

**Coaxial Stereo and Scale-Based Matching**

by

**Itzhak Katz**

**Technical Report 85-13  
September 1985**

Laboratory for Computational Vision  
Department of Computer Science  
University of British Columbia  
Vancouver, B.C.  
Canada V6T 1W5



COAXIAL STEREO AND SCALE-BASED MATCHING

by

ITZHAK KATZ

B.Sc. in Computer Science

California State University at Northridge, 1979

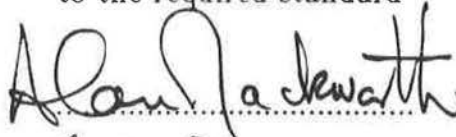

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF  
THE REQUIREMENTS FOR THE DEGREE OF  
MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES  
DEPARTMENT OF COMPUTER SCIENCE

We accept this thesis as conforming

to the required standard

  
.....  
  
.....

THE UNIVERSITY OF BRITISH COLUMBIA

August, 1985

© Itzhak Katz, 1985





## ABSTRACT

The past decade has seen a growing interest in *computer stereo vision*: the recovery of the depth map of a scene from two-dimensional images. The main problem of computer stereo is in establishing correspondence between features or regions in two or more images. This is referred to as *the correspondence problem*.

One way to reduce the difficulty of the above problem is to constrain the *camera modeling*. Conventional stereo systems use two or more cameras, which are positioned in space at a uniform distance from the scene. These systems use *epipolar geometry* for their camera modeling, in order to curb the search space to be one-dimensional - along *epipolar lines*.

Following Jain's approach, this thesis exploits a non-conventional camera modeling: the cameras are positioned in space one behind the other, such that their optical axes are collinear (hence the name *coaxial stereo*), and their distance apart is known. This approach complies with a simple case of epipolar geometry which further reduces the magnitude of the correspondence problem.

The displacement of the projection of a stationary point occurs along a *radial line*, and depends only on its spatial depth and the distance between the cameras. Thus, to simplify (significantly) the recovery of depth from *disparity*, complex logarithmic mapping is applied to the original images. The logarithmic part of the transformation introduces great distortion to the image's resolution. Therefore, to minimize this distortion, it is applied to the features used in the

matching process.

The search for matching features is conducted along radial lines. Following Mokhtarian and Mackworth's approach, a *scale-space image* is constructed for each radial line by smoothing its intensity profile with a *Gaussian filter*, and finding *zero-crossings* in the second derivative at varying scale levels. Scale-space images of corresponding radial lines are then matched, based on a modified uniform cost algorithm. The matching algorithm is written with generality in mind. As a consequence, it can be easily adopted to other stereoscopic systems.

Some new results on the structure of scale-space images of one dimensional functions are presented.

## TABLE OF CONTENTS

1	Preface .....	1
1.1	Introduction .....	1
1.2	Overview .....	2
2	A review of computational stereo .....	5
2.1	Introduction .....	5
2.2	Image acquisition .....	7
2.3	Camera modeling .....	8
2.4	Feature acquisition .....	10
2.5	Matching .....	12
2.6	Determining depth .....	14
2.7	Motion .....	14
2.8	Survey .....	15
2.8.1	Stereoscopic systems .....	15
2.8.1.1	Marr et al .....	15
2.8.1.2	Mayhey and Frisby .....	18
2.8.1.3	Moravec .....	18
2.8.1.4	Bernard and Thompson .....	19
2.8.1.5	Hannah .....	19
2.8.1.6	Gennery .....	20
2.8.1.7	Henderson, Miller and Grosch .....	20
2.8.1.8	Arnold .....	20
2.8.1.9	Baker .....	21
2.8.1.10	Ohta and Kanade .....	22
2.8.1.11	Lowrie and Crowley .....	22
2.8.2	Motion systems .....	23
2.8.2.1	Regan, Beverley and Cynader .....	23
2.8.2.2	Lee .....	23
2.8.2.3	Jain .....	24
3	The geometrical aspects of coaxial stereo .....	25
3.1	Camera configuration .....	25
3.2	Complex Logarithmic mapping .....	26
3.2.1	Introduction .....	26
3.2.2	The mathematical aspects of CLM .....	27
3.2.3	Geometrical features of CLM .....	31
3.2.4	Practical deficiencies of CLM .....	32
4	Scale-space images of one dimensional functions .....	34
4.1	Introduction .....	34
4.1.1	The scale-space image concept .....	34

