions is the unidirectional transmission of information. But global coordination is not needed — this can be achieved by transmitting information from a local interpretation after it has been assigned an unambiguous value. For example, when the edges of a Y-junction have a unique labelling as convex, they are necessarily contiguous, and so can determine the corresponding interpretation in the contiguity stream. This kind of interaction allows local constraints to assist the interpretation process without any danger of increasing the complexity of the process.

4.1.3 Structural Assumptions

As shown in section 3.3, approximations of sublinear complexity can exist when appropriate restrictions are placed on the contiguity and convexity interpretations of L-junctions. These restrictions can be achieved via assumptions about the structure of the polyhedra, and as discussed in section 3.3, there are several sets of assumptions which can be used towards this end. In what follows, interpretation is based on constraints obtained from the assumption of rectangular corners (section 3.3.3).

There are several reasons for this choice. First of all, the visual system is exceptionally good at detecting junctions corresponding to rectangular corners, and using the rectangularity assumption to determine the three-dimensional orientations of the corresponding edges (see section 2.2.2). There is no reason to suppose that this preference is limited only to the higher stages of visual processing.

A second set of reasons involves issues of symmetry and structure. If an angle between two edges in a corner is unknown, 90° is a natural default, simply because it is midway on the range of all possible angles;⁵ in some sense, it may be considered to be an expected

⁵If edges are assumed to be unmarked, a rotation of 180° is an identity transform. Attaching an edge to a

value. Furthermore, the fact that all edges are perpendicular to each other makes it simple to convert between the slants of the edges and the slants of the faces: for rectangular corners, the normal to the surface corresponding to a region is parallel to the edge that is opposite it in the junction.

Finally, there are reasons based on computational complexity. The interpretation of rectangular objects requires relatively little in the way of processing resources, since it is among the least complex of all line interpretation problems. Rectangularity also allows edge slant to be determined rapidly by local processes, something not generally possible for the other domains considered in section 3.3. Indeed, the estimation of edge slant does not even need to wait for the qualitative analysis to be completed (cf. [Sug86]), but can proceed in tandem with the determination of contiguity and positive convexity.

4.1.4 System of External Constraints

Bringing together the requirements outlined above, the rapid recovery of three-dimensional structure is assumed here to be governed by the external constraints shown in figure 4.2. These involve four quasi-independent dimensions: contiguity, positive convexity, slant sign, and slant magnitude. The intra-dimensional constraints are given by the permissible labellings of the junctions; these are essentially the constraints developed in section 3.3.3. Interactions between dimensions, shown by the arrows in figure 4.2, are readily derivable from this same set of constraints.

This system of constraints is largely a reformulation of those of section 3.3.3. In particular, the nonlocal planarity and interior-angle constraints have been replaced by local constraints on slant sign, which then influence contiguity via the inter-dimensional interactions. There also exists a more direct local (nonbijective) constraint against doubly-discontiguous lines. Although not required to attain a process of logarithmic complexity when global coordination is possible (section 3.3.3), this constraint can improve the speed and power of the interpretation process when only local processing is allowed.

Note that the contiguity constraints on obtuse L-junctions cannot in general be put into binary or bijective form. In the absence of a definite interpretation for the two inner or two outer lines, only a single common constraint (that requiring both inner lines to take on the same value) can be applied. But the remaining constraints can be put into binary form if

corner does mark it, but two adjacent edges that differ by 180° are still effectively the same edge.

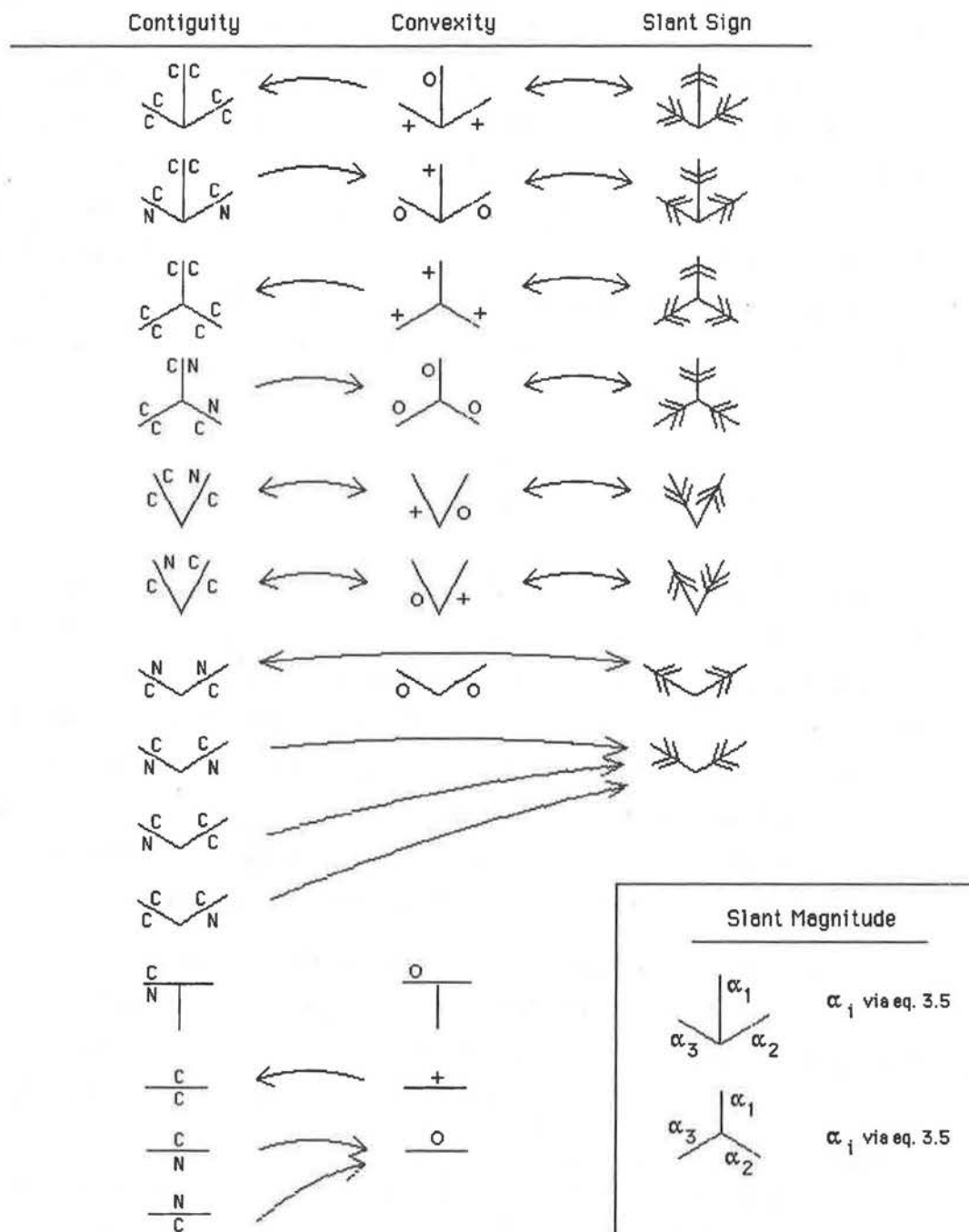


Figure 4.2: System of external constraints. Arrows indicate interactions between dimensions.

a dependency on the state of particular “trigger” variables is introduced so that constraints are put into effect only after a definite interpretation has been assigned.⁶

Consequently, all bijective and binary constraints on the junctions have been kept and the rest reformulated to take on one of these two forms. Although global co-ordination of the type required to achieve the results of section 3.3.3 is no longer possible, these constraints do allow the three-dimensional structure of the scene to be recovered. Given sufficient time, a consistent interpretation without ambiguities or inconsistencies is possible whenever the drawing corresponds to a set of rectangular objects. Because of the ‘I’ and ‘A’ labels, however, the requirements of section 2.3.2 are also met: a scene that does not contain rectangular objects everywhere may still give rise to local interpretations in those regions where the basic structural assumptions are obeyed.

4.2 Internal Constraints

Although external constraints limit the range of interpretations which can be given to a line drawing, they are not generally sufficient to determine its form completely. For example, the marking of edges as inconsistent or ambiguous must be kept to a low level (section 4.1.1), but this requirement conflicts with the prohibition against global measures (section 4.1.1). More generally, the interpretation process must operate within a fixed amount of time, and while the external constraints developed in section 4.1 do select a set of solutions that can be calculated quickly and with a minimum of nonlocal information, they do not provide any guidance as to what should be done when such limits are imposed.

To complete the specification of this mapping, therefore, an independent set of internal constraints must be imposed to help ensure that the best use is made of the available processing time.⁷ These are constraints on the *generation* of the interpretation itself (section 2.4.2). For the process here, these constraints are required to lead to a subset of interpre-

⁶These constraints, obtained from figure 4.2, are as follows:

1. If one of the inner edges is contiguous, no more than one outer edge can be contiguous.
2. If one of the outer edges is discontinuous, both inner edges must be contiguous.
3. If both inner edges are discontinuous, both outer edges must be contiguous.
4. If both outer edges are contiguous, both inner edges must be discontinuous.

⁷The use of such constraints is essentially an elaboration of Marr’s principle of graceful degradation [Mar82, p. 106], extended to cover not only reductions of available information, but reductions of other resources as well.

tations that have a relatively high likelihood of corresponding to the structure actually in the scene. Although the probabilities of various image-to-scene associations depend on the particular scene domain under consideration, exact knowledge of these probabilities is not generally necessary — all that is required is an ordering of the various candidates. As such, it is possible to provide a set of principles that are potentially applicable to many domains encountered in the natural world.

4.2.1 Processing Architecture

Internal constraints act on the flow of information that occurs during the course of computation. In order to develop such constraints it is first necessary to specify — at least at a general level — an “abstract architecture” that describes the way in which information processing and information transmission are carried out.

A. Processing over Zones

From the definition of the rapid recovery problem (section 2.5.2), the only processing resources assumed to be available are a spatiotopic mesh of processors, with each processor having a relatively small set of states. If good use is to be made of these resources, each processor (or group of processors) in the mesh must be assigned to a separate zone in the image, i.e., to a compact contiguous area of limited spatial extent (section 2.1.1).

This requirement stems primarily from considerations of efficiency. When only a fixed amount of time is allowed, each processor can only act on a fixed number of inputs.⁸ Since the number of processors increases with the size of the input (cf. section 2.5.2), effectiveness can be maintained by assigning each processor to a separate region of the image. And if processors are uniform in regards to their processing power (as assumed here), it is best if these regions have the same size.

Since transmission delays within a zone must be kept to a low level, regions must also be contiguous and compact.⁹ (This is related to the preference for local quantities described in section 4.1.1.) The demand for contiguity is forced not only by the need for compactness, but

⁸This is a generalization of an order-limited perceptron [MP69], with the output function being any function that can be calculated in a fixed amount of time.

⁹This restriction means that the process can be carried out by a generalized version of a diameter-limited perceptron [MP69].

by the recognition that the external constraints are highly sensitive to breaks in the lines, and since line drawings may fall anywhere, no part of the image can afford to be skipped. Contiguity also reduces the computational power required of the individual processors, since it is easier to handle a small set of unbroken lines than a large set of disconnected line fragments.

B. Communication between Neighboring Zones

If the mapping between input and output could be described entirely in terms of non-interacting zones, recovery could be carried out on an array of processors, each of which calculates only simple template properties of its corresponding zone. But if anything beyond the most rudimentary line interpretation is to be carried out, communication between processors is required. In what follows, it is assumed that the assignments of processors to zones maintains a spatiotopic organization and that neighboring processors in the mesh are assigned to neighboring zones in the image.

Once again, this is motivated by considerations of efficiency. Since all constraints between the local interpretations are themselves local, there is relatively little to be gained by having some other assignment of zones to processors. Furthermore, the operation of the local processors (as well as the analysis itself) is simplified, since the transmission of information takes place only via the zone-to-zone percolation of information through the "virtual mesh" formed by the lattice of zones over the image.

In particular, this information flow originates from zones containing an interpretable junction, and propagates at a constant rate along the connecting lines. Internal constraints therefore act by controlling the initial assignment of interpretations with a zone, and by controlling the propagation of these values along the lines of the drawing.

4.2.2 General Principles

At the most general level, internal constraints can take effect in two ways: (i) constraints on the basic operations used, and (ii) constraints on the representations operated upon. These effectively provide general constraints on the propagation of information around the "virtual mesh" that occurs during the interpretation process. Four general principles are used here to provide constraints on this propagation.

A. Maintenance of Interpretive Power

If the process is to have a good chance of recovering some part of the scene structure, it must not be too quick to throw away possible interpretations. Consequently, a liberal interpretation strategy is used: a candidate interpretation is kept unless an inconsistency is detected. Since inconsistencies are determined by the set of external constraints, this becomes the requirement that internal constraints must not exclude any interpretation consistent with the external constraints. In other words, internal constraints must not have any eliminative power — they must be entirely concerned with the ordering of the various possible solutions.

In order to allow all possible interpretations to be handled in a systematic way while keeping true to the demand that only two values exist in each constraint (section 4.1.2), the outputs of each of the four streams are split into two separate subsystems:

Contiguity: Two complementary subsystems to represent the possibility of contiguity and noncontiguity.

Convexity: Two complementary subsystems to represent the possibility of convexity and nonconvexity.

Slant Sign: Two complementary subsystems to represent the possibility of the two types of slant sign.¹⁰

Slant Magnitude: Two different subsystems — a quantitative subsystem to carry the value of the estimate, and a qualitative subsystem to represent the possibility that this value can legitimately be assigned.

For the complementary subsystems, the existence of a possible interpretation is signalled by a 'possible' state attached to the relevant edge, while its impossibility is likewise signalled by an 'impossible' state (figure 4.3).¹¹ The use of these subsystems allows all possible interpretations to be represented quite simply:

Definite: assignment of 'possible' to an edge in one of the subsystems and 'impossible' to its complement.

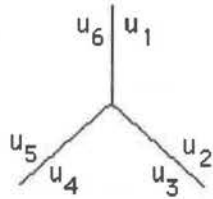
Ambiguous: assignment of 'possible' in both subsystems.

Inconsistent: assignment of 'impossible' in both subsystems.

¹⁰Slant towards or away from the viewer is not a pure scalar like the other two quantities — it has a directional component that must be taken into account (cf. section 3.2.3). This can be handled simply by having each subsystem represent an increase in depth as the line segment is traversed from one of the ends.

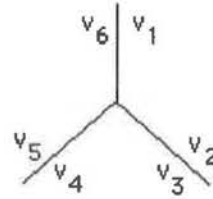
¹¹This is somewhat analogous to the use of relevance logic in reasoning (see, e.g., [Lev86]).

Contiguity Subsystem



$$u_i \in \{ \text{possible, impossible} \}$$

Noncontiguity Subsystem



$$v_i \in \{ \text{possible, impossible} \}$$

Figure 4.3: Example of complementary labelling.

B. Locally Irreversible Computation

If a process is to make good use of available time, it must avoid doing and undoing the same operations without any net effect. This requirement — essentially a form Marr’s principle of least commitment [Mar82, pp. 106-107] — rules out hypothesize-and-test strategies, favoring instead “one-shot” processes that require only a few steps for their completion. In the absence of global control, this must be done by a local mechanism that forces the process to avoid redundant processing while simultaneously ensuring that it will not exclude any consistent interpretation.

To combine this principle with that of maintaining interpretative power, the following scheme is used:

1. A ‘possible’ state is initially assigned to all values of all complementary subsystems as well as the qualitative subsystem of the slant magnitude stream.
2. Whenever a local inconsistency is found, the corresponding value is marked as ‘impossible’, and this value will never be withdrawn.

This is essentially a simple form of Waltz filtering (section 2.2.1), with an initial maximum uncertainty steadily reduced to the point where no local inconsistencies remain. Given the structural assumptions that have been made, little nonlocal information is required for a local interpretation. Convergence to a definite interpretation in each zone is therefore likely to be fast. Even if only a limited amount of time is available, the result is likely to provide at least some information to higher stages of processing.

The situation is similar for the slant magnitude stream, except that the qualitative subsystem serves to indicate the confidence of the corresponding magnitude estimate. When

local inconsistencies are found in the estimates of slant magnitude, an 'impossible' label is assigned to the variables involved, and then propagated along the line.

Note that the state of the interpretation can be described in terms of the distribution of the 'possible' states over the edges in the complementary subsystems. More precisely, local uncertainty exists only when a value and its complement are both possible. This allows the overall uncertainty attached to an interpretation to be described by an "entropy" measure that includes all the individual local uncertainties. Although never used by the actual process itself, this measure can provide a way to describe the overall state of uncertainty in the interpretation at any given moment.

Local irreversibility can be incorporated into complementary labelling by a reformulation of the constraints found at the local junctions. In order to preserve the power of the original set of constraints, this reformulation is subject to the following condition: any set of definite values ruled out by the original constraints of figure 4.2 must also be ruled out by the new set of constraints.

From figure 4.2, it is seen that all constraints except the contiguity constraints on obtuse L-junctions are "context free", i.e., the particular set of constraints depends only on the geometrical configuration in the image, and not on the set of labels attached (cf. figure 4.2). Reformulation is based on the idea that once such constraints have been set up, elimination of possible interpretations can occur by a simple priority mechanism that allows an 'impossible' state to replace any 'possible' state. Since 'possible' states are initially assigned to all variables, the reformulation involves only the ways in which 'impossible' states are to be transmitted.

The situation for state-dependent junctions is similar, except that no constraints are applied until definite assignments have been made to the inner or outer edges. Consequently, it is possible to reformulate the constraints in a way that allows the process to be locally irreversible while maintaining the complete set of external constraints:

i) Unary constraints

Since only two values can exist in a subsystem, a unary constraint (i.e., a constraint that acts on a single variable) necessarily requires the variable to have a unique value. For example, a unary constraint exists on the inside edges of an arrow-junction that force them to be contiguous.

Given two complementary subsystems, unary constraints can be easily enforced by marking the corresponding value in the complementary subsystem as 'impossible'. The value in the original subsystem, however, remains unaffected. This leaves a definite interpretation which can only be altered by becoming inconsistent (i.e., the value in both subsystems being 'impossible'). For an arrow-junction, therefore, the inside edges in the non-contiguity subsystem are marked as 'impossible' and the corresponding edges in the contiguity system retain their original state of 'possible'.

ii) Bijective constraints

Bijective constraints are such that a 1:1 correspondence exists between the values of the variables involved (section 3.1.2). For example, a bijective constraint exists on the contiguity labels of the inside edges of an acute L-junction, since both of these edges must be either contiguous or discontinuous (section 3.2.1).

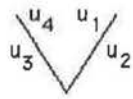
Since only two values exist for each variable, bijective constraints take on a simple form: either the variables have the same values, or else they have opposite values. If two adjacent lines are required to have the same values, both must have the same definite values, i.e., the corresponding variables in the complementary subsystem must be 'impossible'. This constraint can be enforced by the requirement that if one of the corresponding variables in a subsystem is 'impossible', so must be the other variable in the same subsystem. If two adjacent lines are required to have opposite values, their corresponding variables are similarly constrained, except that now the constraint applies to variables in "opposing" subsystems (figure 4.4). Note that this latter type of constraint provides a binding between the two subsystems, which are otherwise largely independent.

iii) Nonbijective constraints

The intradimensional constraints that are not bijective involve a single prohibition against a particular combination of values (figure 4.2). These constraints can be reformulated quite simply. If one of these values in such a prohibited combination definitely occurs (i.e., its complement is 'impossible'), then the other value must be excluded (i.e., its "direct" state is 'impossible'). Otherwise, nothing else is done. Note that in contrast to bijective constraints, only a one-way transmission of 'impossible' states is involved.

For example, if one side of a line has been marked as definitely discontinuous (i.e., its value in the contiguity subsystem is 'impossible') then the other side must be contiguous

Contiguity Subsystem

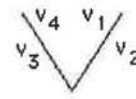


$$u_1 \in \{ \text{possible}, \text{impossible} \}$$

$$\text{if } (v_1 = \text{impossible}) \rightarrow (u_4 = \text{impossible})$$

$$\text{if } (v_4 = \text{impossible}) \rightarrow (u_1 = \text{impossible})$$

Noncontiguity Subsystem



$$v_1 \in \{ \text{possible}, \text{impossible} \}$$

$$(v_2 = \text{impossible})$$

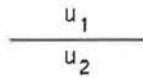
$$(v_3 = \text{impossible})$$

$$\text{if } (u_1 = \text{impossible}) \rightarrow (v_4 = \text{impossible})$$

$$\text{if } (u_4 = \text{impossible}) \rightarrow (v_1 = \text{impossible})$$

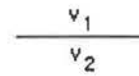
Figure 4.4: Example of reformulation of bijective constraint.

Contiguity Subsystem



$$u_1 \in \{ \text{possible}, \text{impossible} \}$$

Noncontiguity Subsystem



$$v_1 \in \{ \text{possible}, \text{impossible} \}$$

$$\text{if } (u_1 = \text{impossible}) \rightarrow (v_2 = \text{impossible})$$

$$\text{if } (u_2 = \text{impossible}) \rightarrow (v_1 = \text{impossible})$$

Figure 4.5: Example of reformulation of nonbijective constraint.