4.1.2	Some new observations .....	35
4.1.3	Approach and assumptions .....	36
4.2	On the generation of open contours .....	37
4.2.1	Case 1 .....	37
4.2.1.1	Case 1.1 .....	38
4.2.1.2	Case 1.2 .....	40
4.2.1	Case 2 .....	41
4.2.2.1	Case 2.1 .....	43
4.2.2.2	Case 2.2 .....	43
4.3	Crossing contours .....	45
4.3.1	The even-double step signal .....	45
4.3.1.1	Multi step signals which contain one or more EDSs .....	50
4.3.2	Wave signals .....	50
4.3.3	Other cases .....	52
4.4	Binary scale-space images .....	54
4.5	Conclusion .....	57
5	Scale-space matching .....	59
5.1	Introduction .....	59
5.2	Approach considerations .....	59
5.3	The difficulties in establishing correspondence .....	60
5.4	The matching algorithm .....	64
5.4.1	Establishing the initial matching nodes .....	64
5.4.2	Expanding a node .....	66
5.4.3	Attending to skipped open contours .....	67
6	Implementation .....	68
6.1	Computer system .....	68
6.2	A collection of programs .....	68
6.3	Programming languages .....	68
6.4	System description .....	69
6.4.1	System flow .....	69
6.4.2	Image transformation .....	69
6.4.3	Constructing SSIs .....	73
6.4.3.1	Program scale-space .....	73
6.4.3.2	Filling gaps in the SSI .....	76
6.4.3.3	Building a representation for a SSI .....	76
6.4.3.4	Constructing a two-tone binary SSI .....	78
6.4.4	Matching .....	78
6.4.5	Computing depth .....	80
7	Experiments .....	81
7.1	Introduction .....	81
7.2	Two dimensional scene .....	81

7.2.1	Experiment #1 .....	83
7.2.2	Error analysis .....	85
7.2.3	Experiment #2 .....	86
7.2.4	Experiment #3 .....	87
7.2.5	Conclusions .....	88
7.3	Three dimensional scenes .....	88
7.3.1	Round cake scene .....	88
7.3.2	Inverted square cake scene .....	91
7.3.3	Occlusion and loss of information .....	95
8	Extensions and conclusions .....	100
8.1	Extensions to this project .....	100
8.1.1	A different matching algorithm .....	100
8.1.2	Application to other stereo systems .....	101
8.1.3	Multiple input images .....	101
8.1.4	Occlusion .....	101
8.1.5	Moving contours .....	101
8.1.6	Two-tone scale-space images .....	102
8.2	Summary and concluding remarks .....	102
	References .....	105
	Appendix I Step signals .....	112
	Appendix II The Gaussian filter and its derivatives .....	114
	Appendix III Proofs concerning open contours .....	115
	Appendix IV Synthetic image construction .....	128
	Appendix V Manual pages of a few programs in the public library .....	130