(i.e., its value in the discontinuity subsystem is ‘impossible’). But if one side is marked as contiguous, nothing else necessarily follows, and so no transmission results (figure 4.5).

iv) State-dependent constraints

The contiguity constraint to be applied to obtuse L-junctions depends on the interpretation attached to its edges. This kind of constraint can be reformulated in a straightforward way by imposing the appropriate set of constraints when the “trigger” variables take on definite values, i.e., when their values in the noncontiguity subsystem are ‘impossible’.

v) Interdimensional constraints

Since information from one dimension is sent to another only when a definite interpretation has been achieved (section 4.1.2), the reformulation of the relevant constraints is fairly

straightforward: when a particular set of values has been definitely assigned to the edges about a junction (i.e., the corresponding complementary values are 'impossible'), the associated values in the other stream can be given definite values (i.e., their complements are set to 'impossible').

Note that if a variable is deemed to be inconsistent, its value in both subsystems is 'impossible'. Consequently, the transmission of information across dimensions can cause the corresponding variable in some other dimension to also be labelled as inconsistent. Since processing time is limited, however, the propagation of these inconsistencies is unlikely to affect greatly the quality of the final interpretation, an assumption borne out by tests on a variety of line drawings (chapter 6).

C. Minimization of Inconsistency

It is important to control the propagation of labels so that minimal inconsistency results, i.e., 'impossible' are assigned to no more variables than necessary. This condition is automatically obeyed if the drawing corresponds to a rectangular object, for the constraints are such that appropriate values can always be assigned to the variables. Indeed, when redundant constraints are added, more routes become available for propagation, and the faster spread of 'impossible' states then speeds up the interpretation process.

However, when the scene contains objects that do not conform to the underlying structural assumptions, inconsistencies can arise in the resulting interpretation. Consequently, the more routes available for propagation, the greater the spread of inconsistent interpretations.

In order to limit the spread of such inconsistencies, therefore, some care must be taken when selecting the particular set of constraints to be used. In particular, if a variable is subject to a unary constraint, no other constraints must be applied to it.

For example, the inner edges of an arrow-junction are constrained to be contiguous. If they are also constrained to have the same value, the interpretative power of the system is not affected regarding rectangular objects, since no interpretation exists in which these lines can be assigned definite interpretations as discontinuous. However, if an 'impossible' label has been transmitted to one of these lines, such a constraint will cause it to be propagated to the others and assign them values that could never exist in any polyhedral scene. By disallowing such a constraint, the opportunity for inconsistencies to spread is minimized while the power of the original system of constraints is maintained.

D. Priority Marking

The preceding principles have brought with them a shift in emphasis from *individuals* to *ensembles*. But human perception tends towards individual interpretations. When viewing a Necker cube, for example, perception alternates between single unambiguous cubes, rather than being a superimposed set of all possible interpretations. If the recovery process is to reduce the set of interpretations that are simultaneously possible, and if it is to remain effective, it must focus on those that have the greatest likelihood of corresponding to the structure in the scene.

This can be done by marking such preferred interpretations as distinct. If this marking does not otherwise affect the interpretation process, it can allow priority to be given to the most likely candidates while still keeping available all other possible interpretations.

The simplest way to incorporate priority marking is to extend the set of values that can be given to a variable – in addition to ‘possible’ and ‘impossible’, include a ‘preferred’ state, which has priority over a ‘possible’ state.¹² In regards to all constraints developed so far, the ‘possible’ and ‘preferred’ states can be treated as equivalent. The introduction of this distinction can be viewed as a way to split the recovery process into two concurrent substreams, dealing with ensembles and individuals respectively. As such, the additional constraints required for priority marking must be limited entirely to ‘possible’ and ‘preferred’ values. The only exception is a requirement that when the complement of a ‘possible’ value is marked as ‘impossible’, the value itself must be upgraded to ‘preferred’. However, this “intra-stream” transmission does not affect the set of constraints dealing with ensembles, since these are involved entirely with the propagation of ‘impossible’ states. Consequently, the introduction of the possible-preferred distinction has no adverse effects on the ability of the recovery process to eliminate inconsistent interpretations.

If desired, the final output can be represented using “standard” labels that express the two definite interpretations, the inconsistent interpretation, and the ambiguous interpretation:

1. If one subsystem has a ‘preferred’ state and the other does not, take its value as a definite interpretation.
2. Otherwise, if both subsystems have ‘preferred’ or ‘possible’, set the interpretation to be ambiguous.

¹²Although interpretations can be even better distinguished by the use of several different priority levels, selection is usually from just a few alternatives, so that this system is sufficient for present purposes.

3. Otherwise, both subsystems must have 'impossible' states. Set the interpretation to be inconsistent.

Priority marking is a "dual" to the interpretation of ensembles. Instead of using a liberal strategy to eliminate impossible interpretations, it uses a conservative strategy to generate likely ones. It is therefore based on the same set of external constraints as the "ensemble" system, except that it involves 'preferred' and 'impossible' labels. Initially, all variables are set to 'possible', with the exception of a small initial set that are assigned a 'preferred' state. Constraints are enforced in much the same way as those of the ensemble substream, except that they involve the transmission of 'preferred' states to the "direct" subsystems instead of 'impossible' states to the complementary subsystems. Consequently, the complexity of priority marking is the same as that of the ensemble substream.

The only asymmetry between the two processes is that interpretations in the ensemble substream can override those of the priority substream, but not vice versa. In particular, when an 'impossible' state is assigned to a variable in some subsystem, it is effectively withdrawn from priority marking. In addition, the value of the corresponding variable in the complementary subsystem is upgraded from 'possible' to 'preferred'. This asymmetry reflects the basic difference underlying the assignment of the two kinds of label: possible-impossible distinctions are based on necessary consequences of the set of assumptions, while possible-preferred distinctions are generally based on considerations of likelihood.

Since complementary subsystems are not used for slant magnitude, there is no need to distinguish between 'possible' and 'preferred' values. However, 'preferred' labels can be used to signal when there is some evidence for a definite assignment of magnitude (e.g., magnitudes obtained via Perkins' laws). This proves especially useful in distinguishing slants that are zero by default from those that have been determined to be zero, since the latter can be treated equivalently to any nonzero slant magnitude.

4.2.3 Selection of Initial Candidates

The course of processing is controlled not only by constraints on the dynamic flow of information, but also by constraints on the initial interpretations that are considered. In particular, the priority marking mechanism developed in the previous section provides a way to distinguish a subset of selected candidates, but does not itself provide any principles to guide this selection. Although the most appropriate selection of initial candidates depends on the

particular domain being modelled, a few general principles appear to be widely applicable.

A. Contiguity

The first of these principles is that of *maximum interior contiguity*, which assumes that the inner surfaces of corners are contiguous whenever possible. This principle is an extension of the one used to reduce the complexity of labelling convex and compound convex objects by selecting a preferred set of interpretations (sections 3.3.1 and 3.3.2). It may also be a basis for the human perception of line drawings [Mac74][Hor86, p. 355].

The principle of maximum interior contiguity applies only to the contiguity stream. The particular set of junction markings, shown in figure 4.6, is as follows:

Arrow-junctions: Contiguity is preferred for all lines except for the outer pair, which do not generally form an interior angle. Since contiguity is necessary for inner lines, noncontiguity is impossible.

Y-junctions: Contiguity is preferred for all lines.

L-junctions (obtuse): Contiguity is preferred for the inside edges, since most of the possible interpretations assign them this value. The outer edges of these junctions, however, cannot be given preferred interpretations, for although one of these edges can be contiguous, the other cannot, and the symmetry of the situation makes it impossible to prefer one over the other.

L-junctions (acute): Outer edges are necessarily contiguous, while symmetry makes it impossible to prefer any interpretations of their inner edges (figure 4.2).

T-junctions: The crossbars have a necessary contiguity relation, so that preferred values follow immediately. No preference is given to values on T-junction stems, since these are effectively isolated lines.¹³

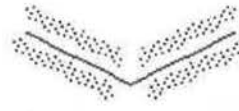
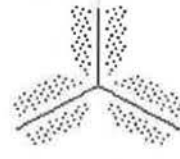
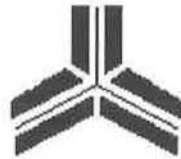
B. Convexity

Preference in the convexity stream is determined by the principle of *maximum convexity*, in which assumes that all trihedral corners in the scene have positive convexity. This principle

¹³Strictly speaking this is not true for more realistic surfaces, where the stem represents not only an edge of a different surface, but can also represent a crack or thin surface marking on the same surface. To address this issue more fully would require the development of a rapid-recovery system based on a more extensive set of labels, and this is beyond the scope of the present work.

Contiguity Subsystem

Noncontiguity Subsystem



■ ■ ■ ■ Impossible ▤ ▤ ▤ ▤ Possible ■ ■ ■ Preferred

Figure 4.6: Initial contiguity interpretations.

stems from the observation that convex corners are more common than concave ones; indeed, concave corners do not necessarily correspond to actual structures of the object itself, but may instead result from contact between adjacent objects [Bie85].

The initial set of junction markings in the convexity stream, shown in figure 4.7, is as follows:

Arrow-junctions: Convexity is preferred for the stems, while non-convexity is preferred for the outer wings.

Y-junctions: Convexity is preferred for all lines.

L-junctions (obtuse): All lines are necessarily non-convex, leading to preferred values in the nonconvexity subsystem.

L-junctions (acute): Although constrained to have one convex and one nonconvex side, symmetry makes it impossible to assign a preference.

T-junctions: The crossbars of the T-junctions correspond to occluding edges, and so are necessarily non-convex.

C. Slant Sign

Owing to the close connection that exists between convexity and slant sign when the corners are rectangular (section 3.2.3), the principle of maximum convexity can also determine preferred states for values in the slant sign stream. These are shown in figure 4.8. Since the corners are assumed to be rectangular, the convex edges in arrow- and Y-junctions correspond directly to edges that are slanted away from the viewer, and non-convex edges to edges slanted towards the viewer. Consequently:

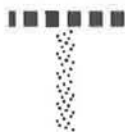
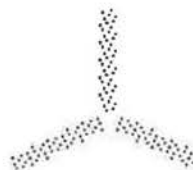
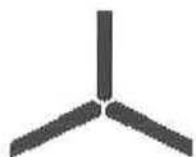
Arrow-junctions: Slant toward the viewer is preferred for the stems, while slant away is preferred for the outer wings.

Y-junctions: Slant away from the viewer is preferred for all lines.

Other junctions do not contain enough information to determine slant sign directly, and so no preference can be assigned on their account.

Convexity Subsystem

Nonconvexity Subsystem



■■■■ Impossible

..... Possible

———— Preferred

Figure 4.7: Initial convexity interpretations.

Slant Sign Subsystem - away

Slant Sign Subsystem - toward

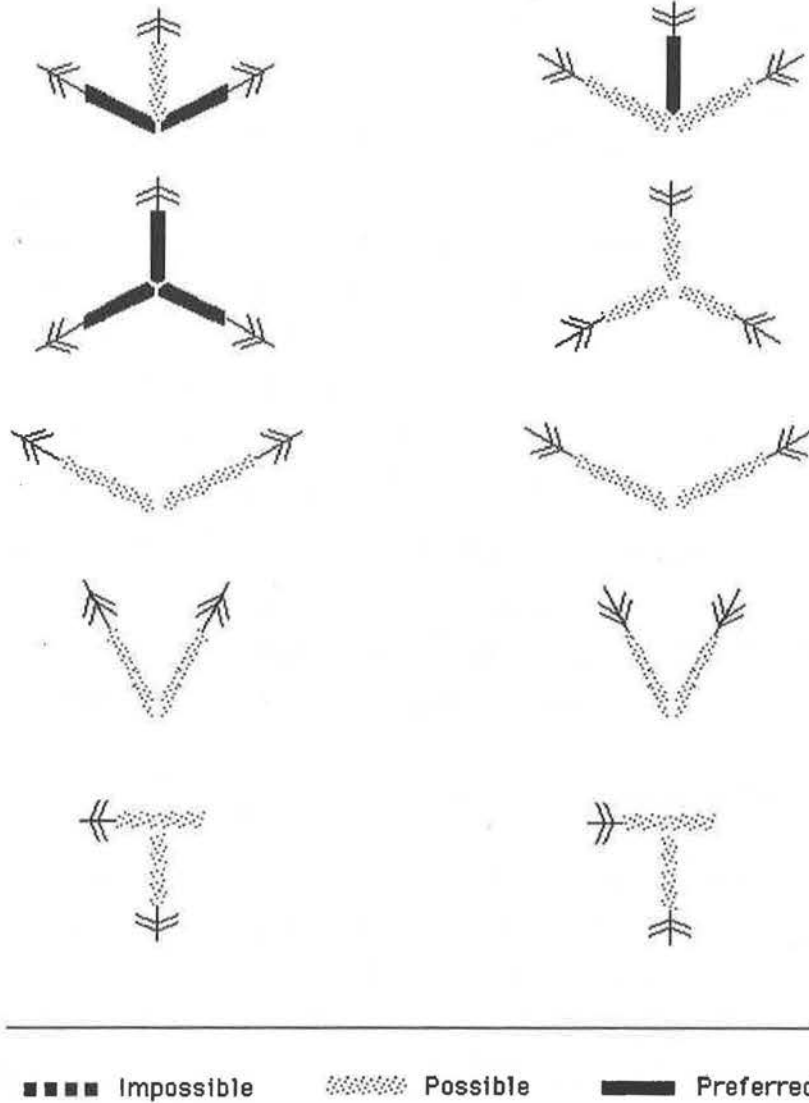


Figure 4.8: Initial slant-sign interpretations.

D. Slant Magnitude

If an arrow- or Y-junction obeys Perkins' laws (section 3.2.4), the values in its quantitative subsystem are assigned the corresponding slant magnitudes, and the values in the qualitative subsystem are set to 'preferred' to show that a definite interpretation has been made. Otherwise, the magnitude of the edges is set to a default value of zero, and the corresponding qualitative label is set to 'possible' so that it can be overridden by any definite interpretation.

4.3 The Rapid Recovery Process

Taken together, the external and internal constraints developed above go a long way towards specifying a mapping that allows a large amount of scene structure to be recovered in very little time. Minimal assumptions have been made about processing resources — it is assumed only that a mesh of relatively simple processors is available, and that the time required for local computation is less than that of data transmission to nearby locations. Consequently, these constraints are largely independent of the details of the underlying mechanism.

If the theory is to be complete, however, it must lead to a mapping that is uniquely specified. Several architectural parameters must therefore be specified. It must also be shown how the external and internal constraints can be smoothly combined into a rapid recovery process that is robust to small perturbations in the input.

4.3.1 Architectural Specifications

The constraints developed in the previous sections have the advantage that they are applicable to a range of possible processors. Because they depend on a few aspects of the processor, however, these aspects must be given a definite specification if the resultant mapping is to be unique. The choices made here are intended to be as general as possible, and to reflect what is known of the human visual system when the specification of particular parameters is unavoidable.

To begin with, the processing elements are assumed to be finite-state, making it necessary to convert continuous quantities such as two-dimensional orientation and slant magnitude into discrete form. Spatial location must be represented with a high degree of precision, reflecting the high acuity possible even at early stages in human vision (see, e.g., [WB82]). Each cell

is therefore assumed to be able to represent location to within 1/16th of its own size.¹⁴ On the other hand, the measurement of line orientation in the early visual system is based on channels of a half-amplitude bandwidth of about 10–20° [TG79], and so is much less precise. Consequently, orientation measurements are quantized to intervals of 10°.

The estimates of slant magnitude must also be quantized. Like two-dimensional orientation, these are given a relatively coarse-grained representation, with magnitude quantized to intervals of 20°, centered around values of 0°, 20°, 40°, 60°, and 80°.

Another issue is the way in which the zones can be arranged over the image. Three main types of regular tessellation are possible: rectangular, triangular, and hexagonal. The particular choice does not greatly matter when processing does not involve coordinate-dependent quantities, but this must be made definite for purposes of analysis. In order to simplify the implementation as much as possible, it is assumed that all zones have the same shape and size, and that they form a rectangular lattice over the image.

The coordination of communication between zones must also be specified. Processing over each zone is carried out by a separate processor or group of processors, and communication between these processors may proceed either synchronously (coordinated by a global clock) or asynchronously. Since the process acts via an irreversible priority override mechanism, and since the available propagation paths are constant,¹⁵ precise temporal coordination of operation is not important. Consequently, the issue of synchronous communication has little impact on the performance of the process. The major difference between the two types of communication is therefore in the ease of implementation and analysis. In what follows, synchronous communication is assumed.

Finally, an appropriate size must be chosen for the zone themselves. This depends in part on the absolute number of available processors, or more precisely, on the ratio of processors to the size of the input. It is assumed here that each zone can be made small enough to contain at most three lines (i.e., enough for a single junction). Beyond these requirements, the exact size of the zones is unimportant for present purposes — since processing speed is dominated by transmission time (section 2.5.2), changing the size of the zones only leads to

¹⁴Since each cell is later assumed to correspond to a visual area of roughly 10 min arc (section 5.3), this yields an precision of less than 1 min arc, roughly comparable to the limits of human visual acuity [WB82].

¹⁵State-dependent constraints are similar, the only difference being that a delay is introduced by the requirement that a definite set of labels be assigned to the critical variables.

a rescaling of the time course of the process.¹⁶

4.3.2 Robustness

The assumption of rectangularity carries with it an obligation to protect the process from the instabilities that result when lines in the image are nearly parallel or are nearly at right angles to each other. For arrow- and Y-junctions, techniques similar to those of section 3.3.3 can be applied in a straightforward fashion, at least locally. In particular, an arrow- or Y-junction containing a 90° angle is treated as if the angle were slightly larger.

Since a global broadcast of the reassigned angles is not feasible using a mesh architecture, ambiguous L-junctions cannot be immediately resolved. They are consequently treated here as junctions containing constraints common to both acute and obtuse L-junctions (see, e.g., [Mal87]). One such constraint is that at least one edge must be nonconvex (see figure 4.2).

The sensitivity of slant magnitude estimation can be reduced by a few additional measures. For junctions in clear violation of Perkins' laws (section 3.2.4), edges are given no initial preferred slant magnitude (i.e., the values in the qualitative system are set to 'possible'). Constraints are also weakened so that neighboring estimates are acceptable only if they are within adjacent ranges. Finally, to limit the accumulation of errors that would result if estimates of slant magnitude were propagated via L-junctions, estimates are taken only from direct sources (i.e., at the arrow- and Y-junctions), with values propagated only as far as the next junction.

4.3.3 Basic Operation

Given the additional refinements of the sections 4.3.1 and 4.3.2, the process is completely specified. Since many of the constraints apply to the generation of the interpretation, and not simply its final form, the image-to-scene mapping cannot be given a closed-form description. Instead, the interpretation of a given line drawing can only be obtained by carrying out the process itself.

The detailed operation of the recovery process is discussed in chapter 5, where an algo-

¹⁶Note that the absolute scale is important for any real system, leading to a preference for cell sizes that are as large as possible. Thus, the absolute size of a cell involves a time-space trade-off (cf. section 5.1.3): a larger number of smaller, simpler cells increases computational simplicity, while a smaller number of larger, more complex cells reduces transmission time (when internal transmission is not a factor). The choice of appropriate size is likely to be based on some compromise between these two sets of conflicting requirements.

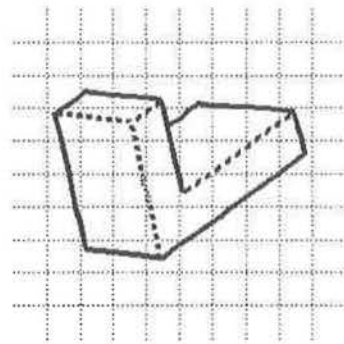
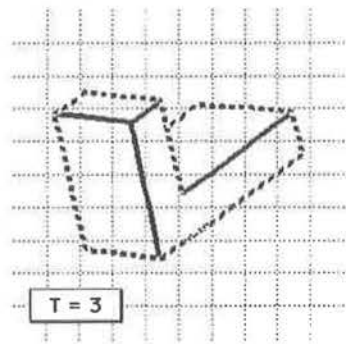
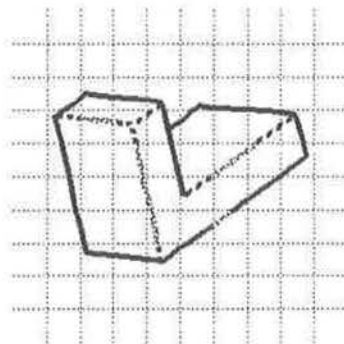
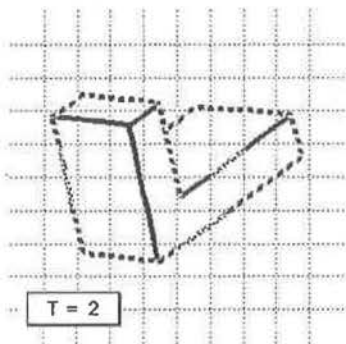
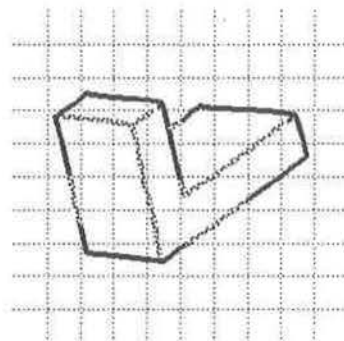
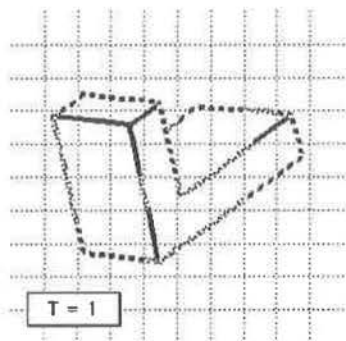
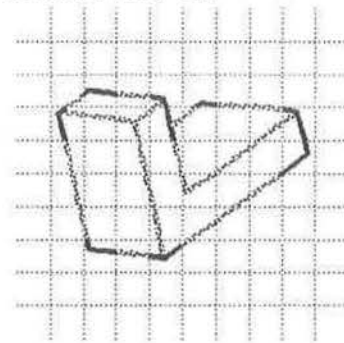
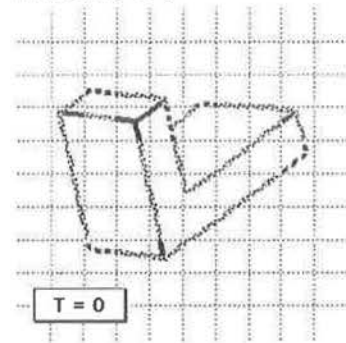
rithm is developed that embodies all the relevant external and internal constraints. However, the recovery process itself is slightly more abstract than this, since it is completely specified without the additional details of the algorithm. The basic elements of its operation, taking place in each zone concurrently, are as follows:

1. Initial measurements are made of the termination locations and the orientations of the line segments within the zone. Terminations include not only true endpoints of the lines, but also crossings of the zone boundaries. The locations of these terminations are represented with high precision (1/16 of the zone size). Orientation measurements are quantized in units of 10° .
2. The type of junction present (if any) is the zone is established, and the angles between its lines determined.
3. Initial interpretations are assigned to all variables in all substreams. If the zone contains one or more disconnected lines, all values are assigned 'possible'. If it contains a junction, the lines are labelled according to the rules described in section 4.2.2. This is done separately for the values and complementary values in each of the streams.
4. Values are propagated along connecting lines to neighboring zones via the priority mechanism described in section 4.2.2. This is done in tandem for both subsystems in all streams. Since communication is only possible between zones that are immediate neighbors (section 4.2.1), this leads to a percolation of information along the lines at a constant speed. Propagation of labels proceeds by assigning 'impossible' states to eliminate inconsistent interpretations, and by assigning 'preferred' labels to select a preferred subset of the remaining possibilities.
5. Simultaneous with this "intra-dimensional" process, an "inter-dimensional" propagation is also occurring, transmitting information from zones that contain a variable with a definite interpretation. This transmission applies only to zones at the same location in the image, and follows the rules given in figure 4.2.
6. The transmission of information along lines and between dimensions continues until the time limit is reached. Inconsistent interpretations are identified by the assignment of 'impossible' to an edge in both subsystems. Ambiguous interpretations are identified by the assignment of 'possible' in both subsystems. Of the remaining interpretations, those deemed to be most likely are distinguished by the 'preferred' state.