## LIST OF FIGURES

Figure 2.1	The geometry of conventional binocular stereo .....	5
Figure 2.2	Epipolar geometry .....	9
Figure 2.3	Epipolar lines .....	9
Figure 3.1	The geometry of coaxial stereo .....	25
Figure 3.2	Rectilinear optical flow .....	26
Figure 3.3	Complex image plane .....	28
Figure 3.4	3D projection space .....	29
Figure 3.5	The exponential mapping .....	31
Figure 3.6	Cartesian to polar mapping .....	32
Figure 4.1	The Dirac delta function and its scale-space image .....	39
Figure 4.2	A pulse function and its scale-space image .....	39
Figure 4.3	A single cycle wave function and its scale-space image .....	42
Figure 4.4	An asymmetric wave function and its scale-space image .....	42
Figure 4.5	A single step function and its scale-space image .....	44
Figure 4.6	Double step functions of uneven steps and their scale-space image .....	44
Figure 4.7	A double step function and its scale-space image .....	46
Figure 4.8	The scale-space image construction stages of a double step func- tion .....	47
Figure 4.9	The smoothness degree of a convolved double step function .....	49
Figure 4.10	Crossing closed contours .....	49
Figure 4.11	A function composed of two double step signals and its scale- space image .....	51
Figure 4.12	A wave function of five cycles and its scale-space image .....	51
Figure 4.13	A function composed of two even double cycle wave signals and its scale-space image .....	53
Figure 4.14	A function composed of two pulses of opposite signs and its scale-space image .....	53
Figure 4.15	A multi-pulse function of intermittent sign change and its scale- space image .....	55
Figure 4.16	A signal composed of two pulses of various energies and their scale-space image .....	55
Figure 4.17	A rabbit-ears contour that crosses a closed contour .....	56
Figure 4.18	A binary scale-space image .....	56
Figure 5.1	The sensitivity of scale-space images to changes in intensity and texture density .....	62
Figure 5.2	Movement of an open contour .....	63
Figure 6.1	System flow .....	70
Figure 6.2	Cartesian-to-polar mapping .....	70

Figure 6.3	CLM versus polar transformations .....	72
Figure 6.4	Jain's interpolation scheme .....	72
Figure 7.1	The 2D scene images in cartesian space .....	82
Figure 7.2	The 2D scene images in polar space .....	82
Figure 7.3	The zero-crossings extracted from the closer image .....	84
Figure 7.4	The 2D scene depth map .....	84
Figure 7.5	A typical set of scale-space images .....	84
Figure 7.6	A round cake scene .....	89
Figure 7.7	The round cake scene images in cartesian space .....	89
Figure 7.8	The round cake scene images in polar space .....	90
Figure 7.9	A typical set of scale-space images .....	90
Figure 7.10	The round cake scene depth map .....	90
Figure 7.11	An inverted square cake scene .....	92
Figure 7.12	The square cake scene images in cartesian space .....	92
Figure 7.13	The square cake scene images in polar space .....	94
Figure 7.14	A set of scale-space images of radial line $0^\circ$ .....	94
Figure 7.15	The square cake scene depth map .....	94
Figure 7.16	A ring and cylinder scene .....	96
Figure 7.17	The ring and cylinder scene in cartesian space .....	96
Figure 7.18	The ring and cylinder scene in polar space .....	97
Figure 7.19	A set of scale-space images .....	99
Figure A1.1	A single step signal .....	113
Figure A1.2	A multi step signal .....	113
Figure A3.1	A convolved function with three fluctuation points .....	120
Figure A3.2	A convolved function with four fluctuation points .....	121
Figure A3.3	A function that crosses the zero twice .....	124
Figure A3.4	A convolved function with five fluctuation points .....	125
Figure A3.5	A function that crosses the zero three times .....	126
Figure A4.1	Standard thin lens geometry .....	128



*LIST OF TABLES*

Table 7.1	Scale-space matching results .....	83
Table 7.2	Scale-space matching results .....	91
Table 7.3	Scale-space matching results .....	93



## Acknowledgements

First I would like to thank my supervisor, Dr. Alan Mackworth, for his continued support and for always willing to listen, advise and share his tremendous knowledge. I truly feel fortunate to have had him for a supervisor.

I also want to thank Dr. Bob Woodham for his helpful comments on an earlier draft of this thesis.

I wish to thank Farzin Mokhtarian and Jim Little for their support and advice and hours of useful discussions, and to Mark Majka for his technical advice in using the system.

Last but not least, I want to thank my wife, Dora, for her support, love and understanding.



# CHAPTER 1

## PREFACE

### 1.1. INTRODUCTION

*Vision* is the process, as performed by human beings, in which visible light is passively sensed to produce an understanding of the physical environment. *Computational vision* is a field in *artificial intelligence*, which pursues this goal of constructing an explicit and meaningful three dimensional description of objects from their projection onto retinal images. A major thrust towards reaching this goal is provided by *computational stereo*. The objective of computational stereo is to determine the three dimensional structure of the physical environment from two or more retinal images obtained at a given spatial interval. Monocular approaches to computing depth from images tend to use photometric or statistical assumptions. Stereo, on the other hand, uses mostly direct image measurements, and hence, its attractiveness.

Some of the main applications of computational stereo are in the areas of interpreting aerial images for cartography and surveillance, robotics for assembly line tasks as well as for autonomous vehicle navigation, and systems simulating biological stereopsis systems pursuing the enhancement of their understanding.

The key problem in computational stereo is *the correspondence problem*, which is the matching of corresponding elements in the given images. One way to ease the problem is to constrain the camera configuration such that the

search space is reduced to one dimension. Another approach is to match acquired features in groups rather than individually. In existing systems, these groups of features represent edges in the scene. This approach is justified by *the continuity assumption* about the physical world: objects are cohesive and usually opaque.

In this project, we propose a different grouping approach based on filtering the images in a range of different spatial frequencies, which are quantized in a *scale space*.

## 1.2. OVERVIEW OF THIS DOCUMENT

The first part of chapter 2 introduces the reader to the world of stereo. The notion of stereoscopic disparity is defined. Each phase in a five fold procedure to determine depth from stereo is described. This procedure characterizes all stereoscopic systems. The second part of this chapter provides the reader with a survey of existing stereoscopic and related motion systems.

Chapter 3 describes the principles and geometrical aspects of *coaxial stereo*. We define coaxial disparity and the notion of *focus-of-expansion*, and explain the optical flow generated by *motion-in-depth*. The notion of *complex logarithmic space* is described and its advantages and disadvantages discussed.

Chapter 4 is devoted to scale-space images of one dimensional functions. A scale-space image of a one dimensional intensity profile is obtained by convolving it with the second derivative of a *Gaussian filter* at varying levels of detail, and extracting *zero-crossings* at each level. A theoretical analysis of such scale-space images is presented. The analysis brings to light some new observations about open contours. We show that such functions contain in

their scale-space image between one and five open contours. We prove that functions with finite energy contain exactly two open contours which are asymptotic to two imaginary straight lines diverging symmetrically in a  $45^\circ$  angle. We prove the validity of incidents which violate claims that a scale-space image has a hierarchical structure, i.e. that no contours cross each other. We will show that closed contours may be intersected by an open contour or by another closed contour.

Chapter 5 discusses the difficulties in establishing correspondence between the two scale-space images. It describes the matching algorithm which is based on the  $A^*$  algorithm. This algorithm matches contours in the scale-space image in a coarse to fine order and uses height to find the order in which contours will be matched. This approach enables this algorithm to be easily adopted to other stereoscopic systems.

Chapter 6 explains the implementation of the theory described in the preceding chapters. The computational part of the system was implemented in C. That includes the image transformation, building the scale-space images and representing them. The matching algorithm and depth computations have been implemented in Franz LISP.

Chapter 7 describes a series of experiments conducted with synthetic images. A two dimensional scene is constructed and used for error analysis of the system. A depth map is computed for three dimensional scenes. An example, demonstrating how occlusion and loss of information influence a scale-space image, is given.

Chapter 8 gives some conclusive remarks about the various subject matters covered in this thesis. A look at some follow-up research avenues

concludes this project.



## CHAPTER 2

### A REVIEW OF COMPUTATIONAL STEREO

#### 2.1. INTRODUCTION

The basis for computational stereo is in the *disparity* measurements. The notion of binocular disparity was first introduced by Sir Charles Wheatstone to the Royal Society of London, in 1832. He invented an apparatus, he called *the stereoscope* (which is much the same today), through which pairs of stereograms were easily fused into a single image that appeared to the viewer as extending in depth. Disparity is defined as the angular difference (or displacement in position) of the projection of a scene element onto the two retinal images, relative to some fixed point. It varies for each point in the scene depending on the spatial position of each scene element and the camera configuration.