An example of this process is shown in figure 4.9, which illustrates how the initial convexity estimates assigned to a drawing evolve into a more complete interpretation.

Convexity (+)

Nonconvexity (o)



○ Magnitude ■ Impossible ~~■~~ Possible — Preferred
Narrow gray lines mark cell boundaries

Figure 4.9: Example of the rapid recovery process.

Chapter 5

Algorithm and Implementation

The computational analysis of chapter 4 has yielded a set of external constraints on the “static” associations between image and recovered scene, and a set of internal constraints on the “dynamic” aspects of the recovery process itself. These specify a unique image-to-scene mapping, and provide some general limitations on the transformations that are to be used. What is now required is to show that these constraints can be incorporated into a complete, well-defined system. In particular, the process must be decomposed to the point where it can be carried out via the operations available on a device having the processing limitations assumed in the computational analysis (section 2.4).

The analysis here is based on a device called the *cellular processor*. This is a type of cellular automaton (section 5.1.2) formed by partitioning a dense mesh of processors into a relatively sparse set of disjoint “cells”, each of which is assigned a simple processing element to carry out the local interpretations. It is shown that the basic operations of this mechanism can be implemented on a mesh of simple finite-state processors. The algorithm itself is then developed via a simple program based on these basic operations. The general properties of this mechanism are shown to be compatible with what is known of primate cortical structure, and a tentative suggestion put forward regarding the way in which it might be implemented in human visual cortex.

5.1 The Cellular Processor

If it is to be effective, a rapid recovery process must be based on estimates made over regions of the image that are contiguous and compact, i.e., over *zones* (section 4.2.1). This introduces two different spatial scales into the recovery process: (i) a fine-grained scale that supports the

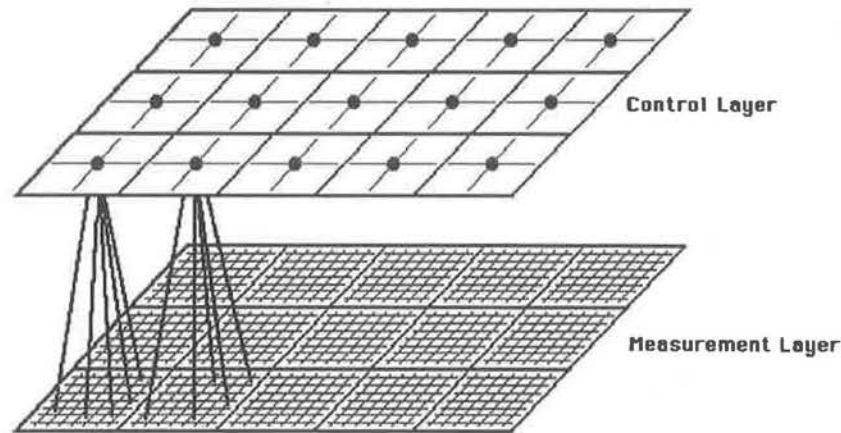


Figure 5.1: Cellular processor architecture.

high resolution of the input and output representations, and (ii) a coarser-grained scale based on the size of the zone. A useful mechanism to handle this situation is the *cellular processor*. This is a device consisting of two spatiotopic meshes: (i) a dense mesh of *measurement elements* that determine basic image properties (e.g., color, contrast, and orientation), and a sparser mesh of more complex *control elements* that carry out the local interpretations (figure 5.1).

5.1.1 Basic aspects

The cellular processor allows algorithmic analysis to be carried out in a straightforward way, with issues of measurement and control separated as much as possible. Each measurement element (ME) can be loosely identified with a mechanism that measures some template property, such as the color or orientation of lines. These MEs are assumed to have a small set of possible output values that are determined entirely by the contents of a spatially-limited neighborhood around the corresponding point in the image. As such, they have no internal states and operate independently of each other.

The spatiotopic order of the set of inputs is assumed to be maintained in the set of ME outputs, so that the array of MEs and the array of their outputs can both be referred to as the "measurement layer", the distinction between the MEs and their outputs being clear from context. Adjacent elements in this layer may or may not have overlapping input regions. It is assumed that the density of MEs is sufficiently high that that no information in the image is lost.¹

¹This layer has some interesting similarities with the dense set of localized filters found in the striate cortex

To carry out more complex operations, the measurement layer is partitioned into a number of compact, contiguous sections (or “cells”),² with the outputs in each cell assigned to a control element (CE) at the corresponding location in a higher-level “control layer”. Each CE is assumed to be sufficiently complex that it can respond to all possible combinations of outputs in its cell. Towards this end, each CE is given a small finite memory to hold intermediate quantities derived from the ME outputs (e.g., the number of line segments it contains, their location, and the areas of any region bounded by them). Note that these quantities need not be determinate — in situations where space or time is extremely limited, or where there is some inherent uncertainty in the measurements themselves, statistical quantities may be preferred (see, e.g., [Ros86]).

It also is assumed that each CE can control at least some of its MEs via backprojections that override the ME output.³ In addition, each CE is assumed to have a small set of operations that it can perform on its memory locations and on its MEs. These operations form the basis of the local processing carried out by the processor.

In contrast to the isolated elements of the measurement layer, elements in the control layer are able to interact with their nearest neighbors, having access to at least some aspects of their neighbor’s current state. This adds a degree of “lateral” control to the “bottom-up” and “top-down” strategies generally employed in visual processing.

5.1.2 Cellular Processors as Cellular Automata

Since each ME is an isolated mechanism performing a single operation, all interesting aspects of the recovery process are carried out by the processors in the control layer. Consequently, the evolution of a cellular processor can be completely described by a rule that maps the current state of each control element onto a new state, the new value being determined by (i) the outputs of the MEs within its cell, (ii) the contents of its memories, and (iii) the states of its immediate neighbors. Since processing must be indifferent to the absolute spatial coordinates in the image, this mapping must be spatially uniform. Furthermore, the process is assumed to operate via synchronous communication between all zones (section

of primates (see section 5.3).

²The meaning of the term ‘cell’ corresponds to that of ‘zone’, but at the level of architecture rather than that of image.

³This can be accomplished by special internal memories, each capable of overriding the outputs of one particular ME. In this formulation, the output of the CE can be expressed either as the set of ME outputs or as the set of CE memory states.

4.3.1). Described in this way, a cellular processor (or more precisely, its control layer) is seen to be a special type of *cellular automaton*.

Cellular automata are discrete deterministic systems formed by a d -dimensional grid of identical processors operating according to a fixed local law. More precisely, a cellular automaton (CA) can be defined as a quadruple [Kar90]

$$A = (d, S, N, f), \quad (5.1)$$

where d is a positive integer describing the dimension of A , S is a finite set of states, N is a set of n neighborhood vectors (each of the form $\vec{x} = (x_1, \dots, x_d)$), and f is the local transition function from S^n to S . The cells of A are arranged along an infinite d -dimensional grid, their positions indexed by elements Z^d , the d -dimensional space of integers.

Cellular automata were developed originally by Ulam and von Neumann as tractable approximations of highly nonlinear differential equations in biological systems (see [TM90]). But they are also interesting in their own right, since local rules can lead to a variety of complex, spatially-extended structures (see, e.g., [CHY90, Smi90, TM90]). Cellular automata have been used for simple image operations, including thresholding, pointwise arithmetic on image pairs, and convolution (see, e.g., [Gol69, PDL⁺79, Ros83]). Other operations include the shrinking and expansion of elements in the image, and the formation of their convex hull [PDL⁺79]. Indeed, it is likely that CAs can do considerably more than this, since given the appropriate transition functions and initial configurations, they are capable of universal computation, i.e., computing any function that can be computed by a Turing machine (see, e.g., [CHY90]).

In order to conform with the general constraints of the recovery process, a two-dimensional grid is used, and the neighborhood set N is the set of cells at most a unit distance away in the horizontal or vertical direction.⁴ Thus, the neighborhood is composed of nine cells: the cell itself, and a layer formed by overlapping 3×1 arrays of cells immediately to the top, bottom, right, and left. Consequently, the transition function f is described by a mapping $S^9 \rightarrow S$ that associates each possible pattern of neighborhood states to the new state of the center cell.

⁴A rectangular tessellation is not necessary for cellular automata that operate on images — several applications (e.g., [Gol69]) are based on a hexagonal array. Indeed, any CA with an arbitrary neighborhood N is equivalent in its computing power to one with a von Neumann neighborhood, i.e., one with neighbors to the top, bottom, left, and right (see [PDL⁺79]).

5.1.3 Programming

A. Basic Considerations

Viewing the control layer of a cellular processor as a cellular automaton, its programming reduces to the design of an appropriate transition function and selection of an appropriate initial configuration of values. There are, however, three important constraints particular to its operation.

First, to rule out the necessity for any kind of higher-level global mechanism, the initial value of each CE must be determined entirely by the MEs within its corresponding cell. This means that the initial configuration of values must be in spatial register with the input image, thereby prohibiting the use of the special-purpose initial patterns or auxiliary elements often used in general CA design. Similarly, the final configuration also is required to be in register with the image, since the output is required to be a spatiotopic map. This rules out algorithms that deform the spatial organization found in the input, such as the shrinking process used to count the number of items in an image [Gol69]. Finally, the operation of the processor itself must be in-place, i.e., the memory in inactive cells cannot be used as scratch space for intermediate calculations. The use of scratch space is a viable option when the initial configurations are such that known subsets of the grid can be guaranteed to remain inactive (see, e.g., [Arb87, ch. 7]). However, this condition cannot in general be met when an arbitrary set of input images (and therefore initial configurations) is possible.

The power of a cellular architecture cannot therefore be harnessed in the manner used for many classes of general CA problems, viz., by designing an appropriate initial configuration. Instead, the appropriate information must be stored locally in each cell. This can be done by increasing the number of states in S (i.e., increasing the number of states in each control element). Increasing power in this way also allows the transition function to have a more natural structure, simplifying the design and analysis of the system's behavior [Arb87, ch. 7].

At the lowest possible level, therefore, the programming of a cellular processor reduces to the selection of a set of states for each CE, together with a transition function that operates on these states. But to help ensure that the processor respects the constraints described above, it is convenient to program at the slightly higher level of simple operations on particular properties accessible by the CE. Once such a set of operations has been specified, any particular recovery process can then be specified by the appropriate concatenation of

operations. This is effectively a general mechanism for the “abstract programming” of parallel processes, with the resultant program loaded into each of the CEs, where it acts somewhat like parallelized version of a visual routine [Ull84].

B. Elementary Structures and Operations

The data structures to be used in programming the control elements are straightforward: the values of the MEs in the corresponding cell, the contents of the internal CE memories, and the accessible properties of the adjacent CEs. These are all simple scalars, with only a small set of possible values. As such, they can be handled in a uniform way.

More latitude exists in the choice of elementary operations. There is in some sense a “natural” set of basic operations — if too few exist, it may not be possible to carry out all the intra-cell operations within a single time step; if too many exist, they merely add to the space required by the CE. The elementary operations chosen here are simple forms of data input, output, and transformation:

- 1. Input of information from MEs to memory elements.** Connectivity within a cell is assumed to be high enough to allow a CE to establish direct access from any ME to any of its internal memory elements.
- 2. Output of information from memory elements to MEs.** Connectivity also is assumed high enough to allow a CE to establish backprojections from any of its internal memory to at least some of its MEs. The interpretation output by the processor takes the form of values of these latter MEs (or equivalently, of the corresponding memory elements that override them).
- 3. Simple operations on information in memory elements.** It is assumed that each CE can add, subtract, multiply, and perform integer division on the contents of the memory elements. It also is assumed that a two numbers can be compared to determine the higher value. Inputs and outputs for these operations are always taken from the memory elements; transfer of contents between memory elements is simply a special case where no operation is performed.

In addition, each CE is assumed to have an input from higher levels that provides a simple control on its operation. Depending on the value of this signal, the CE either resets its memories to some default state, begins/continues its operation, or halts its operation.

C. Combining Basic Operations

The cellular processor is programmed by creating an appropriate transition function and set of states from the elementary structures and operations described above. This can be done most simply by concatenating elementary instructions together into a sequence, an operation which corresponds to the composition of the corresponding transition functions. Both simple and compound operations can be concatenated into new compound operations. Note that the resultant transition need not be carried out in a sequence of separate transitions — it can be “fused” into a more complex function that can be carried out in one step.

The replacement of a sequence of simple operations by a single transition corresponds to the use of a lookup table (cf. section 7). In this sense the process is consistent with the loading-in of a complete object model based on its features in the image (e.g. [PE90]). However, the approach here involves items of a smaller “local models” composed entirely of locally-definable properties. Note that the issue here centers around the advantages of a larger sequence of simple transitions as opposed to a smaller sequence of more complex transitions — an instance of the basic time-space tradeoff found in more general models of computing (see, e.g., [Har87]).

Operations can similarly be combined via the “if-then” conditional construct, the result simply formed from the two alternative functions. The loop construct of conventional programming languages also is allowed, but only if the body of the loop is carried out a limited number of times. As used here, the loop is a simple programming convenience, which is “unrolled” in the actual implementation of the transition function. A loop controlled by a variable can be translated into several separate unrolled loops, which are then selected via conditional constructs. In a similar fashion, procedures can also be used to help specify the process, but each is to be treated as a macro that is replaced in the actual transition function by the set of instructions it contains. As such, procedures cannot call each other recursively.

Finally, the program given to the cellular processor does not need an explicit ‘halt’ command, since it is assumed that the processor is suspended (as well as started) by an explicit command from higher levels.

5.2 Algorithm for Rapid Recovery

Given the set of operations available to the cellular processor (section 5.1.3), it remains to use these as the basis of an algorithm capable of carrying out the recovery process sketched in section 4.3. Although the constraints on the recovery process and on the cellular processor are not sufficient to specify a unique algorithm, this is not important for the present purpose, which simply is to show that such an algorithm can exist. The algorithm used here can be summarized as follows:

For each control element:

1. Obtain from the measurement elements the locations of all line terminations and the orientations of all line segments within the cell. Terminations include not only true endpoints of the lines, but also points at which the zone boundaries are crossed. As required by the specifications of section 4.3.1, orientation measurements are quantized in units of 10° .
2. Determine the type of junction(if any) that is present, and make explicit several of its properties, such as the values of the angles involved.
3. Assign initial labels to the lines according to the rules described in section 4.2.2.
4. For each subsystem of each stream, repeat the following:
 - a. Read the relevant values from any neighboring CE that shares one of the lines. Update the current values via the priority mechanism described in section 4.2.2.
 - b. Read the relevant values from those streams containing a variable with a definite interpretation, and update the current values according to the rules given in figure 4.2. Since this transmission applies only to zones at the same location in the image, only the internal memories of the CE are involved.
 - c. Apply the intra-line constraints according to the rules given in figure 4.2.2. These eliminate any local inconsistencies that may have arisen in the new set of values.
5. Stop iteration when the time limit is reached. The final interpretations are determined from the assignment of the 'possible', 'preferred', and 'impossible' labels in each subsystem, according to the rules given in section 4.2.2.

The following sections describe in greater detail how these operations are carried out by the cellular processor.

5.2.1 Determination of Basic Image Properties

The measurement elements in each cell describe the image basic properties available to the control element. These include the locations of the line terminations and the orientations of the line segments in the area subtended by the cell. There are a variety of ways this can be done. Here, each ME is assumed to signal the existence of a line centered at the corresponding location in the image array. Line segments of different orientation, horizontal length, and vertical length are represented by different sets of MEs, each signalling the presence or absence of its particular type of segment by a simple binary output. As required by the architecture specifications given in section 4.3.1, these elements represent length and position to a high degree of precision, with orientation represented only coarsely.

These outputs contain a complete (in fact, redundant) description of all line segments in the cell, and can therefore support the determination of all the image properties needed by the control element. Three properties are of particular interest, all of which are represented via a bank of finite-state memory elements:

Number of lines in the cell: This can be determined from a count of the number of ME outputs that are active. A maximum of three is assumed (section 4.3.1).

The endpoints of each segment: These are calculated for each segment from the knowledge of the relevant center point and the horizontal and vertical lengths. No more than six endpoints need to be stored.

The orientation of each segment: These can be taken directly from the orientation label of the appropriate ME. No more than three values need to be stored.

5.2.2 Determination of Junction Properties

The next step is to obtain those properties of the junction useful for subsequent interpretation. These only need to be calculated once, their values then stored in an appropriate bank of memory elements. Five particular sets of properties are used here: junction position, junction angles, junction type, junction rectangularity, and an auxiliary set of line descriptions (two for each line) used for the interpretation of contiguity. As required by the recovery process, all quantities are finite.

A. Junction Position

Junctions are detected simply by finding the intersection point of the lines in the cell. It is assumed that each cell is sufficiently small that at most one junction (and therefore one intersection point) can exist within the area it subtends. The existence of the intersection point is determined by testing for the identity of the endpoints. For the case of T-junctions, a slightly different procedure is used, based on a unique zero distance from an endpoint of one line segment to a different line segment.

If no intersection point is found, no junction exists within the cell. Note that this is possible even if the junction contains several lines, since these lines may be noncontacting. If an intersection point is found, its location is stored into an appropriate memory element.

B. Junction Angles

The angle θ_{ij} between each pair of connected lines \vec{a}_i and \vec{a}_j is simply the absolute value of the difference of the two orientations. The only real difficulty here is to determine how the lines are connected – as seen from figure 5.2, each pair of lines can be combined in two different ways, corresponding to acute and obtuse forms.

These can be distinguished via the dot product of the two lines, defined to be (see, e.g., [Tho72])

$$\cos(\theta_{ij}) = (\vec{a}_i \cdot \vec{a}_j) / |\vec{a}_i| |\vec{a}_j|.$$

The disambiguation of acute and obtuse junctions can be based on the sign of the cosine: positive for acute angles, negative for obtuse. If the difference between two line orientations actually corresponds to angle θ_{ij} , it will therefore have a value between $0^\circ - 90^\circ$ for pairs with a positive dot product, and between $90^\circ - 180^\circ$ for a negative dot product. If these conditions do not hold, the angle must be 180° minus this value (figure 5.2).

Since only the sign of the dot product is important, division by the magnitudes need not be performed, and so can be readily carried out by the control element. Note also that the dot product is a true scalar quantity (see, e.g., [Tho72]), so that no artifacts are introduced by the selection of any particular co-ordinate system. Among other things, this takes care of any problems introduced by the discontinuity in orientations at 180° . It also means that orientation can be taken with reference to any co-ordinate system, the only requirement being that the same system is used locally for any junction.