In human binocular stereo, for example, disparities arise because the eyes converge slightly to *fixate* on some point *a* in the scene (fig. 2.1), called the

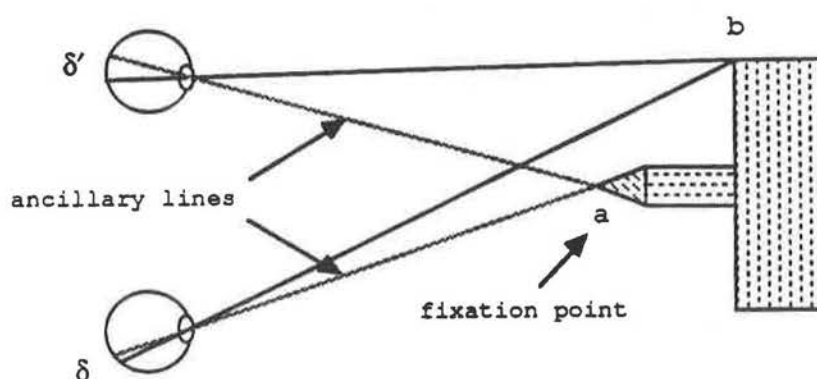


Figure 2.1: the geometry of conventional stereo.

*fixation point*. The position of each eye is referred to as a *vantage point* [Gibson 1950]. The viewing axis is termed *ancillary line* [Pirenne 1970]. It always casts a point at the center of the retinal image. A neighboring point  $b$  in the scene is projected onto the retina at some distance  $\delta$  from the center of vision. This distance is different for each eye, a difference to which we refer as disparity.

Determining depth from stereo is a five step procedure:

1. **image acquisition** - type of images used.
2. **camera modeling** - the spatial geometry of the cameras.
3. **feature acquisition** - the projection of one or more scene elements must be selected from one image.
4. **matching** - the corresponding image element(s) in the other retinal image must be identified.
5. **determining depth** - the distance between the camera and scene element(s) is calculated from the disparity measured between two corresponding image elements.

Obtaining a match between two corresponding image elements is by far, computationally, the most difficult task in computational stereo.

The stereo paradigm can be viewed as a case in the concept of *motion* or *optical flow*. In conventional stereo, two images are obtained simultaneously by two cameras of known displacement. In motion, a single camera records a sequence of images, while moving in an arbitrary path.

## 2.2. IMAGE ACQUISITION

There are three principal considerations involved in image acquisition:

- type of scene.
- image resolution.
- camera modeling.

A scene can be classified into two types: *natural* scenes and *man-made* scenes. Images of natural scenes reflect natural terrain and cover: land, water, rolling mountains, rivers, trees, snow and so on. Man-made scenes refer to cities, buildings, streets, roads, machines and their parts and so on. The difficulty that often arises with natural scenes, is the lack of details and homogeneity. Such are images of forests, oceans, prairies and others. The difficulty introduced by man-made scenes, is high repetition and occlusions. Such is the case for crisscross streets and roads, buildings and building windows, and so on.

The camera configuration and the resolution of the images depend on the application. The cameras may be positioned at various distances from each other. They may be rolled, tilted or panned, all with respect to their optical axis [Bernnard and Fischler, 1982]. If they are rolled or tilted, it is normally in the same direction and by the same amount. Panning is employed either inwards or outwards by the same amount.

Aerial images involve a variety of terrain types and are usually of low resolution. That is, each pixel represents a relatively large area in the scene. The cameras are set parallel to one another.

In applications in which the human visual system is model, a wide range of resolutions is used. Grimson [1981] used in his system highly repetitive

random-dot stereograms, which offer no monocular clues.

In robotics, high resolution images are used. In system for autonomous vehicle navigation, the cameras are either parallel or panned inwards.

### 2.3. CAMERA MODELING

The key problem in computational stereo is the correspondence problem. It involves matching the projections of scene elements upon two or more image planes. Constraining the camera configuration can drastically reduce the magnitude of the problem. In its worst case, the search space has a two dimensional (2D) neighborhood. The search space can be reduced to one dimension by using epipolar geometry [Gimel'farb et al 1972, Mori et al 1973, Henderson et al 1979, Moravec 1980, Gennery 1980, Arnold 1983, Baker 1982, Lowrie 1984, Bernard and Fischler 1982].