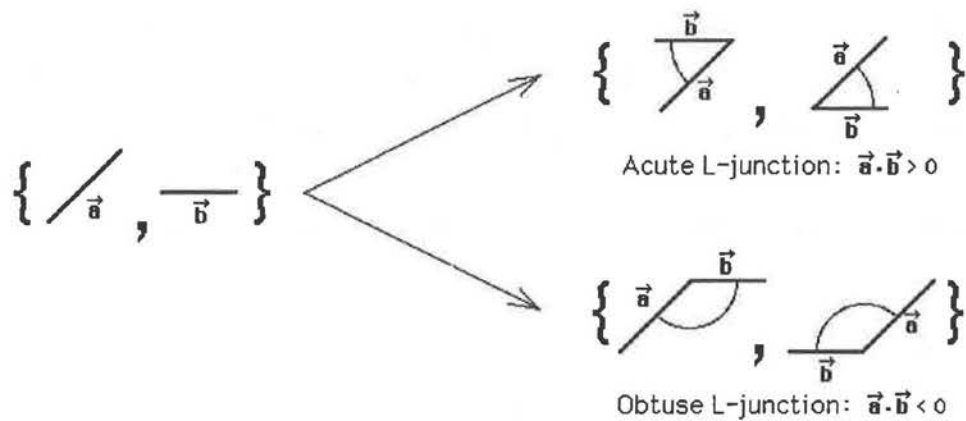


Figure 5.2: Calculation of orientation differences.

C. Junction Type

Junctions are classified by a two-stage process. The initial classification based on their arity, i.e., the number of lines existing at the common intersection point. This value can be obtained simply by counting the lines that have an endpoint identical with the intersection point. Junctions are then marked as follows:

No junction: No intersection point exists

T-junction: One endpoint contacts the intersection point

L-junction: Two endpoints contact the intersection point

Arrow-, Y-junction: Three endpoints contact the intersection point

Further disambiguation can be based on the values of the angles between the lines:

L-junction (acute): angle is less than 90°

L-junction (obtuse): angle is greater than 90°

Arrow-junction: angles sum to less than 360°

Y-junction: angles sum to exactly 360°

Complications can arise when junction angles are nearly orthogonal, since the uncertainty in the sign of the dot product makes it difficult to discriminate acute L-junctions from obtuse ones in a reliable way. It also becomes difficult to distinguish arrow- from Y-junctions if two such angles are present in a junction (i.e., one of the line pairs is almost collinear with another). Various techniques can lend robustness to the recovery process under these conditions

(cf. section 3.3.3), but in the interests of simplicity, only a few are used here (section 4.3.2). In particular, L-junctions with angles determined to be 90° (based on the quantized estimates in memory) are treated as a separate type of L-junction that has the constraints common to both obtuse and acute L-junctions. As for the other kinds of L-junctions, right-angled L-junctions can be determined by a simple test of junction angles.

D. Junction Rectangularity

An important basis for the recovery of slant magnitude is the assumption that the junction corresponds to a rectangular corner in the scene, with slant magnitudes assigned only to those edges belonging to a junction obeying Perkins' laws (section 4.3.2). Consequently, it is important to indicate whether or not a junction can correspond to such a corner. This can be done via a simple test based on the angles and type of the junction:

L-junction: no

T-junction: no

Arrow-junction: one angle $> 90^\circ$ and two angles $< 90^\circ$

Y-junction: three angles $> 90^\circ$

Rectangularity is flagged simply by assigning a zero value to all angles in junctions that do not pass this test.

In the interests of robustness, this procedure must be extended to handle right angles as well. Note that since two angles of 90° in a junction form a T-junction, only one right angle is allowed in any arrow- or Y-junction, allowing the extension to be done in a simple way:

Arrow-junction: one angle $\geq 90^\circ$ and two angles $\leq 90^\circ$.

Y-junction: three angles $\geq 90^\circ$.

E. Contiguity Lines

In contrast to the other interpreted properties, the contiguity of lines with their flanking regions requires the assignment of two values per line, one for each side (section 3.2.1). Contiguity is therefore represented here by a pair of contiguity lines ("c-lines") obtained by

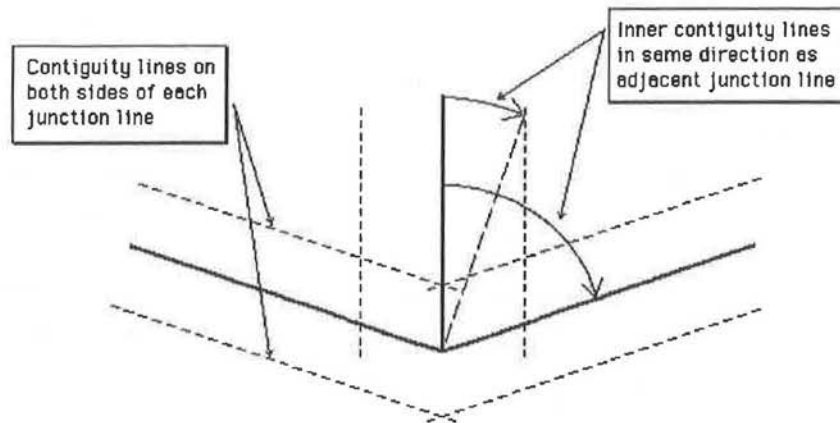


Figure 5.3: Determination of contiguity relations.

offsetting the original “parent” line a few pixels on either side (figure 5.3). The value of each c-line indicates whether its corresponding region is contiguous with the parent line in the junction. In all respects, c-lines are treated as regular junction lines, taking on the states of ‘possible’, ‘preferred’, and ‘impossible’.

Because several constraints apply to c-lines that share a common region, it is useful to have a record of which c-lines are connected to each other inside the junction. Since almost all connected c-lines are the inside edges of adjacent lines (cf. section 3.2.1), the test for connectedness reduces almost completely to a search for these inside edges.⁵ The problem, then, is to determine which c-lines are on the “inside”, i.e., which c-line faces the junction line opposite its parent (figure 5.3).

A simple way to solve this problem is based on the cross product, which for lines \vec{a} and \vec{b} is defined as the determinant (see, e.g., [Tho72])

$$\vec{a} \times \vec{b} = \begin{vmatrix} \mathbf{e}_x & \mathbf{e}_y & \mathbf{e}_z \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = |\vec{a}||\vec{b}| \sin(\theta_{ab})\mathbf{e}_z,$$

where the \mathbf{e}_i are unit vectors in the x, y, and z directions. In the case of two dimensions, the cross product describes the area swept out by the two vectors, its sign depending on the sense of the rotation required to align \vec{a} with \vec{b} .

Consider first one of the junction lines. The cross product of this line with an adjacent junction line can be readily determined by the control element, the sign of this quantity describing the sense (either clockwise or counterclockwise) in which this line must be rotated

⁵The only exception is for the outer edges of the arrow-junction, and these can be handled straightforwardly.

to line up with the adjacent line. Consider now the associated c-lines, together with the lines formed by joining their outer points to the junction intersection. Only the c-line on the inside edge (i.e., the c-line facing the adjacent junction line) can give rise to a line in the same rotational direction as that of the adjacent junction line (figure 5.3). Repeating the same procedure for the c-lines of the adjacent line then yields the pair of inside edges.

The cross product is a quantity independent of the co-ordinate system (see, e.g., [Tho72]). It therefore allows processing to be unaffected by the particular co-ordinates used in representing the orientation of the lines.

5.2.3 Initial Assignment of Interpretations

Once the basic image and junction properties have been determined, the next step is to assign the initial interpretations to the variables in each of the four streams. The states of these variables are held in eight separate banks of memory elements, one element per subsystem. Only three possible values can be attached to any complementary variable, and only five are possible for slant magnitude (section 4.3.1). Consequently, these memory elements only need to take on a few possible states.

The assignment operation itself is a straightforward procedure that sets the values of the relevant memory elements, the particular choice of values depending only on the junction type (section 4.3). This can be carried out by using a set of conditional statements. Together with the values describing the structure of the junctions, the resulting set of CE memory states provides the initial configuration for the iterative part of the interpretation process.

5.2.4 Propagation of Interpretations

Given an array of initial interpretations, it remains to transmit these values to neighboring locations and streams. As discussed in section 4.2.2, this is done by an iterative process that at each iteration replaces values of low priority with values of higher priority, i.e., 'preferred' replacing 'possible', and 'impossible' replacing 'possible' and 'preferred'. The constraints at each junction guide the local transmission of these values, resulting in "waves" of higher-priority states circulating around the lines in the image. The propagation of these waves continues until an equilibrium state is reached or until the process is timed out.

The way the propagation process is carried out is much the same in all subsystems: all

variables sharing constraints with the "target" variable are accessed, and if any of these has a value of higher priority than the value of the target variable, the memory element is set to this value. This can be done even for the case of state-dependent constraints (section 4.2.2, since only an additional conditional construct is required to put the appropriate constraints into effect.

Information access occurs via four different avenues: neighboring cells, neighboring streams, intra-junction constraints, and intra-line constraints (section 4.3). Since transmission is based on a simple priority mechanism, the order of the access operations within each stream is unimportant. This allows the propagation process to be carried out by a relatively simple set of operations.

A. Updates from Neighboring Cells

The updating of values from sources outside the cell can be done concurrently for each line segment. To access the appropriate values from neighboring locations, first determine which neighbors contain a continuation of the relevant line segment. This is done by reading the set of endpoint locations stored in each neighboring control element and testing for equality against the endpoints of the local line segment. Since a line crossing a cell boundary is divided into two segments that terminate at the same point (i.e., the boundary), this test succeeds if and only if the segment continues into the neighboring cell.

For each cell containing a continuation of the segment, access the relevant set of memory elements and compare their values against those of the current cell. Since continuations are required to have the same values, updating follows the rules for bijective constraints described in section 4.2.2.

B. Updates from Neighboring Streams

Just as information is transmitted from neighboring locations, it also is transmitted from neighboring streams. The only difference between the two situations is that whereas inter-cell transmission is based simply on priority, inter-stream transmission usually has an additional dependence on the particular type of junction and on the particular line in that junction (section 4.2.2). This dependence is fixed for each junction type, with updating carried out by a set of conditional assignments between the appropriate variables. Since this updating is based on simple priority, the order in which streams are evaluated is unimportant.

C. Updates from Intra-junction Constraints

After assigning new values to the lines based on sources external to the junction, the next step is to impose the set of local constraints on the lines of the junction itself. Updating follows the rules described in section 4.2.2, with the particular constraints depending on junction type. Consequently, it can be carried out by a set of conditional constructs. Since the final result depends only on priority of the values involved, the order of evaluation of the lines is unimportant, and can even be done in parallel.

D. Updates from Complementary Subsystems

A final path of information transmission originates in the constraint between the values in complementary subsystems: if any value in a subsystem has been set to 'impossible', the value of its complement is upgraded to 'possible' (see section 4.2.2). Since this constraint holds for all lines at all times, it can be carried out via a conditional assignment incorporated into the assignment mechanisms used in the other access paths.

5.2.5 Final Assignment of Results

After the propagation of values has been halted, a final "postprocessing" phase can be used to transform the states of the sets of complementary variables into a more "standard" representation that expresses the two definite interpretations, the inconsistent interpretation, and the ambiguous interpretation. The rules of this transformation are given in section 4.2.2. The transformation itself can be carried out straightforwardly on the relevant memory elements since only a simple remapping of values is involved.

5.3 Neural Implementation

The final requirement of a computational analysis is that it demonstrate the existence of a physical system capable of carrying out the process — in particular, one compatible with the neural mechanisms believed to underlie human vision [Mar82]. But the detailed knowledge about the neurophysiology of vision is limited mostly to processes that measure simple image properties such as contrast and orientation (see, e.g., [Bis84, Sch86]). An detailed analysis of the neural implementation of rapid recovery is therefore not possible at the present time.

However, the cellular processor developed in section 5.1 is largely compatible with what is known about the anatomy and physiology of primate visual systems. The main stream believed to be involved in form vision begins with retinal cells that measure simple properties such as the contrasts and motions of luminance gradients in the image. The outputs of these cells extend to the lateral geniculate nucleus, and the geniculate cells in turn extend to the visual cortex, with a spatiotopic order maintained at all points along the way (e.g., [Bis84]). The visual cortex serves as the location where the outputs of this stream are brought together with those of the other streams. It contains cells sensitive to a variety of simple properties, including contrast, color, and line orientation (e.g., [dYvE88]). These form a dense spatiotopic map, with each point in the array containing a description of the various image properties at the corresponding point in the input image. As such, this map is an array similar in many respects to the measurement layer of the cellular processor.

The spatiotopic ordering of cells in the primate visual cortex is not quite point-to-point. Rather, it is "patch-to-patch", with each set of ganglion cells in a retinal patch projecting to a separate module (or "hypercolumn") in the visual cortex [HW74, Hub81, Bis84]. Each hypercolumn is a vertical section of the cortex with an area of approximately $1\text{mm} \times 1\text{mm}$; the primate visual cortex is thought to have about 2000 such columns, each containing at least several thousand cells [Hub81]. The corresponding patch of the visual field increases with eccentricity from the fovea, but around the fovea itself it has dimensions of about $10'$ arc (i.e., $1/6^\circ$) [Hub81]. All the measurements made over each patch are brought together in the corresponding hypercolumn, allowing it to completely analyze its section of the visual field. A similarity with the control layer of the cellular processor is evident. This similarity is reinforced by the observation that most connections between cells are vertical ones within the column itself, lateral connections to other areas being much sparser and shorter, often with lengths of only 1–2 mm (i.e., extending only to nearest neighbors) [Hub81].

If hypercolumns can be identified with the control elements of the cellular processor, it would imply that hypercolumn operation is more sophisticated than generally believed. But such sophistication would not be implausible given the number of cells in each hypercolumn and the density of their internal connections. In this context it is important to note that hypercolumn organization is extremely common, being found in most parts of the cortex in virtually all mammalian species [GJM88]. Thus, it is not absolutely essential that rapid recovery is carried out in the hypercolumns of the visual cortex — the hypercolumns of the extra-striate visual areas (see, e.g., [MN87]) could also be used for this purpose.

Chapter 6

Tests of the Theory

The final stage of the analysis is to test the theory on actual line drawings of polyhedral objects. Two sets of issues are of interest here. The first is how well the recovery process handles various kinds of line drawings. The process is tested on drawings of objects that violate the underlying assumptions about scene structure, and on drawings that cannot be given a consistent global interpretation. It is shown that a substantial amount of three-dimensional structure can be recovered under a wide range of conditions.

The second set of issues concerns the ability of the theory to explain the recovery of three-dimensional structure at the preattentive level of human vision. It is shown that the theory can explain — at least in broad outline — how early visual processing can recover three-dimensional orientation from some kinds of line drawings, and why it cannot do so for others.

6.1 Performance on Line Drawings

To examine the power and the limitations of the recovery process, it is tested on a range of line drawings, including those in which all underlying assumptions are obeyed as well as those in which various assumptions are violated. Although the resulting interpretations are not perfect indicators of the overall effectiveness of the process, they do provide an idea of the relative ease or difficulty of interpreting the various kinds of line drawings.

Since the speed of the process is determined primarily by the speed of information transmission, the absolute size of the line drawing has virtually no influence apart from a rescaling

of the time course (section 2.5.2).¹ The effects of size are therefore eliminated by rescaling all drawings so that their average line length is the same. For the drawings considered here, the average line length is set to 5 cell widths.

Transmission speed can be similarly factored out by measuring time in terms of the number of transitions between adjacent cells, or equivalently, by the number of iterations. This value is essentially a free parameter, which can have different values when recovery is used in different situations or for different purposes. However, in order to obtain an indication of the relative difficulty of recovery for various kinds of line drawings, it is useful to base comparison on one particular time limit.

As a representative value, the number of transitions is such that information is propagated along a distance of twice the average line length. This allows enough time (on average) for the estimates from each junction to be transmitted to their nearest neighbors, and for any updated values to be transmitted back. Since the average length is 5 cell widths, 10 transitions are used.

6.1.1 Rectangular Objects

When scenes contain only rectangular objects, all assumptions about the structural constraints (section 3.1.1) are true, giving the process the best chance to obtain a globally consistent interpretation of all scene-based properties. The corresponding line drawings therefore test the ability of the process to obtain such interpretations under ideal conditions.

i) Convex objects

The objects most amenable to rapid recovery are simple convex rectangular blocks (figure 6.1), since these not only obey all structural assumptions, but also obey the principle of maximum convexity that is used to select the initial set of interpretations (section 4.2.2). As figure 6.1 shows, almost all the three-dimensional structure has been recovered, with unambiguous preferred values assigned to all the lines in all four streams, and with almost all the alternatives ruled out as impossible.

A remnant of uncertainty remains in the center of the drawing, where the alternative convexities and slant signs are not yet completely ruled out. The propagation of the 'impossible'

¹Performance does change as the size of the entire object approaches the dimensions of a zone, since the assumption of no more than three lines per cell (section 4.3.1) can no longer be held. However, drawings here are assumed to be large enough that this is of no concern.

values from neighboring cells does, however, provide these areas with a definite interpretation after a few more iterations. This illustrates a common feature of the process — ambiguity is typically eliminated by proceeding from the outside of the drawing to the inside. This is largely due to the low ambiguity of the L-junctions, which are most often found on the outside border of the drawing.

The other area of uncertainty is the assignment of contiguity to the outer edges of the drawing. This is due to the inherent ambiguity of the line drawing itself, which can be interpreted as a block attached to various surfaces (floor, wall, ceiling) or as a block without any attachments at all. The recovery process has no means for preferring one over the other, and so the interpretation of these values remains ambiguous.

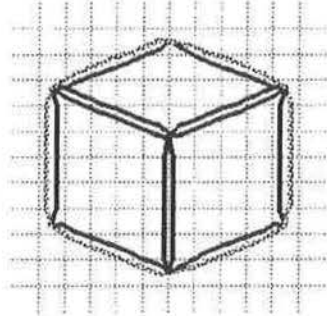
ii) Nonconvex objects

Nonconvex rectangular objects obey all structural constraints, but contain nonconvex corners that are initially assumed to be convex (section 4.2.2). As seen from figure 6.2, the initial assignment of an incorrect set of values to the nonconvex junction does not seriously affect the final interpretation. Contiguity is assigned unambiguously and correctly to almost all surfaces, with the exception of the outer edges, which — as for the case of the convex block — cannot be given an unambiguous interpretation. Note that the preference for contiguity of the edges of Y-junctions (section 4.2.2) has caused the lower edge to be given a ‘preferred’ value, although the opposite interpretation has not been definitely ruled out. The other streams similarly contain edges that either have a definite interpretation or involve preferred interpretations.

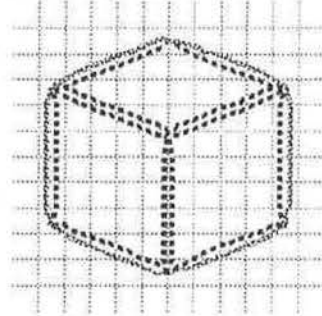
Ambiguous convexity and slant sign interpretations exist on the edges of the concave Y-junction in the center of the drawing. This is due to its initial preference as a convex junction and to the subsequent assignment of preferred complementary values based on values from its neighbors. The corresponding ambiguity in these neighbors (i.e., the convex Y-junctions) is removed via the certainty in the L-junction interpretations. This again illustrates that many of the unambiguous interpretations are first formed on the outside of the drawing and then propagated inward.

Because slant magnitude does not depend on the convex/concave distinction, it is unambiguously assigned to all lines, limited only by the transmission distance.

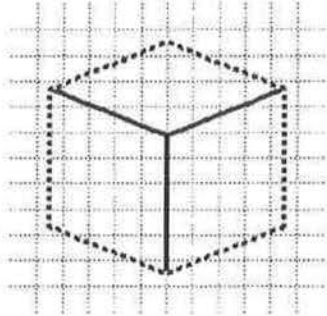
Contiguity (C)



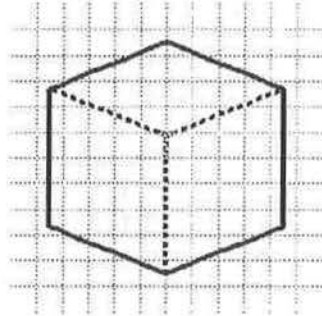
Noncontiguity (N)



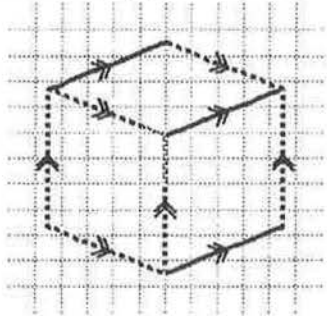
Convexity (+)



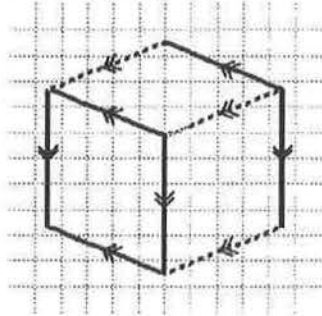
Nonconvexity (o)



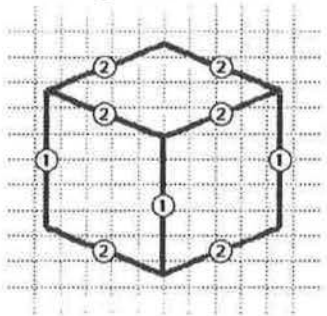
Slant Sign (1)



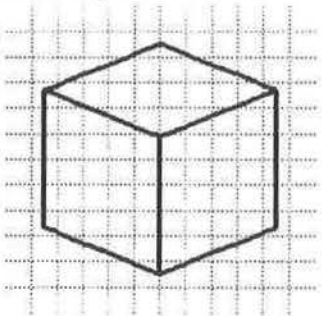
Slant Sign (2)



Slant Magnitude (Value)



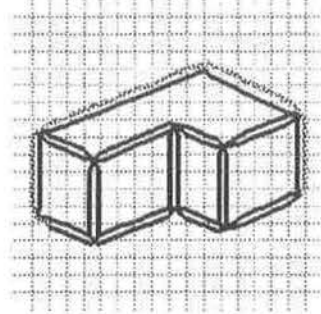
Slant Magnitude (Confidence)



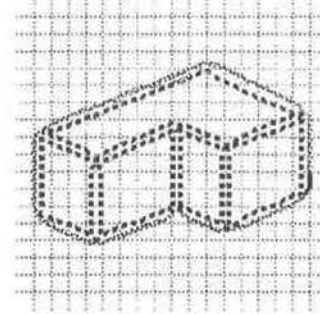
Magnitude	Impossible	Possible	Preferred
Narrow gray lines mark cell boundaries			

Figure 6.1: Interpretation of convex rectangular object. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

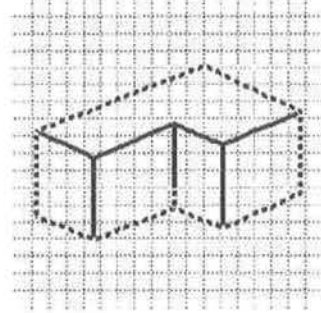
Contiguity (C)



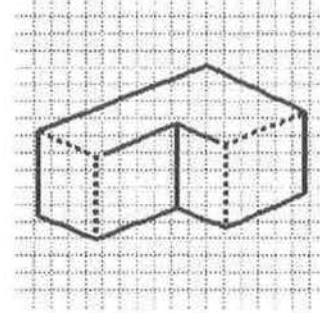
Noncontiguity (N)



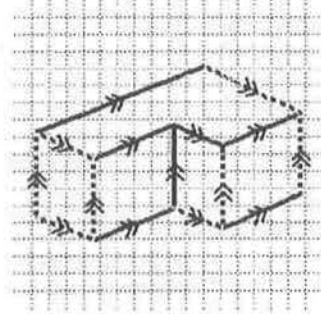
Convexity (+)



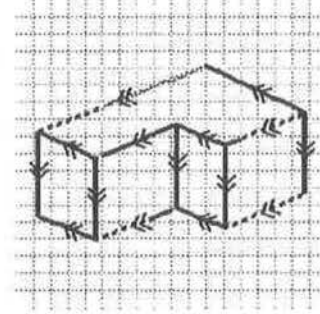
Nonconvexity (o)



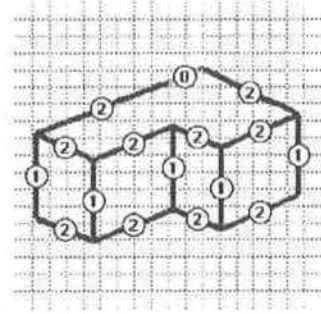
Slant Sign (1)



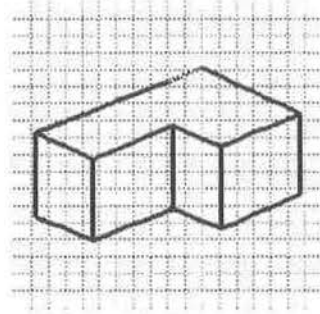
Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)



Magnitude	Impossible	Possible	Preferred
Narrow gray lines mark cell boundaries			

Figure 6.2: Interpretation of nonconvex rectangular object. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

iii) Occluded objects

When several objects exist in the scene, projection to the image plane often results in the partial occlusion of one object by another. Information from the occluded junctions is lost, a loss which is only partially compensated for by the constraints from the T-junctions.

As figure 6.3 shows, however, the recovery process is fairly robust against the effects of occlusion. Assignments of contiguity are as good as those for individual blocks; indeed, they are somewhat less ambiguous, since the T-junctions have added extra information to the crossbars. Convexity and slant are almost unimpaired, with only a slight increase in the area of uncertainty around the central Y-junctions.

The only significant loss of information occurs in the line connected to an occluded arrow-junction on one end, and to an L-junction on the other. The L-junction can provide an assignment of slant sign, but cannot transmit slant magnitudes. Consequently, the line must remain uninterpretable within this stream.

6.1.2 Nonconforming Objects

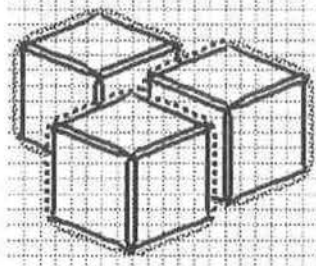
Another test of rapid recovery concerns its ability to interpret line drawings of "nonconforming" objects, i.e., those that do not conform to all the structural assumptions that underlie the recovery process. The ability of the process to recover various scene properties under such conditions provides an indication of its robustness in domains beyond those for which it is optimal (cf. section 2.3).

i) Nonrectangular objects

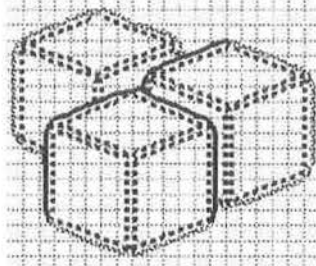
Given the importance of rectangularity for the initial assignment of slant magnitudes (section 3.2.4) and the constraints on convexity (section 3.2.2) and slant signs (section 3.2.3), it is important to determine how recovery is affected when these assumptions are no longer true of the scene. From figure 6.4, it is seen that the process can still recover a fair amount of structure. The inner edges of all lines are interpreted unambiguously as contiguous. The outer edges of the drawing remain largely uninterpreted. When more iterations are allowed the contiguity interpretation assigned to the acute L-junction spreads around the outside of the drawing.

Most of the trilinear junctions have been given unambiguous interpretations in the convexity and slant sign dimensions. Although a contradiction in slant magnitude has been

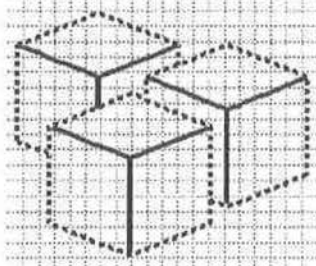
Contiguity (C)



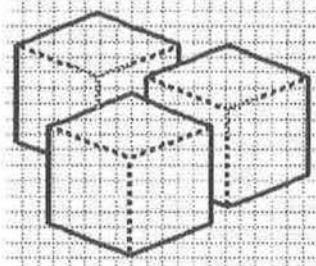
Noncontiguity (N)



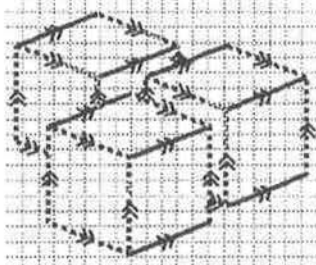
Convexity (+)



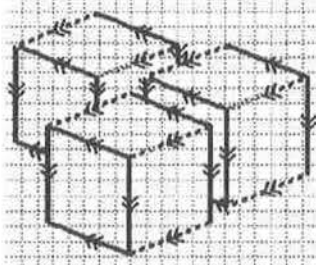
Nonconvexity (o)



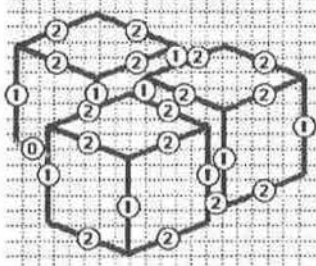
Slant Sign (1)



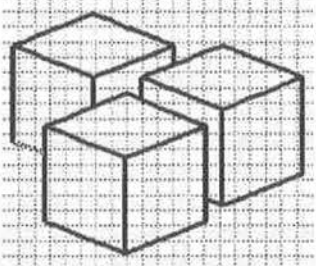
Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)



Magnitude	Impossible	Possible	Preferred
Narrow gray lines mark cell boundaries			

Figure 6.3: Interpretation of occluded rectangular objects. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

found for one of the edges, and cannot be assigned to two others (since the junctions violate Perkins' laws), the remaining four edges have been assigned definite values.

ii) Origami objects

Another class of objects that do not conform to the structural assumptions are the *origami objects* [Kan80], formed by joining extremely thin polygonal plates to each other along their edges. Although they are similar to polyhedra in having planar surfaces, origami objects are never solid, and so their projections cannot be interpreted as solid polyhedra. An example of such a drawing is the chevron shown in figure 6.5.

As seen from figure 6.5, the interpretation process is fairly robust to the violation of this assumption. Most of the outer edges are interpreted as contiguous, an interpretation at odds with that given to the convex block. But three of the four inner edges of the rectangles are still interpreted unambiguously as being contiguous. The results in the other three streams are largely unaffected by the violation of this assumption, with the interpretations matching almost exactly with those of the solid convex block.

iii) Nonplanar objects

Much of the power of a line interpretation process stems from a basic assumption that the surfaces of the corresponding object are planar (see section 2.2.1). The drawing in figure 6.6 violates this basic assumption, the upper surface being uninterpretable as a plane.

The local nature of the rapid recovery process, however, allows much of the structure of nonplanar objects to be recovered, since global consistency is not enforced. This is illustrated in the interpretations shown in figure 6.6. Contiguity is assigned correctly almost everywhere, with inconsistent interpretations assigned only to the inner edges of the notch in the upper surface. Similar considerations apply to convexity and slant sign. Furthermore, slant magnitudes are unambiguously assigned to all lines, a result due to the absence of a check on slant magnitude at L_r junctions (section 4.1.3).

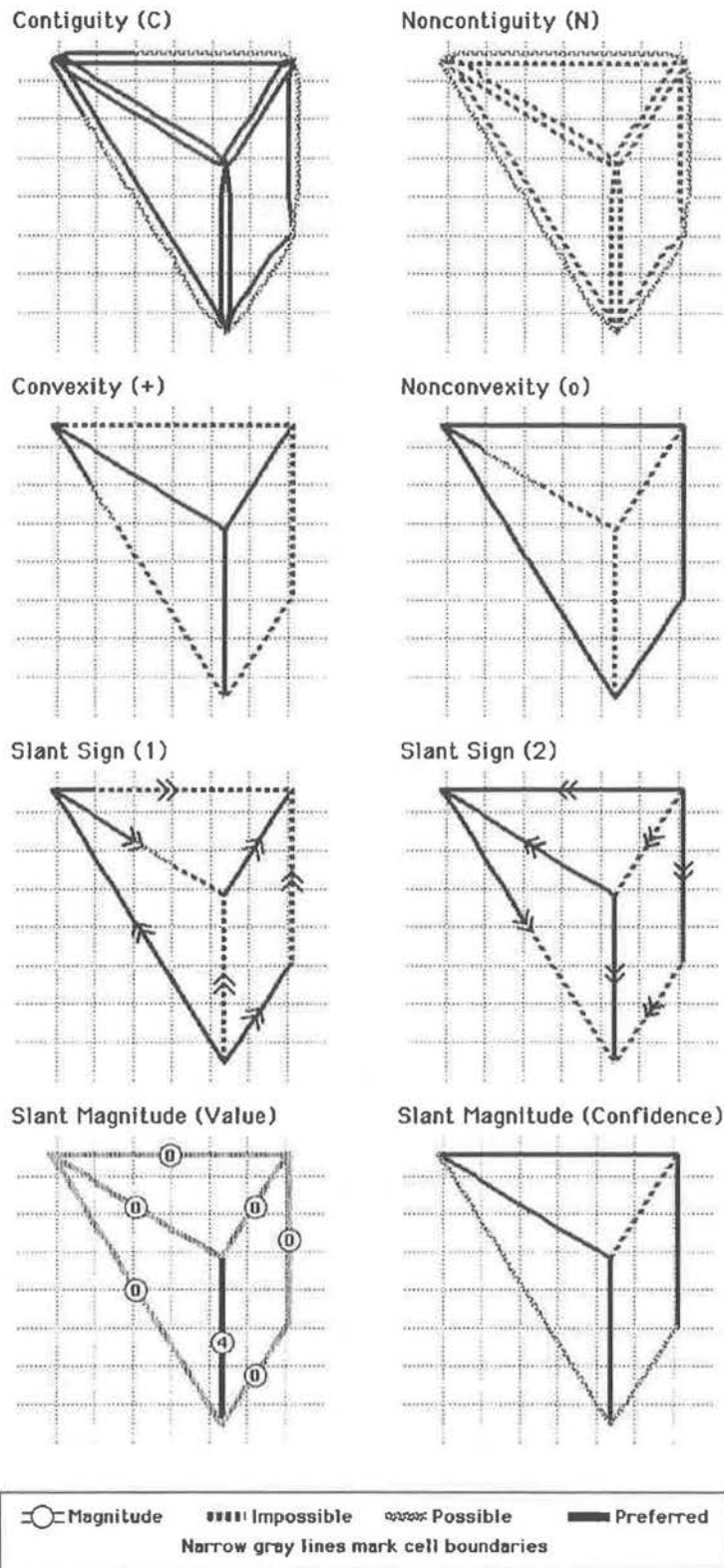
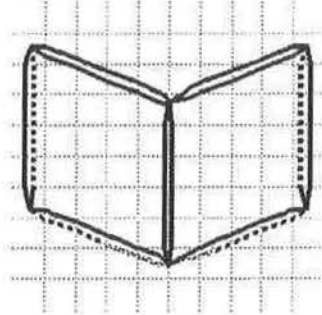
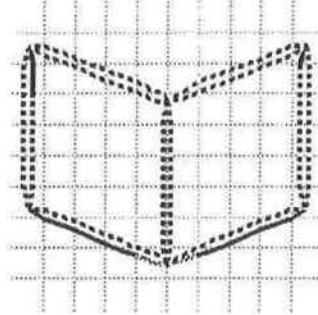


Figure 6.4: Interpretation of nonrectangular object. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

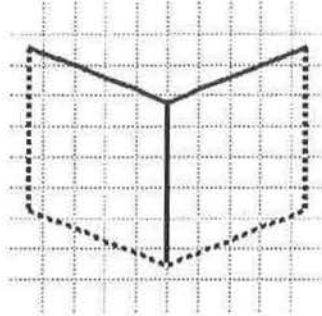
Contiguity (C)



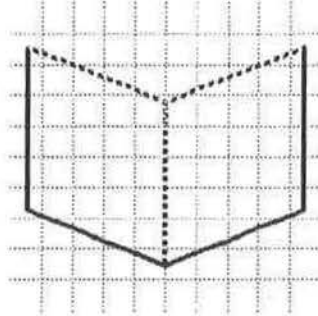
Noncontiguity (N)



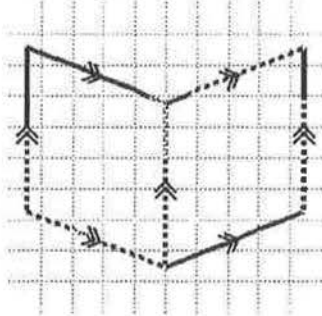
Convexity (+)



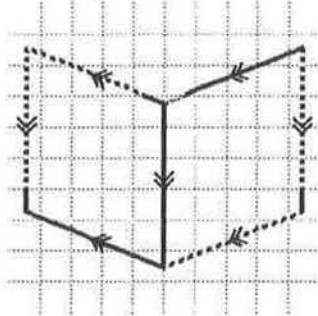
Nonconvexity (o)



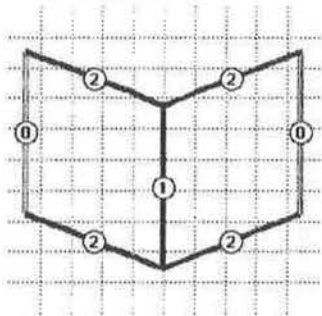
Slant Sign (1)



Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)

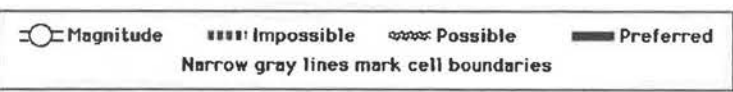
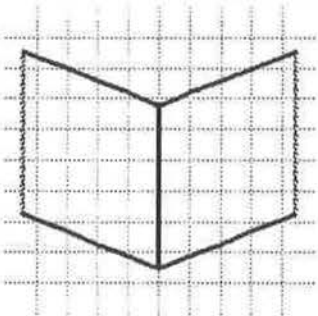
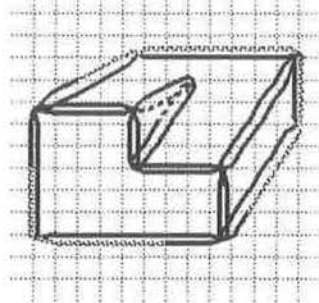
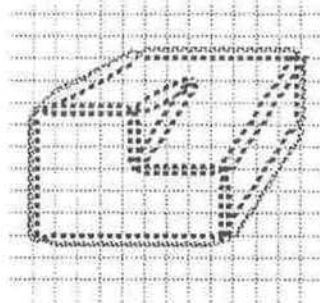


Figure 6.5: Interpretation of origami object. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

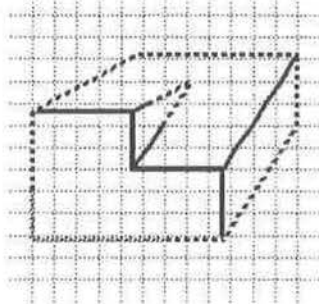
Contiguity (C)



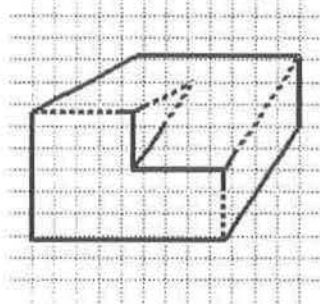
Noncontiguity (N)



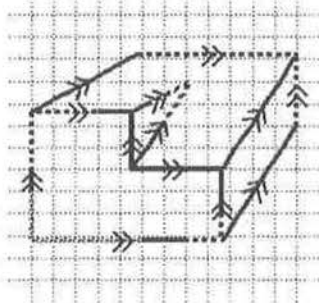
Convexity (+)



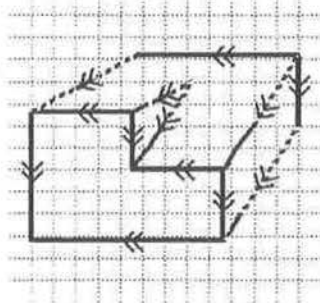
Nonconvexity (o)



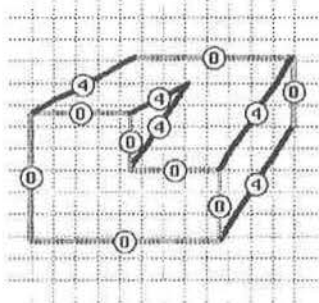
Slant Sign (1)



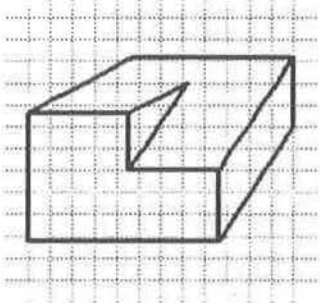
Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)



Magnitude	Impossible	Possible	Preferred
Narrow gray lines mark cell boundaries			

Figure 6.6: Interpretation of nonplanar object. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

6.1.3 Impossible Objects

Objects are said to be “impossible” if they cannot exist under the assumption that connecting lines in the image correspond to connecting edges in the scene. If accidental alignments are allowed, connecting image lines can correspond to disconnected edges in the scene, so that a corresponding object can be found for any line drawing [Kul87]. But the conditions required for this are extremely unstable, violating the general viewpoint constraint (section 3.2), so that such interpretations are not generally allowed. Instead, the drawing is interpreted as an impossible figure containing a set of globally inconsistent interpretations.

As a final test of its abilities, the rapid recovery process is applied to drawings of these impossible objects. To keep the influence of other factors to a minimum, all junctions are such that they can be consistently interpreted as rectangular corners. The application of the recovery process to these drawings consequently provides a good test of how well it can handle global inconsistency.

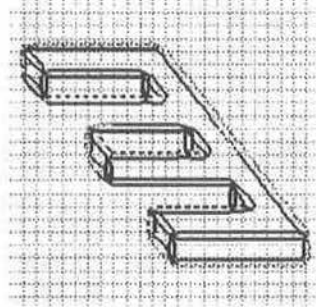
i) Objects of inconsistent contiguity and convexity

The first class of impossible objects are those that correspond to drawings that cannot be given a consistent set of contiguity and convexity labellings. The example considered here is shown in figure 6.7. Such drawings violate the basic assumption that a surface contiguous with a given edge remains contiguous throughout its entire length; among other things, this eliminates the distinction between object and background [Kul87]. In addition, several of the lines cannot be given a consistent convexity interpretation along their length, providing a second source of inconsistency.

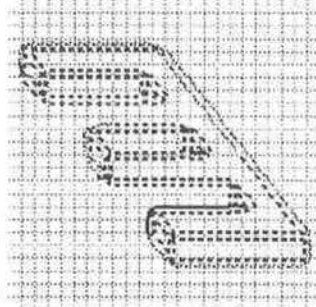
Because the interpretation process involves only local sections of the drawing, however, it is relatively robust to such inconsistencies. This is illustrated in figure 6.7. Here, almost all lines are given an unambiguous contiguity interpretation that is correct locally. The only exceptions in this stream are two horizontal lines that have been interpreted as inconsistent.

Inconsistencies in convexity and slant sign are also picked up, but these are restricted entirely to the inner lines, the outer sections having a completely unambiguous interpretation. Slant magnitude is completely unaffected by the inconsistencies in contiguity and convexity, with unambiguous interpretations assigned to virtually all lines.