Fig. 2.2 depicts epipolar geometry as applied in a conventional binocular stereo configuration. The line connecting the focal points of the cameras is called *the stereo baseline*. Any plane containing the stereo baseline is an *epipolar plane*. The intersection of an epipolar plane with an image plane is called *epipolar line*. In a camera configuration which complies with epipolar geometry, the search for corresponding points needs only be done along (known) corresponding epipolar lines, and hence, it is reduced to be 1D.

If the cameras are panned inwards, as is the case for human eyes, the epipolar lines in each image are not parallel, but fanning outwards [Lowrie 1984] (fig. 2.3b). If the two image planes are coplanar, the images are said to be *rectified* [Baker and Binford 1981]: all epipolar lines are parallel (fig. 2.3a) and coincide with the scan line of the images.

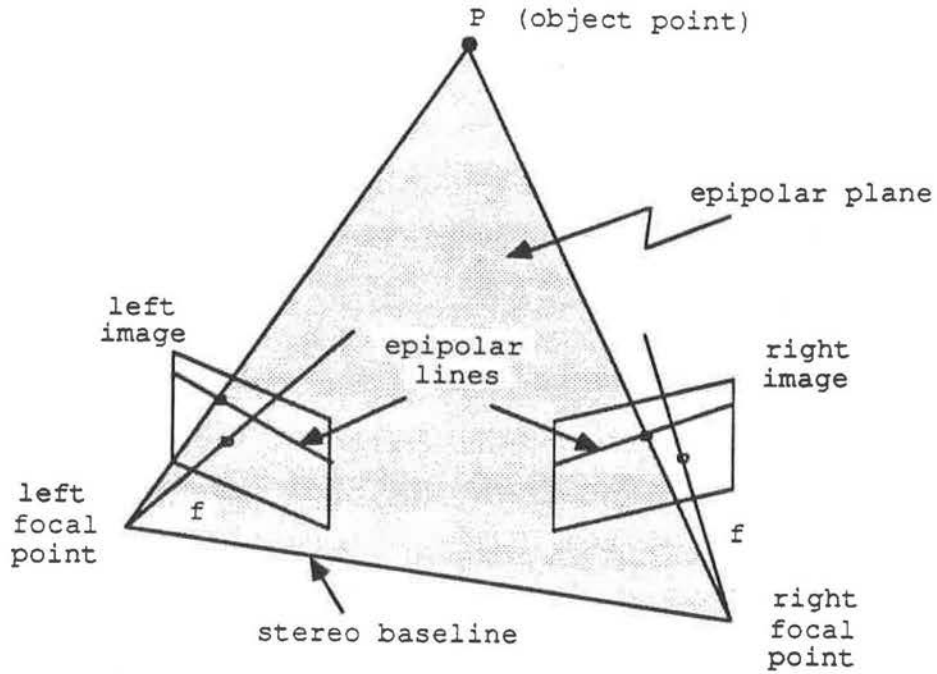


Figure 2.2: epipolar geometry of binocular stereo

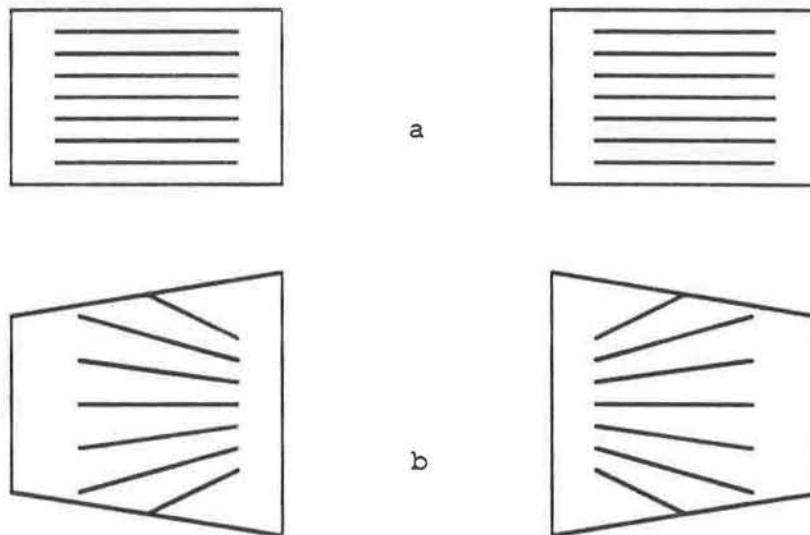


Figure 2.3: epipolar lines with parallel areas (a) and with panned cameras (b).

In a non-conventional stereo configuration, as suggested by O'Brien and Jain [1984], the two cameras are positioned in space such that their optical axes are collinear. Hence the name *coaxial stereo*. The system developed in this project, proposes a similar geometry. We shall elaborate on this configuration in chapter 3.

One should notice the potential for parallel processing, when epipolar geometry is employed, as there is no interline dependence.

#### 2.4. FEATURE ACQUISITION

Featureless areas of nearly homogeneous intensity cannot be matched with confidence. Thus, some distinguishable features must be present in both images, in order to be able to match them. The more such features exist, the more accurate will the yielding depth map be.

It is assumed that a sharp gradient in intensity corresponds to physically significant events in the scene, and it is those events which are of interest. However, it should be noted that some intensity changes may have no physical significance.

The properties of a feature depend mostly on local monocular intensity patterns. They can be summarized as follows:

- **size:** spatial frequency.
- **contrast:** in intensity.
- **semantic content:** what does it represent.
- **density:** the more features the more accurate is the depth map.
- **ease to measure:** less computation time.

- **distinguishability:** can be easily matched.