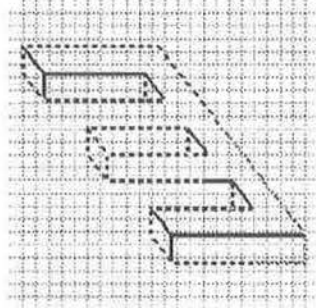
Contiguity (C)



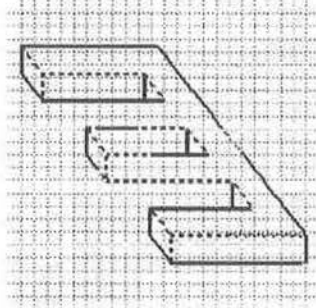
Noncontiguity (N)



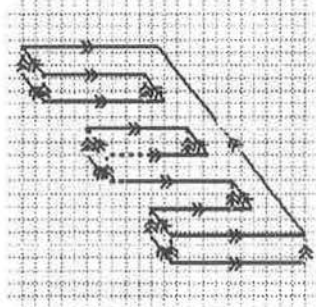
Convexity (+)



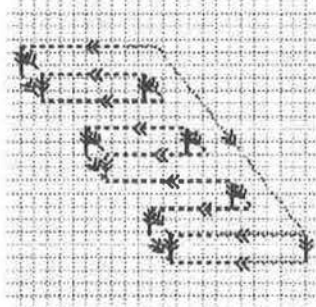
Nonconvexity (o)



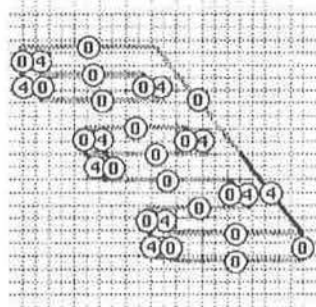
Slant Sign (1)



Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)

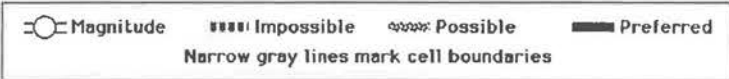
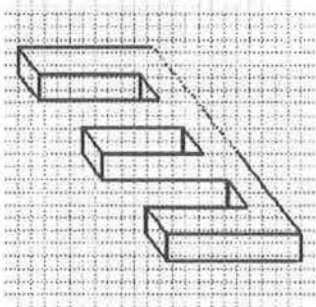


Figure 6.7: Interpretation of object of inconsistent contiguity and convexity. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

ii) Objects of inconsistent slant

Another class of impossible objects give rise to drawings in which the lines cannot be given a consistent set of slant estimates. An example is shown in figure 6.8. Such inconsistency negates the basis for the propagation of slant estimates along common edges.

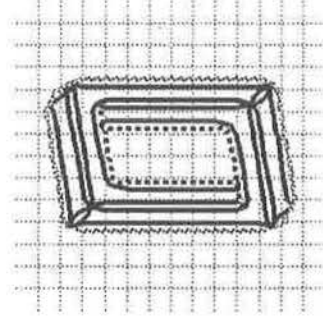
The results of the recovery process are shown in figure 6.8. As seen from this figure, much of the (local) three-dimensional structure is still recovered. Contiguity is assigned correctly to all lines, the only uncertainty existing in the outer edges. Convexity also is largely unaffected, although inconsistencies have begun to appear in the Y-junctions. These inconsistencies are more severe in the slant sign stream, although the arrow-junctions and L-junctions retain unambiguous interpretations. Because the estimation of slant magnitudes is independent of slant sign, unambiguous magnitude estimates are assigned to all the lines.

iii) Objects of inconsistent depth

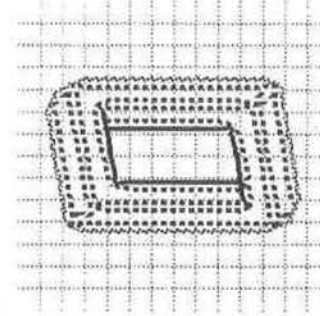
Part of the reason for the speed of the rapid recovery process is that it avoids global checks of the resulting description, using the consistency of the world itself as the basis for coherent interpretations. One important example of this is the complete lack of any check on depth information (section 3.1.1). This renders the process susceptible to a number of "illusions" on drawings for which the corresponding surfaces have globally inconsistent depths. An example of such a drawing is the Penrose triangle, shown in figure 6.9.

As seen from this figure, all four streams result in interpretations that are largely unambiguous for all lines. The only exceptions are uncertain contiguity estimates for the outer lines of the drawing, and uncertain slant estimates for the innermost lines. Both of these are to be expected, since the uncertainty in outer contiguity occurs for almost all drawings, and the uncertainty in slant estimates is a consequence of the inner lines contacting only L- and T-junctions, neither of which can give rise to a magnitude estimate. Virtually all local structure is therefore recovered, with no inconsistencies being detected. The interpretation is an illusion of exactly the type expected, with virtually all edges assigned definite interpretations even though the corresponding object cannot be realized.

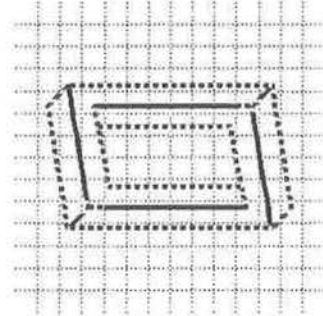
Contiguity (C)



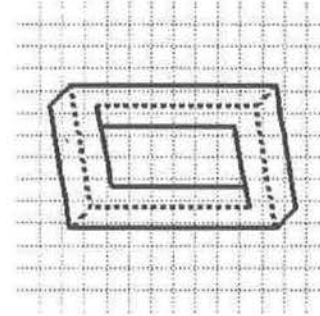
Noncontiguity (N)



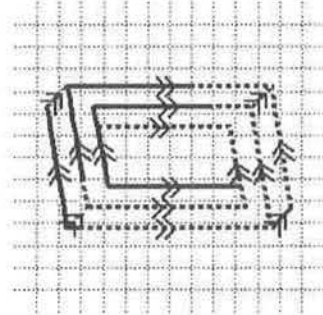
Convexity (+)



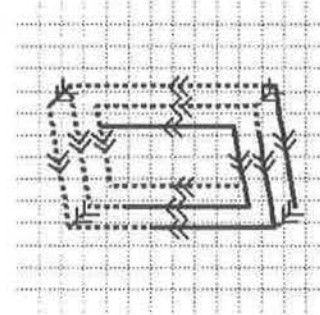
Nonconvexity (o)



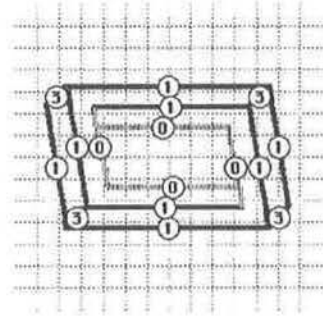
Slant Sign (1)



Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)

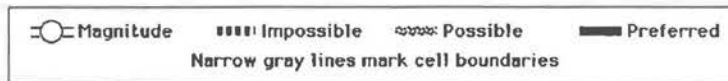
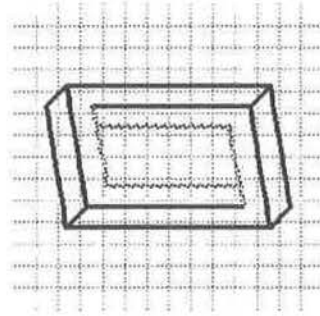
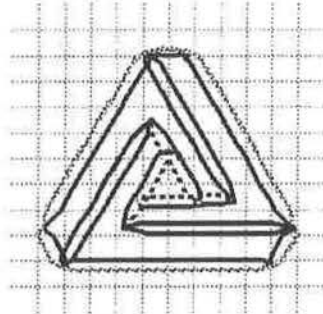
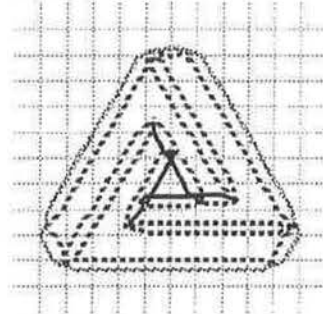


Figure 6.8: Interpretation of object of inconsistent slant. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

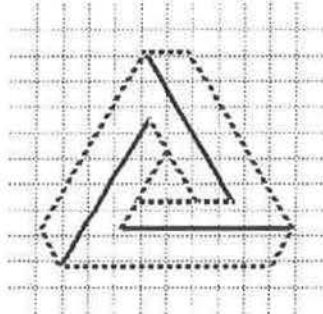
Contiguity (C)



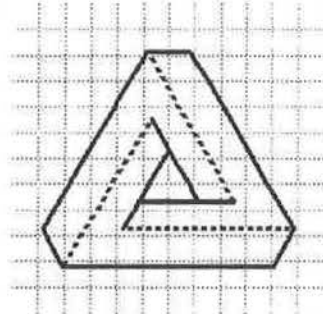
Noncontiguity (N)



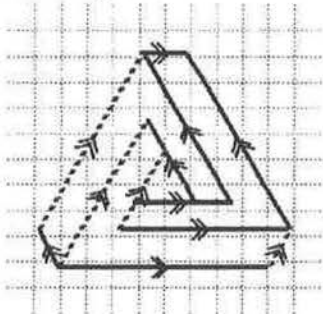
Convexity (+)



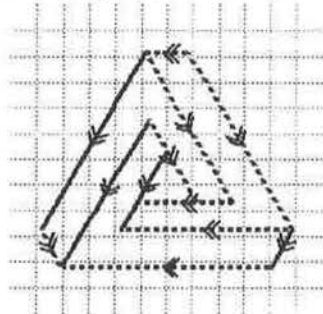
Nonconvexity (o)



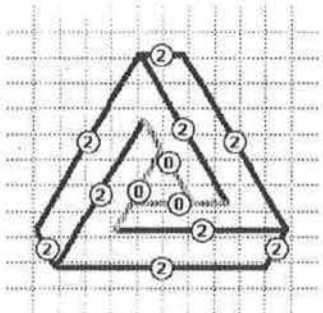
Slant Sign (1)



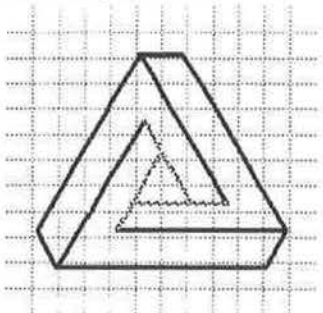
Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)



Magnitude	Impossible	Possible	Preferred
Narrow gray lines mark cell boundaries			

Figure 6.9: Interpretation of object of inconsistent depth. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

6.2 Preattentive Recovery of Scene Structure

The ultimate goal of the theory developed here is to explain the rapid recovery of three-dimensional structure in human early vision. In particular, the goal is to explain why certain kinds of line drawings can be rapidly detected in visual search tasks, and why others cannot. Figure 6.10 shows the set of results considered. The search items, together with the search rates, are taken from [ER91] and [ER92]. In all cases, two search rates are presented – those for displays in which the target is present, and those for which it is absent. The recovery ratio ρ is the measure developed in section 6.2.1 to explain these rates. Although not exhaustive, this set is representative of what is known about search rates for various kinds of line drawings.

By making several relatively simple assumptions about the relation of recovered structure to search rates, the theory is able to explain the relative difficulty of search for all cases examined. Because these assumptions are fairly general, they also allow predictions to be made for drawings not yet tested.

6.2.1 Basic Assumptions

Time and Space Parameters

To carry out the analysis, it is necessary to specify both the size of the drawings and the amount of time to be allocated. In what follows, drawings are scaled to have the same maximum extension. This is done so that the relative sizes match those of the drawings used for the experiments described in [ER91] and [ER92]. The extent of the drawings is taken to be 5 cells. If cells are related to hypercolumns (section 5.3), this will correspond closely to the actual number of hypercolumns involved.

The time limit is set at 5 iterations — enough for a one-time propagation of information across the maximum extent of the drawing. This is only meant to be a representative value, useful as the basis for a comparison of the difficulty of interpretation for various kinds of drawings.

Relating structure to search rates

Since the goal of this work is to explain the relative preference for certain kinds of line drawings over others, and not the phenomenon of rapid detection *per se*, no commitment is

















<u>Condition</u>	<u>Search Items</u>		<u>Rates (ms/item)</u>		ρ
	<u>Target</u>	<u>Distractor</u>	<u>Present</u>	<u>Absent</u>	
A.			7	12	∞
B.			51	96	0.4
C.			6	9	∞
D.			22	31	0.0
E.			35	65	1.2
F.			37	66	0.0
G.			52	80	1.1
H.			63	101	0.0

Figure 6.10: Results explained by theory. The search items and search rates are taken from [ER91] and [ER92]. The recovery ratio ρ , discussed in section 6.2.1, describes the difference in the recovered three-dimensional structure of the target and distractor items. The correlation between ρ and search rate is evident.

made here to any particular model of visual attention or visual search. Instead, a set of four relatively general assumptions is used to relate recovered structure to search rates:

1. **Search rates increase with greater target-distractor distinctiveness.** This assumes that search rates are largely governed by a signal-to-noise ratio that compares the relative number of distinctive features in the target to the number of features it shares with the distractors. This is a widely-accepted assumption used to explain search rates for many kinds of visual stimuli (e.g., [TG88, DH89]).
2. **Target-distractor distinctiveness is based on differences in slant.** It is assumed that the slant sign and slant magnitude of each line in the interpreted drawing are combined into a single quantity that acts as an irreducible feature, capable of being detected almost immediately when sufficiently distinct (section 2.1.2). This assumption is supported by the finding that the speed of search for line drawings can be better explained in terms of three-dimensional rather than two-dimensional orientation [ER90b].
3. **Common uninterpretable lines increase target-distractor similarity.** Uninterpretable lines are assumed to be part of the "noise" that interferes with the process of distinguishing target from distractor in visual search tasks. Such interference could exist for a variety of reasons. If, for example, the rapid-recovery system acted only to eliminate impossible interpretations, lines without a definite slant estimate would be assigned all possible values. This set of values would therefore be common to both target and distractor.
4. **Slant is represented as a departure from zero.** This takes slant to be a quantity like two-dimensional orientation, which is represented as a departure from the canonical orientations of vertical or horizontal [TG88]. Here, the canonical value is assumed to be zero, i.e., a three-dimensional orientation perpendicular to the line of sight.

To obtain a quantitative measure of target-distractor similarity, additional assumptions are needed to refine the original set:

- 1'. **Search rates increase with the recovery ratio ρ .** The quantity ρ is defined here as the ratio of the target-distractor difference over the target-distractor similarity. Although this is a considerable simplification that among other things completely ignores configurational effects among two-dimensional features, it nevertheless provides a rough quantitative measure that captures something of the trade-off between distinctiveness and similarity.
- 2'. **Differences are based on unambiguous slant estimates.** Unambiguous estimates are those for which a slant magnitude has been assigned to the line (section) and for which one of the slant signs is preferred. In light of assumption 4, only

differences in nonzero slants contribute to the distinctiveness measure — since target and distractor always differ by a 180° rotation in the image; the slants of corresponding lines differ in their sign. Consequently, the slant difference is always twice the value of the slant itself. Since the exact location of a feature is not important in visual search (section 2.1.2), ambiguity may also arise if two lines of the same orientation have different slants.

3'. Similarities are based on ambiguous slant estimates. If an ambiguous interpretation exists for the slant sign, or if a slant magnitude is not possible, the corresponding line segment is considered to add to similarity in the same way as uninterpreted lines.

4'. Slant signals are proportional to line length along each orientation.

In effect, each small section of line is assumed to signal the value of the slant at its location, and to pass this value on to the mechanisms governing visual search. Since the location of a feature is not important for this purpose (cf. section 2.1.2), all signals from a common orientation can simply be summed together. The total signal is therefore proportional to the cumulative length along a particular direction. In order to avoid specifying different weights for different slants, each nonzero slant and slant difference are assigned the same value.

In summary, then, search rates are assumed to increase with the recovery ratio ρ , defined as

$$\rho = \frac{\sum_{\theta} \sum_j (\text{segment } \sigma_{\theta j} \text{ has unambiguous nonzero slant})}{\sum_{\theta} \sum_j (\text{segment } \sigma_{\theta j} \text{ has ambiguous slant})},$$

where $\sigma_{\theta j}$ denotes a line segment of orientation θ in the image. Because of the asymmetry between upward and downward slants [ER90b] (also see fig 1.1), this ratio is taken to apply only to cases where the object corresponding to the target is slanted *upward*. Again, it should be emphasized that the theory developed here is not addressed towards explaining such an asymmetry, but rather is only intended to explain the relative difficulty of search.

6.2.2 Explanation of Psychophysical Results

Context Effects

The first test of the theory is to see if it can explain why different contexts influence the detectability of a Y-junction among a set of similar junctions rotated by 180°. The detectability of this junction is greatly affected by the presence and shape of the surrounding outline, as shown in figure 6.10, taken from [ER91]. When Y-junctions are surrounded by a

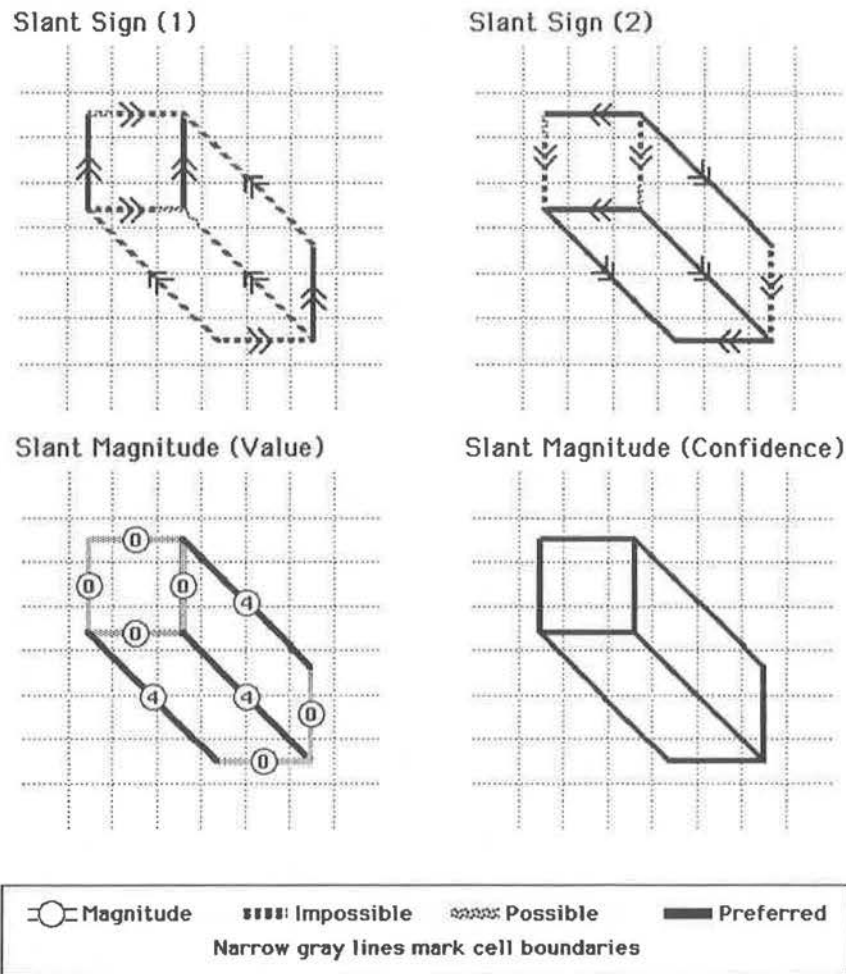


Figure 6.11: Slant estimates for Condition A. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

quasi-hexagonal frame (Condition A) they are detected quite rapidly (7 ms/item for target present; 12 ms/item for target absent). But a square surround (Condition B) causes search to slow down considerably (51 ms/item for target present; 96 ms/item for target absent).

A comparison of the interpretations for Condition A (figure 6.11) and for Condition B (figure 6.12) shows that this effect is readily explained in terms of the recovered three-dimensional structure. The interpretation of Condition A contains no ambiguity in regards to slant, with a considerable difference between target and distractor. Since there are no nonzero slants in common, the recovery ratio ρ is infinite, accounting for the fast search that occurs for this condition.

The drawing of Condition B, on the other hand, has ambiguous estimates for several lines.

Although the long stem of the Y-junction is assigned a unique slant, this is only one-third the "signal" obtained from the drawing of Condition A. Furthermore, a considerable amount of uninterpretable structure exists. The recovery ratio therefore has a relatively low value ($\rho = 0.4$), which explains the much lower search rate.

Contiguity

To determine whether the slow search found in Condition B is due to the failure of the recovery process or simply due to the presence of T-junctions, consider the drawings of Condition C and Condition D, taken from [ER92]. As seen from the figure, search for the target in Condition C is fast, with about the same speed as for that of Condition A. Consider now the drawings of Condition D. Since targets composed of two items can be easily detected in a background of single items [TG88], the target should be easy to detect if the distractor is not segmented into two groups. The target also differs in overall shape in the image, which can only help to speed search. But the search rates (22 ms/item for target present; 31 ms/item for target absent) clearly show that search is relatively difficult.

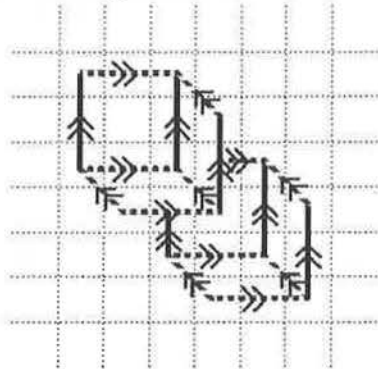
The interpretation of the drawing in Condition C is shown in figure 6.13. The T-junctions have partitioned the drawing into two groups, each of these being interpreted as a complete block with unambiguous slants assigned to all lines. The high recovery ratio is therefore high ($\rho = \infty$), explaining the high speed of search.