In general, an image function is represented by a gray-level array. A sharp intensity change in the gray-level array gives rise to an extremum, when the image function is convolved with the first derivative of a Gaussian distribution function. In the second derivative, it corresponds to a *zero-crossing* (ZC): a location where the second derivative crosses zero as it falls from positive values to negative ones, or rises from negative to positive. An extensive discussion on the properties of such features can be found in Hildreth [1980], Marr [1982], Koenderink and van Doorn [1980], Yuille and Poggio [1983a, 1983b], Yuille [1984].

Basically, there are two types of features which are used for matching: *point-like* and *edge-like*. Point-like features are nondirectional, and hence, can be detected in any direction. Edge-like features, which can be constructed by connected ZCs, are directional, and therefore will not be detected if the direction of the search coincides with the direction of the edge. Edge-like features require less matching activity. However, the camera model needs to be known and the features must be oriented across the epipolar lines.

One may also use *semantic* features, which are based on physical properties or spatial geometry. Such features are occluding edges, vertices of linear structures and prominent surface markings. They are not widely used as they are difficult to determine.

The features selected for any particular system is closely related to the matching scheme used, which is discussed next.

## 2.5. MATCHING

We have seen that the key problem of computational stereo is the correspondence problem. It is the problem of matching elements in the two images, which correspond to the same location in the scene, without the recognition of objects or their parts.

There may be several elements in one image which could correspond to a particular element in the other image. Most of those possible matches do not represent corresponding elements, and are called *false targets* [Marr and Poggio 1979, Marr 1982] or *ghosts* [Mayhew and Frisby 1981]. They represent the ambiguity in the correspondence between the two images.

Two implicit assumptions about the visual world are made to constrain the correspondence problem:

**Uniqueness** - a given point on a physical surface has only one position in space at any one time. Thus, each element in either image has a unique disparity and can match only one element in the other image.

**Continuity and opacity** - physical objects are cohesive and usually opaque. Thus, the disparity varies smoothly over the image (variation in depth are generally smooth in the sense that the surface variations are very small compared with their distance from the camera), except at object boundaries.