The distractors in Condition D, being identical to those of Condition C, have likewise been interpreted as a pair of blocks. Since all nonzero slants match those of the separate blocks in the target item, however, no slant differences exist, and so ρ is zero. Target and distractor differ only in the relative location of their parts. and since relative location cannot be determined at early levels (e.g., [Jul84a, Tre88], search is to be expected to be relatively slow. Although search is faster than in Condition B, this can easily be attributed to some weak effect resulting from the overall difference in two-dimensional shape.

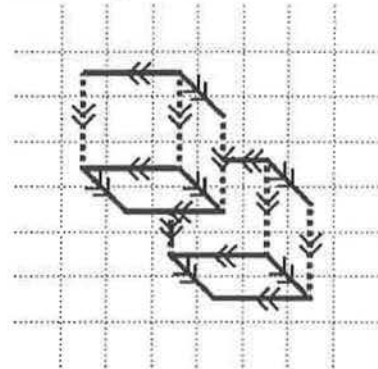
Rectangularity

Conditions E and F (taken from [ER91]) provide a direct test of the rectangularity constraint. These drawings have been distorted so as to violate the assumption of rectangularity in two different ways. In Condition E, the internal Y-junction has been altered so that the system of junctions cannot be consistently interpreted as rectangular; indeed, the top surface is no

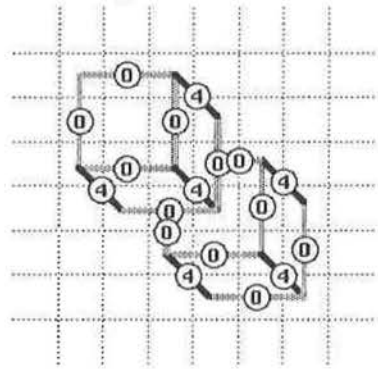
Slant Sign (1)



Slant Sign (2)



Slant Magnitude (Value)



Slant Magnitude (Confidence)

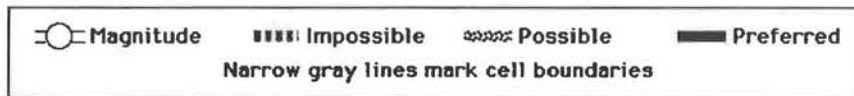
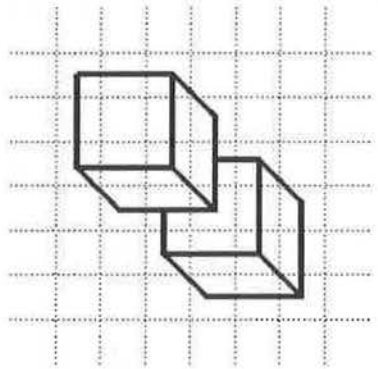


Figure 6.13: Slant estimates for Condition C. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

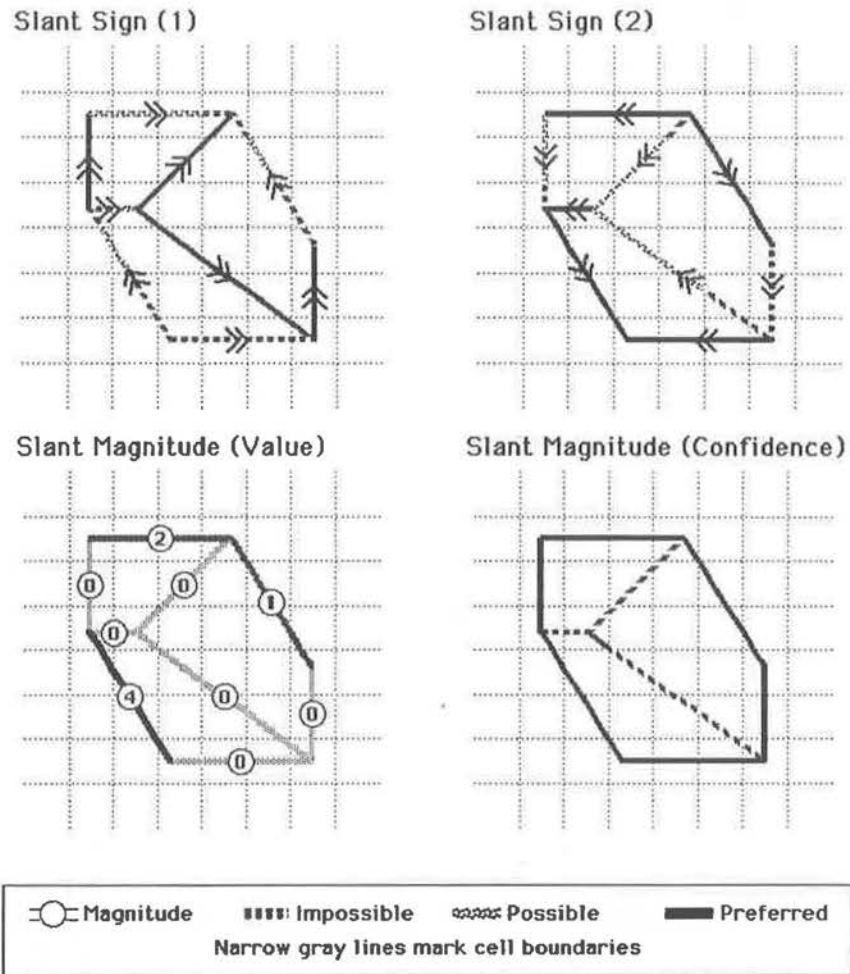


Figure 6.14: Slant estimates for Condition E. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

longer even planar. This condition leads to slow search. To control for the possibility that parallelism rather than rectangularity is the key property, Condition F uses a cube stretched vertically, so that parallel lines remain parallel while both the Y-junction and arrow-junction now violate Perkins' laws. Search is again slowed.

The interpretation of the drawing in Condition E is shown in figure 6.14. Here, the distortions of the junctions have created conflicts in both slant sign and slant magnitude along several lines. The low value of the recovery ratio ($\rho = 1.2$) then explains the slow search speeds found. The results of Condition F are also easily explained – since the junctions violate Perkins' laws, an initial assignment of slant magnitude is not even attempted. Consequently, ρ is zero and search is slow.

Connectedness

To test the possibility that rapid recovery is based on the direct lookup of complete objects rather than via the interaction of more local structures (cf. section 2.3), search rates were determined for the drawings of Conditions G and H (taken from [ER91]). Condition G corresponds to the rectangular block of Condition A, with gaps introduced midway along the lengths of the lines. If lookup depends on the presence of local features alone, search rates should be similar to those for Condition A. However, search slows down dramatically for this condition (52 ms/item for target present; 80 ms/item for target absent). A similar situation arises in Condition H, where the junctions themselves have been removed, leaving only a set of isolated lines in place. Again, search slows down considerably (63 ms/item for target present; 101 ms/item for target absent). These results show that junctions are necessary for three-dimensional orientation to be recovered, but that they are not sufficient.

Although difficult to account for by a process based on the lookup of complete objects, these results are readily explained by the rapid-recovery process developed here. The interpretation of the drawing in Condition G is shown in figure 6.15. The introduction of the gaps results in two major differences from the estimates for Condition A: (i) instead of a single object, the drawing gives rise to a number of smaller parts scattered about the image, and (ii) the isolation of the L-junction prevents them from receiving any kind of slant estimate. Two sources of slowdown therefore emerge: not only are there a larger number of items to be considered, but the recovery ratio itself has a low value ($\rho = 1.1$) due to the uninterpreted L-junctions.²

An even simpler explanation can be given for the results of Condition H. Here, the absence of junctions prevents any slant estimate from being assigned to the lines. As such, they are left as sets of simple two-dimensional objects, which require higher-level processing to be grouped into assemblies corresponding to three-dimensional objects.

²A scatter in slant estimates would also result if lines in the drawings are sufficiently small that accurate orientation measurements cannot be made. This scatter could only reduce search rates further.

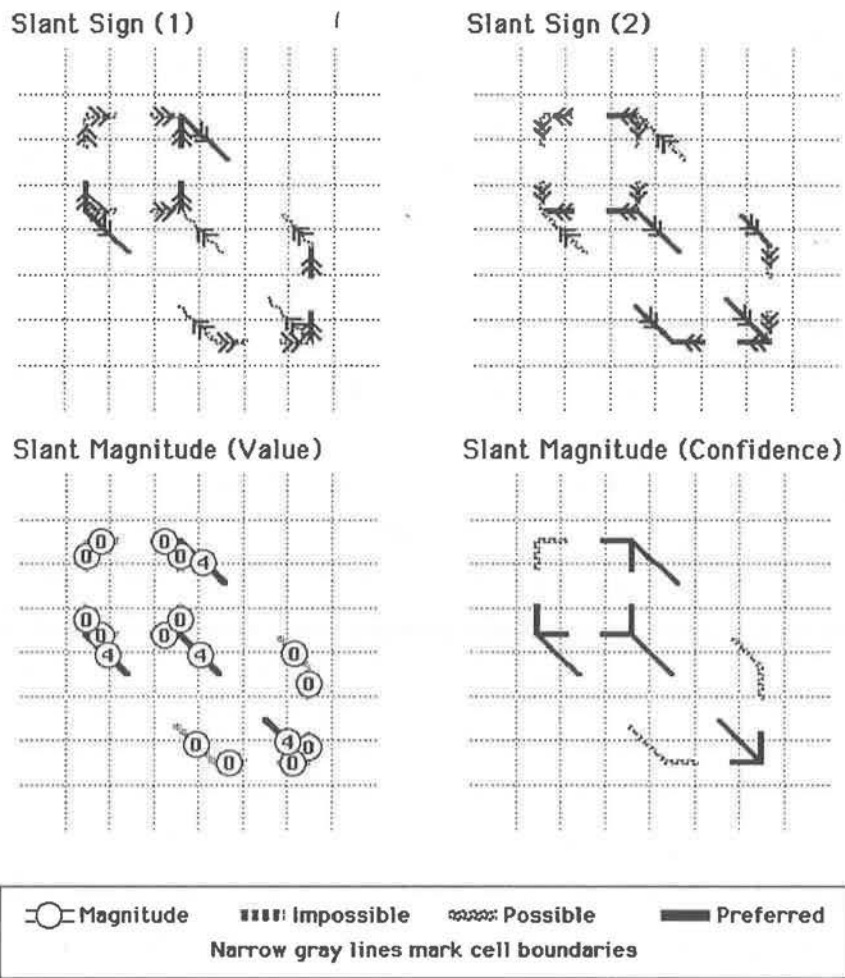


Figure 6.15: Slant estimates for Condition G. Slant angle (in degrees) obtained by multiplying slant magnitude number by 20.

Chapter 7

Summary and Conclusions

A computational theory is developed to explain the rapid interpretation of line drawings at early levels of human vision. This is done by first extending the framework of Marr [Mar82] to allow processes to be analyzed in terms of limits on their computational resources. The problem of rapid line interpretation is then examined along two dimensions: (i) reducing the total amount of information to be transmitted, and (ii) making effective use of the information that is processed. The first of these is addressed by developing constraints on the structure of the recovered object that allow it to be interpreted in sublinear time. The second is handled by constraints on the dynamic operation of the recovery process so that it considers the most likely interpretations first. It is shown that the resulting process can be implemented on a mesh of simple processing elements, and that it can recover a considerable amount of three-dimensional structure in very little time. It also is shown that such a process can explain the ability of human vision to recover three-dimensional orientation at preattentive levels.

These results are relevant to several areas of study. First, the extension of Marr's framework developed in section 2.4 provides a way to discuss the various factors involved when a process is to be explained in terms of limited computational resources. This extension has elements contained in previous attempts to incorporate resource limitations (e.g., [FB82, Tso87]) into a computational framework, but it also puts forward several new distinctions (e.g., external vs. internal constraints, constraint vs. limitation), and treats these in a more systematic way. Although still in rudimentary form, this framework can help guide the development of computational theories for other resource-limited processes.

Another, more concrete framework is the taxonomy of image mappings proposed in section 2.1.1. Here, mappings are grouped on the basis of information flow across the image,

which in turn is related to lower bounds on their computational complexity. The structure of this framework remains conjectural at the moment. If proven, these results would be interesting extensions of the work of Minsky and Papert [MP69] on the abilities of simple parallel architectures to carry out various kinds of operations on images.

The developments in chapter 3 provide several interesting results concerning the complexity of collapsed constraint satisfaction problems. These results support earlier observations (e.g., [Mac74, Mal87]) that such systems can often be solved quite easily. They also show that careful selection and coordination of such “collapsed” subsystems can lead to approximations that are not only soluble in sublinear time, but that also retain much of the information in the original set of constraints. It would be interesting to see whether the approach developed here (viz., separation into weakly interacting subsets of binary and bijective constraints) could be usefully applied in other domains.

The complementary subsystems developed in chapter 4 provide an interesting way to handle local inconsistencies and ambiguities. In particular, their incorporation into a pair of liberal and conservative interpretation schemes suggests a general way to handle interpretation problems that require inconsistency and ambiguity to be explicitly represented and treated in a systematic fashion.

Finally, the results of chapters 5–6 provide support for the view of early vision sketched in section 2.3.2 — that the “horizontal” modules formed by different levels of processing can be complemented by “vertical” columns capable of providing interpretations that are locally consistent. This has implications for the study of both machine and biological vision systems. The algorithms developed in chapter 5 show that this style of processing can be easily incorporated into a machine vision system, allowing it to obtain rapid estimates of scene-based properties at all points in the image. It is seen from the results of section 6.1 that a considerable amount of scene structure can often be recovered this way. Consequently, a rapid recovery process can greatly facilitate the overall operation of a machine vision system.

The results of section 6.2 hold a similar implication for biological vision systems — rapid recovery at early levels can be used to help quickly construct higher-level descriptions of the world. Furthermore, given that line interpretation is relatively difficult at early levels (cf. section 1.1), the results of chapter 6 make it plausible that other kinds of rapid recovery processes may also exist at these levels.

Open Questions and Future Directions

Many of the results concerning the actual performance of the rapid recovery process are based on time and space parameters assumed to be representative of early visual processing. Although suitable as a first approximation, the selection of these values is nevertheless somewhat arbitrary. It would therefore be useful to carry out a set of psychophysical experiments to examine the time course of this process in greater detail, and to see if these values truly are representative. Among other things, such experiments might be able to confirm or refute the theory in regards to the order in which various properties are actually recovered.

A related set of issues applies to the recovery ratio of section 6.2, used to relate recovered structure to search rate. This quantity is sufficient for present purposes, but is only a rough indicator of search difficulty, and ideally would be replaced by a more reliable measure. The general idea that a signal-to-noise ratio largely governs search speed is widely accepted (e.g., [TG88, DH89]), but a more precise measure is not currently known. As such, this problem is not limited to explaining the results of search for line drawings. But as data accumulates from more search experiments, it might at least be possible to refine the recovery ratio to take into account such possibilities as several canonical slant values, and different weights for different slant magnitudes.

A more general set of concerns involves the way in which rapid recovery is related to object recognition. One of the main roles assumed for rapid recovery is to provide early estimates of scene-based properties that facilitate later processes, including those involved with object recognition (section 2.3.2). It is entirely possible, however, that object recognition proceeds by a lookup mechanism that uses simple image properties to retrieve a complete globally-consistent model of the object (e.g., [PE90]). If so, rapid recovery at early levels could be accounted for entirely in this way. The results of section 6.2, however, show that recovery is destroyed by nonrectangular corners and by the introduction of gaps into the drawings, something rather difficult to account for in terms of this mechanism. Furthermore, a theoretical objection can also be raised against the indiscriminate use of lookup tables, since an enormous amount of memory would be required to store all possible views of each object at all possible angles (see section 2.3).

Lookup for a limited number of objects, however, is entirely possible. Indeed, the process developed here can itself be viewed as using a simple form of lookup (cf section 5.1.3), the initial interpretations based on a small number of "local" models invoked by the junctions

and the resulting interpretations then weeded out by *in situ* constraints. Since even the consistency of global models with each other must also be established in some way, the issue is therefore one of determining the appropriate granularity of the models involved. An interesting direction for future research is to ascertain the various levels of granularity that might be used, and to determine how models of different granularity might interact.

In any event, it has been shown here that smaller-grained "local" models are sufficient to allow a substantial amount of three-dimensional structure to be recovered in very little time. It has also been shown that the properties recovered in this way can be used to explain why particular kinds of line drawing are or are not interpreted at early levels of human vision. As such, the central point of this work has been established — substantial amounts of scene structure can be recovered in very little time by splitting a process into quasi-independent streams that are each concerned with a single aspect of scene structure. This principle is, of course, not limited to rapid line interpretation, and it will be interesting to see if it can be applied to other forms of rapid perception and rapid cognition.

Bibliography

- [AF69] F. Attneave and R. Frost. The determination of perceived tridimensional orientation by minimum criteria. *Perception & Psychophysics*, 6B:391–396, 1969.
- [Arb87] M.A. Arbib. *Brains, Machines, and Mathematics (2nd ed.)*. New York and Berlin: Springer, 1987.
- [Att54] F. Attneave. Some informational aspects of visual perception. *Psychological Review*, 61:183–193, 1954.
- [Att72] F. Attneave. Representation of physical space. In A.W. Melton and E. Martin, editors, *Coding Processes in Human Memory*, pages 11–29. Washington DC: V.H. Winston & Sons, 1972.
- [Att82] F. Attneave. Prägnanz and soap bubble systems: A theoretical exploration. In J. Beck, editor, *Organization and Representation in Perception*, pages 11–29. Hillsdale, NJ: Erlbaum, 1982.
- [BA88] J.R. Bergen and E.H. Adelson. Early vision and texture perception. *Nature*, pages 363–364, 1988.
- [Baa78] S. Baase. *Computer Algorithms: Introduction to Design and Analysis*. Reading, MA: Addison-Wesley, 1978.
- [Bal91] D.H. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.
- [BB82] D.H. Ballard and C.M. Brown. *Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, 1982.
- [BCG90] A.C. Bovik, M. Clark, and W.S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:55–73, 1990.
- [Bec66] J. Beck. Effect of orientation and of shape similarity on perceptual grouping. *Perception & Psychophysics*, 1:300–302, 1966.

- [Bec82] J. Beck. Textural segmentation. In J. Beck, editor, *Organization and Representation in Perception*, pages 285–317. Hillsdale, NJ: Erlbaum, 1982.
- [BGS91] J. Beck, N. Graham, and A. Sutter. Lightness differences and the perceived segregation of regions and populations. *Perception & Psychophysics*, 1991.
- [Bie85] I. Biederman. Human image understanding: Recent research and a theory. *Computer Vision, Graphics and Image Processing*, 32:29–73, 1985.
- [Bis84] P.O. Bishop. Processing of visual information within the retinostriate system. In J.M. Brookhart and V.B. Mountcastle, editors, *Handbook of Physiology, Vol. III, Sensory Processes, Part 1*, pages 341–424. Bethesda, MD: American Physiological Society, 1984.
- [BL86] F. Boselie and E.L.J. Leeuwenberg. A test of the minimum principle requires a perceptual coding system. *Perception*, 15:331–354, 1986.
- [Bla89] A. Blake. Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11:2–12, 1989.
- [Bra77] A. Brandt. Multi-level adaptive techniques for partial differential equations: Ideas and software. In J.R. Rice, editor, *Mathematical Software III*, pages 277–318. New York, NY: Academic, 1977.
- [Bri87] W.L. Briggs. *A Multigrid Tutorial*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1987.
- [Bro81] R.A. Brooks. Symbolic reasoning among 3d models and 2d images. *Artificial Intelligence*, 17:285–348, 1981.
- [BT78] H.G. Barrow and J.M. Tenenbaum. Recovering intrinsic scene characteristics from images. In A.R. Hanson and E.M. Riseman, editors, *Computer Vision Systems*, pages 18–24. New York, NY: Academic, 1978.
- [BWH75] K. Berbaum, N. Weisstein, and C.S. Harris. A vertex-superiority effect. *Bull. Psychonomic Society*, 6:418, 1975.
- [Cae84] T. Caelli. On the specification of coding principles of visual image processing. In P. Dodwell and T. Caelli, editors, *Figural Synthesis*, pages 153–184. Hillsdale, NJ: Erlbaum, 1984.
- [CAT90] P. Cavanagh, M. Arguin, and A. Treisman. Effect of surface medium on visual search for orientation and size features. *Journal of Experimental Psychology*, 16:479–491, 1990.

- [CF82] P.R. Cohen and E.A. Feigenbaum. Vision. In E.A. Feigenbaum and P.R. Cohen, editors, *The Handbook of Artificial Intelligence (Vol. 3)*, pages 139–194. Stanford, CA: Heuris Tech Press, 1982.
- [CHY90] K. Culik II, L.P. Hurd, and S. Yu. Computation theoretic aspects of cellular automata. *Physica D*, 45:357–378, 1990.
- [Clo71] M. B. Clowes. On seeing things. *Artificial Intelligence*, 2:79–112, 1971.
- [CW90] K.R. Cave and J.M. Wolfe. Modeling the role of parallel processing in visual search. *Cognitive Psychology*, 22:225–271, 1990.
- [DH89] J. Duncan and G.W. Humphreys. Visual search and stimulus similarity. *Psychological Review*, 96:433–458, 1989.
- [Dra81] S.W. Draper. The use of gradient and dual space in line-drawing representation. *Artificial Intelligence*, 17:461–508, 1981.
- [Dun89] J. Duncan. Boundary conditions on parallel processing in human vision. *Perception*, 18:457–469, 1989.
- [dYvE88] E.A. de Yoe and D.C. van Essen. Concurrent processing streams in monkey visual cortex. *Trends in Neuroscience*, 11:219–226, 1988.
- [Ede87] S. Edelman. Line connectivity algorithms for an asynchronous pyramid computer. *Computer Vision, Graphics, and Image Processing*, 40:169–187, 1987.
- [EIS76] S. Even, A. Itai, and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, 5:691–703, 1976.
- [ER90a] J.T. Enns and R.A. Rensink. Influence of scene-based properties on visual search. *Science*, 247:721–723, 1990.
- [ER90b] J.T. Enns and R.A. Rensink. Sensitivity to three-dimensional orientation in visual search. *Psychological Science*, 1:323–326, 1990.
- [ER91] J.T. Enns and R.A. Rensink. Preattentive recovery of three-dimensional orientation from line drawings. *Psychological Review*, 98:335–351, 1991.
- [ER92] J.T. Enns and R.A. Rensink. A model for the rapid interpretation of line drawings in early vision. In *Visual Search II*. London: Taylor & Francis, 1992.
- [Fal72] G. Falk. Interpretation of imperfect line data as a three-dimensional scene. *Artificial Intelligence*, 3:101–144, 1972.
- [FB82] J. Feldman and D. Ballard. Connectionist models and their properties. *Cognitive Science*, 6:205–254, 1982.

- [Fel85] J. Feldman. Connectionist models and parallelism in high level vision. In A. Rosenfeld, editor, *Human and Machine Vision II*, pages 86–108. New York, NY: Academic, 1985.
- [Fly72] M.J. Flynn. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, 21:948–960, 1972.
- [GA56] B.F. Green and L.K. Anderson. Color coding in a visual search task. *Journal of Experimental Psychology*, 51:19–24, 1956.
- [GA68] A. Guzman-Arenas. Decomposition of a visual scene into three-dimensional bodies. In *Proc. AFIPS Fall Joint Computer Conf.*, pages 291–304, 1968.
- [GB89] R. Gurnsey and R.A. Browse. Asymmetries in visual texture discrimination. *Spatial Vision*, 4:31–44, 1989.
- [GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.
- [GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: W.H. Freeman, 1979.
- [GJM88] I.I. Glezer, M.S. Jacobs, and P.J. Morgane. Implications of the ‘initial brain’ concept for brain evolution in cetacea. *Behavioral and Brain Sciences*, 11:75–116, 1988.
- [Gla84] F. Glazer. Multilevel relaxation in low-level computer vision. In A. Rosenfeld, editor, *Multiresolution Image Processing and Analysis*, pages 312–330. New York and Berlin: Springer, 1984.
- [Gol69] M.J.E. Golay. Hexagonal parallel pattern transformations. *IEEE Transactions on Computers*, 18:733–740, 1969.
- [Gou89] S.J. Gould. *Wonderful Life: The Burgess Shale and the Nature of History*. New York: Norton, 1989.
- [GR88] A. Gibbons and W. Rytter. *Efficient Parallel Algorithms*. Cambridge: Cambridge University Press, 1988.
- [Gra85] N. Graham. Detection and identification of near-threshold visual patterns. *Journal of the American Optical Society, A*, 2:1468–1482, 1985.
- [Har87] D. Harel. *Algorithmics*. Reading, MA: Addison-Wesley, 1987.
- [Hil84] W.D. Hillis. The Connection Machine: A computer architecture based on cellular automata. *Physica D*, 10:213–228, 1984.

- [HM53] J. Hochberg and E. McAlister. A quantitative approach to figural goodness. *Journal of Experimental Psychology*, 46:361–364, 1953.
- [Hoc78] J.E. Hochberg. *Perception (2nd ed.)*. Englewood Cliffs, NJ: Prentice-Hall, 1978.
- [Hor86] B.K.P. Horn. *Robot Vision*. Cambridge, MA: MIT Press; New York: McGraw-Hill, 1986.
- [Hub81] D.H. Hubel. Exploration of the primary visual cortex, 1955-78. *Nature*, 299:515–524, 1981.
- [Huf71] D.A. Huffman. Impossible objects as nonsense sentences. In B. Meltzer and D. Michie, editors, *Machine Intelligence 6*, pages 295–323. Edinburgh: Edinburgh University Press, 1971.
- [HW74] D.H. Hubel and T.N. Wiesel. Sequence regularity and geometry of orientation columns in monkey striate cortex. *J. Comparative Neurology*, 158:267–294, 1974.
- [HZ83] R.A. Hummel and S.W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:267–287, 1983.
- [JB83] B. Julesz and J.R. Bergen. Textons, the fundamental elements in preattentive vision and perception of textures. *Bell System Technical Journal*, 62:1619–1645, 1983.
- [Joh90] D.S. Johnson. A catalog of complexity classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Vol. A*, pages 69–161. Amsterdam: Elsevier, 1990.
- [Jul81] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, 1981.
- [Jul84a] B. Julesz. A brief outline of the texton theory of human vision. *Trends in Neuroscience*, 7:41–45, 1984.
- [Jul84b] B. Julesz. Toward an axiomatic theory of preattentive vision. In G.M. Edelman, W.E. Gall, and W.M. Cowan, editors, *Dynamic Aspects of Neocortical Function*, pages 585–612. Neurosciences Research Foundation, 1984.
- [Jul86] B. Julesz. Texton gradients: The texton theory revisited. *Biological Cybernetics*, 54:245–261, 1986.
- [Jul87] B. Julesz. Preattentive human vision, a link between neurophysiology and psychophysics. In F. Plum et al., editor, *Handbook of Physiology, Vol V*. American Physiology Society, 1987.

- [Kan80] T. Kanade. A theory of origami world. *Artificial Intelligence*, 13:279–311, 1980.
- [Kan90] K. Kanatani. *Group-Theoretical Methods in Image Understanding*. New York and Berlin: Springer, 1990.
- [Kar84] N. Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4:373–395, 1984.
- [Kar90] J. Kari. Reversibility of 2d cellular automata is undecidable. *Physica D*, 45:379–385, 1990.
- [Kha79] L.G. Khachiyan. A polynomial algorithm for linear programming. *Soviet Math. Dokl.*, 20:191–194, 1979.
- [KI85] J. Kittler and J. Illingworth. Relaxation labelling algorithms – a review. *Image and Vision Computing*, 3:206–216, 1985.
- [KP85] L.M. Kirousis and C.H. Papadimitriou. The complexity of recognising polyhedral scenes. In *26th FOCS*, 1985.
- [KP88] L.M. Kirousis and C.H. Papadimitriou. The complexity of recognizing polyhedral scenes. *Journal of Computer and System Sciences*, 37:14–38, 1988.
- [KU84] C. Koch and S. Ullman. Selecting one among the many: A simple network implementing shifts in selective visual attention. Technical Report A.I. Memo 770, MIT, 1984.
- [Kul87] Z. Kulpa. Putting order in the impossible. *Perception*, 16:201–214, 1987.
- [Lak78] I. Lakatos. *The Methodology of Scientific Research Programmes*. Cambridge: Cambridge University Press, 1978.
- [LAN89] W. Lim, A. Agrawal, and L. Nekludova. A fast parallel algorithm for labeling connected components in image arrays. In P.M. Dew, R.A. Earnshaw, and T.R. Heywood, editors, *Parallel Processing for Computer Vision and Display*, pages 169–179. Reading, MA: Addison-Wesley, 1989.
- [LBC89] J.J. Little, G.E. Blelloch, and T.A. Cass. Algorithmic techniques for computer vision on a fine-grained parallel machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:244–257, 1989.
- [Lee71] E.L.J. Leeuwenberg. A perceptual coding language for visual and auditory patterns. *American Journal of Psychology*, 84:307–350, 1971.
- [Lev86] H.J. Levesque. Making believers out of computers. *Artificial Intelligence*, 30:81–108, 1986.

- [Low85] D. Lowe. *Perceptual Organization and Visual Recognition*. Boston: Kluwer Academic, 1985.
- [Low87] D. Lowe. The viewpoint consistency constraint. *International Journal of Computer Vision*, 1:57-72, 1987.
- [Mac73] A.K. Mackworth. Interpreting pictures of polyhedral scenes. *Artificial Intelligence*, 4:121-137, 1973.
- [Mac74] A.K. Mackworth. *On the Interpretation of Drawings as Three-Dimensional Scenes*. D.Phil. thesis, University of Sussex, January 1974.
- [Mac76] A.K. Mackworth. Model-driven interpretation in intelligent vision systems. *Perception*, 5(3):349-370, 1976.
- [Mac77] A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99-118, 1977.
- [Mac91] A.K. Mackworth. The logic of constraint satisfaction. Technical Report TR-91-26, Department of Computer Science, University of British Columbia, 1991.
- [Mal87] J. Malik. Interpreting line drawings of curved objects. *Int. J. Computer Vision*, 1:73-103, 1987.
- [Mar79] D. Marr. Representing and computing visual information. In *Artificial Intelligence: An MIT Perspective*, pages 15-80. Cambridge, MA: MIT Press, 1979.
- [Mar82] D. Marr. *Vision*. San Francisco: W.H. Freeman, 1982.
- [MD90] J. Mulder and R.J.M. Dawson. Reconstructing polyhedral scenes from single two-dimensional images: The orthogonality hypothesis. In P.K. Patel-Schneider, editor, *Proceedings of the 8th Biennial Conf. of the CSCSI*, pages 238-244, 1990.
- [MF85] A.K. Mackworth and E.C. Freuder. The complexity of some polynomial network consistency algorithms for constraint satisfaction problems. *Artificial Intelligence*, pages 65-74, 1985.
- [MMH85] A.K. Mackworth, J.A. Mulder, and W.S. Havens. Hierarchical arc consistency: Exploiting structured domains in constraint satisfaction problems. *Computational Intelligence*, 1:71-79, 1985.
- [MN87] J.H. Maunsell and W.T. Newsome. Visual processing in monkey extrastriate cortex. *Ann. Rev. Neurosci.*, 10:363-401, 1987.
- [MP69] M. Minsky and S. Papert. *The Perceptron: Principles of Computational Geometry*. Cambridge, MA: MIT Press, 1969.

- [MS87] R. Miller and Q.F. Stout. Data movement techniques for the pyramid architecture. *SIAM Journal of Computing*, 16:38–60, 1987.
- [Nei63] U. Neisser. Decision time without reaction time: Experiments in visual scanning. *American Journal of Psychology*, 76:376–385, 1963.
- [NKP87] L. Ni, C.T. King, and P. Prins. Parallel algorithm considerations for hypercube management. In *Proceedings of the 1987 Intl. Conf. on Parallel Processing*, pages 717–720, 1987.
- [Not91] H.C. Nothdurft. Different effects from spatial frequency masking in texture segregation and texton detection tasks. *Vision Research*, 31:299–320, 1991.
- [NS86] K. Nakayama and G.H. Silverman. Serial and parallel processing of visual feature conjunctions. *Nature*, 320:264–265, 1986.
- [PC80] D.N. Perkins and R.G. Cooper Jr. How the eye makes up what the light leaves out. In M.A. Hagen, editor, *The Perception of Pictures: Vol II*, pages 95–130. New York, NY: Academic, 1980.
- [PD84] K. Preston, Jr and M.J.B. Duff. *Modern Cellular Automata: Theory and Applications*. New York: Plenum, 1984.
- [PDL⁺79] K. Preston, Jr., M.J.B. Duff, S. Levialdi, P.E. Norgren, and J.-I. Toriwaki. Basics of cellular logic with some applications in medical image processing. *Proc. IEEE*, 67:826–856, 1979.
- [PE90] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.
- [Per68] D.N. Perkins. Cubic corners. Technical Report Quarterly Progress Report No. 89, Research Laboratory of Electronics, MIT, 1968.
- [Per72] D.N. Perkins. Visual discrimination between rectangular and nonrectangular parallelepipeds. *Perception & Psychophysics*, 12:396–400, 1972.
- [Per76] D.N. Perkins. How good a bet is good form? *Perception*, 5:393–406, 1976.
- [Per82] D.N. Perkins. The perceiver as organizer and geometer. In J. Beck, editor, *Organization and Representation in Perception*, pages 73–93. Hillsdale, NJ: Erlbaum, 1982.
- [PTK85] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317:314–319, 1985.
- [Rab78] P. Rabbitt. Sorting, categorization, and visual search. In *Handbook of Perception, Vol. 10*. New York, NY: Academic, 1978.

- [Rab84] P. Rabbitt. The control of attention in visual search. In *Varieties of Attention*, pages 273–291. New York, NY: Academic, 1984.
- [Ram85] V.S. Ramachandran. The neurobiology of perception. *Perception*, 14:97–103, 1985.
- [Ram88] V.S. Ramachandran. Perceiving shape from shading. *Sci. Am.*, 259:76–83, Aug 1988.
- [Ree84] A.P. Reeves. Parallel computer architectures for image processing. *Computer Vision, Graphics, and Image Processing*, 25:68–88, 1984.
- [Res82] F. Restle. Coding theory as an integration of gestalt psychology and information theory. In J. Beck, editor, *Organization and Representation in Perception*, pages 31–56. Hillsdale, NJ: Erlbaum, 1982.
- [Ric88] W. Richards. Image interpretation: Information at contours. In W. Richards, editor, *Natural Computation*, pages 17–36. Cambridge, MA: MIT Press, 1988.
- [RM89] R. Reiter and A.K. Mackworth. A logical framework for depiction and image interpretation. *Artificial Intelligence*, pages 125–155, 1989.
- [Rob65] L.G. Roberts. Matching perception of three-dimensional solids. In *Optical and Electro-optical Information Processing*, pages 159–197. Cambridge, MA: MIT Press, 1965.
- [Rob80] J.G. Robson. Neural images: The physiological basis of spatial vision. In C.S. Harris, editor, *Visual Coding and Adaptability*, pages 177–214. Hillsdale, NJ: Erlbaum, 1980.
- [Ros83] A. Rosenfeld. Parallel image processing using cellular arrays. *Computer*, 16:14–20, 1983.
- [Ros86] A. Rosenfeld. Pyramid algorithms for perceptual organization. *Behavior Research Methods, Instruments, & Computers*, 18:595–600, 1986.
- [Ros87] A. Rosenfeld. Recognizing unexpected objects: A proposed approach. In *Proc. of the DARPA Image Understanding Workshop*, pages 620–627, 1987.
- [RP91] R.A. Rensink and G. Provan. The analysis of resource-limited vision systems. In *Proc. Thirteenth Annual Conf. of the Cognitive Science Society*, pages 311–316, 1991.
- [Sal85] S.N. Salthe. *Evolving Hierarchical Systems*. New York: Columbia University Press, 1985.

- [SBG89] A. Sutter, J. Beck, and N. Graham. Contrast and spatial variables in texture segregation: Testing a simple spatial-frequency channels model. *Perception & Psychophysics*, 46:312–332, 1989.
- [Sch86] P.H. Schiller. The central visual system. *Vision Research*, 26:1351–1386, 1986.
- [She81] R.N. Shepard. Psychophysical complementarity. In M. Kubovy and J.R. Pomerantz, editors, *Perceptual Organization*, pages 279–341. Hillsdale, NJ: Erlbaum, 1981.
- [She83] G.M. Shepherd. *Neurobiology*. New York and Oxford: Oxford University Press, 1983.
- [Sim81] H.A. Simon. *The Sciences of the Artificial (2nd ed.)*. Cambridge, MA: MIT Press; New York: McGraw-Hill, 1981.
- [Smi90] M.A. Smith. Representations of geometrical and topological quantities in cellular automata. *Physica D*, 45:271–277, 1990.
- [Ste78] K.A. Stevens. Computation of locally parallel structure. *Biological Cybernetics*, 29:19–28, 1978.
- [Sto87] Q.F. Stout. Pyramid algorithms optimal for the worst case. In L. Uhr, editor, *Parallel Computer Vision*, pages 147–168. New York, NY: Academic, 1987.
- [Sto88] Q.F. Stout. Mapping vision algorithms onto parallel architectures. *Proc. IEEE*, 76:982–995, 1988.
- [Sug86] K. Sugihara. *Machine Interpretation of Line Drawings*. Cambridge, MA: MIT Press; New York: McGraw-Hill, 1986.
- [SV82] Y. Shiloach and U. Vishkin. An $O(\log n)$ parallel connectivity algorithm. *Journal of Algorithms*, 3:57–67, 1982.
- [Tan84] S.L. Tanimoto. Sorting, histogramming, and other statistical operations on a pyramid machine. In A. Rosenfeld, editor, *Multiresolution Image Processing and Analysis*, pages 136–145. New York and Berlin: Springer, 1984.
- [TB90] J.T. Todd and P. Bressan. The perception of 3-dimensional affine structure from minimal apparent motion sequences. *Perception & Psychophysics*, 48:419–430, 1990.
- [TG79] J.P. Thomas and J. Gille. Bandwidths of orientation channels in human vision. *J. Opt. Soc. Am. A*, 5:652–660, 1979.
- [TG88] A. Treisman and S. Gormican. Feature analysis in early vision: Evidence from search asymmetries. *Psychological Review*, 95:15–48, 1988.

- [Tho72] G.B. Thomas. *Calculus and Analytic Geometry*. Reading, MA : Addison Wesley, 1972.
- [TM87] T. Toffoli and N.H. Margolus. *Cellular Automata Machines: A New Environment for Modeling*. Cambridge, MA: MIT Press, 1987.
- [TM90] T. Toffoli and N.H. Margolus. Invertible cellular automata: A review. *Physica D*, 45:229–253, 1990.
- [Tow72] J.T. Townsend. Some results concerning the identifiability of parallel and serial processes. *British J. Math. Statist. Psychol.*, 25:168–199, 1972.
- [Tre82] A. Treisman. Perceptual grouping and attention in visual search for features and for objects. *J. of Experimental Psychology*, 8:194–214, 1982.
- [Tre88] A. Treisman. Features and objects: The fourteenth bartlett memorial lecture. *Quart. J. of Experimental Psychology*, 40A:201–237, 1988.
- [Tre91] A. Treisman. Search, similarity, and integration of features between and within dimensions. *Journal of Experimental Psychology: Human Perception and Performance*, 17:652–676, 1991.
- [TS90] A. Treisman and S. Sato. Conjunction search revisited. *Journal of Experimental Psychology: Human Perception and Performance*, 16:459–478, 1990.
- [Tso87] J.K. Tsotsos. A “complexity Level” analysis of vision. *International Journal of Computer Vision*, 1:346–355, 1987.
- [Tso90] J.K. Tsotsos. Analyzing Vision at the Complexity Level. *Behavioral and Brain Sciences*, 13, 1990.
- [TWW88] J.F. Traub, G.W. Wasilkowski, and H. Wozniakowski. *Information-Based Complexity*. New York, NY: Academic, 1988.
- [Uhr87] L. Uhr. Highly parallel, hierarchical, recognition cone perceptual structures. In L. Uhr, editor, *Parallel Computer Vision*, pages 249–292. New York, NY: Academic, 1987.
- [Ull84] S. Ullman. Visual routines. *Cognition*, 18:97–159, 1984.
- [Vol82] R. Vollmar. Some remarks about the efficiency of polyautomata. *International Journal of Theoretical Physics*, 21:1007–1015, 1982.
- [VP88] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333:364–367, 1988.

- [Wal72] D.L. Waltz. Generating semantic descriptions from drawings of scenes with shadows. Technical Report AI-TR-271, MIT, 1972.
- [Wal87] D. Walters. Selection of image primitives for general purpose visual processing. *Computer Graphics and Image Processing*, 37:261–298, 1987.
- [Wat87] A.B. Watson. Efficiency of a model image code. *J. Opt. Soc. Am. A*, 4:2401–2417, 1987.
- [Wat88] R.J. Watt. *Visual Processing*. Hillsdale, NJ: Erlbaum, 1988.
- [WB82] J.M. Woodhouse and H.B. Barlow. Spatial and temporal resolution and analysis. In H.B. Barlow and J.D. Mollon, editors, *The Senses*, pages 133–164. Cambridge: Cambridge University Press, 1982.
- [WH74] N. Weisstein and C.S. Harris. Visual detection of lines: An object-superiority effect. *Science*, 186:752–755, 1974.
- [Wil90] F. Wilkinson. Texture segmentation. In W.C. Stebbins and M.A. Berkley, editors, *Comparative Perception - Vol II: Complex Signals*, pages 125–156. New York: John Wiley and Sons, 1990.
- [WM78] N. Weisstein and W. Maguire. Computing the next step: Psychophysical measures of representation and interpretation. In A.R. Hansen and E.M. Riseman, editors, *Computer Vision Systems*, pages 243–260. New York, NY: Academic, 1978.
- [WM85] R.J. Watt and M.J. Morgan. A theory of the primitive spatial code in human vision. *Vis. Res.*, 25:1661–1674, 1985.
- [Woo81] R.J. Woodham. Analysing images of curved surfaces. *Artificial Intelligence*, 9:117–140, 1981.
- [ZSS83] S.W. Zucker, K.A. Stevens, and P. Sander. The relation between proximity and brightness similarity in dot patterns. *Perception & Psychophysics*, 34:513–522, 1983.
- [Zuc86] S. Zucker. Early orientation selection: Tangent fields and the dimensionality of their support. In A. Rosenfeld, editor, *Human and Machine Vision II*, pages 335–364. New York, NY: Academic, 1986.
- [Zuc87a] S. Zucker. The diversity of perceptual grouping. In M.A. Arbib and A.R. Hanson, editors, *Vision, Brain, and Co-operative Computation*, pages 231–261. Cambridge, MA: MIT Press, 1987.
- [Zuc87b] S. Zucker. Early vision. In S.C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 1131–1152. New York: John Wiley and Sons, 1987.