The level of difficulty in applying the uniqueness and continuity constraints, is determined by the density of the features to be matched and the disparity range over which a match is sought. The greater this range the greater the number of false targets, and the greater the density the greater the potential for false targets. Thus, to avoid false targets, one should reduce



either the disparity range or density. But then, it is desired both to be able to compute depth over a large disparity range and to obtain a fine detailed depth map of the scene.

There are two main techniques which are used to maintain both high density and a wide disparity range: *relaxation* methods and *coarse-to-fine* methods. The relaxation approach uses the 3D continuity assumption, which is applied to disparities in the images. The coarse-to-fine approach finds coarse disparities at low resolution (low density and wide range) but with low accuracy. These rough disparities are then used to guide the search for more precise disparities at finer resolution (high density and narrow range). Clark and Lawrence [1984] have implemented a coarse-to-fine process in hardware.

Existing stereo systems employ two main approaches to matching: *area matching* and *feature matching*. Area matching is the matching of regularly sized neighborhood around a pixel. It is justified by the continuity assumption. The matching proceeds by selecting an area in one image and correlating it with the other image using local information. The *interest operator* [Moravec 1980] is widely used in area matching. This operator detects regions in an image which are of high variance in four directions (horizontal, vertical and diagonals), over a 3 by 3 pixel neighborhood.

Feature matching is the matching of point-like or edge-like features, usually in the form of ZCs or peaks.

The survey in § 2.8 places an emphasis on the matching method used in each system.

## 2.6. DETERMINING DEPTH

Once the correspondence problem has been solved, the disparity between each pair of matching image elements can be measured. Depending on the camera geometry, the depth of each scene element, that gave rise to a particular disparity, can be calculated using triangulation.

An interpolation step, based on the uniqueness and continuity assumptions, can be applied to interpret occluding features and dissolve final ambiguities left from the matching process.

## 2.7. MOTION

*Motion* can be viewed as a generalization of stereo. In motion analysis, a sequence of images are analyzed. In stereo analysis, two or more "snap shots" of that sequence, with a known interval, are considered.

Many studies have been conducted in recent years on the subject of motion, and there seems to be a confusing interchange in the usage of the terms *motion* and *optical flow*. We would like to take this opportunity and make a distinction between the two. An optical flow field is the measurement of the continuous change in the brightness of an image. Usually, it is a consequence of a moving object or a translating observer. But it can also be generated by other means - a moving light source, for example. A motion field is a 2D change in the retinal position of the projection of a point in the scene. It is caused by either motion of that scenic point, or translation of the observer, or both.

We will inspect some systems that are of relevance to this project: sys-

tems concerning *motion-in-depth*. A more general review on motion can be found in Ballard and Brown [1982] and Ullman [1981].

It has been observed as early as one hundred and fifty years ago [Wheatstone, 1838] that by changing the size of an object, an impression is produced that the object is moving in depth. Hence, *motion-in-depth* is associated with an environment in which a camera moves along its optical axis. The change in size of the perceived image is directly related to the instantaneous distance separating the object from the camera. *Motion-in-depth* has been referred to by various names. We chose to use the term *motion-in-depth* [Ullman 1979, Regan et al 1979a, 1979b]. A more common name used is *looming* [Schiff 1965, Gibson 1979, Marr and Ullman 1979, Marr 1982, Regan and Beverley 1978, Richards 1983]. Other terms used are *time-to-collision* [Lee 1974, 1976, Ullman 1981, Ballard and Brown 1982] and *rectilinear motion* [Lee 1980]. O'Brien and Jain [1984] referred to it as *axial motion stereo*.

Both stereo and motion systems have some deficiencies when used to recover a 3D structure. Richards [1983] showed that when the two methods are combined, only two stereo images and three points are needed to ensure the recovery of the 3D structure.

## **2.8. SURVEY**

### **2.8.1. Stereoscopic Systems**

#### **2.8.1.1. Marr et al (MIT AI Lab.)**

A group of researchers led by David Marr, have developed two systems to model human stereopsis, the first of which they rejected because of

neurophysiological reasons.

Bela Julesz at the AT&T Bell Laboratories, showed [1971] that two retinal images of random dot stereograms are compared by the human visual system point-by-point, before referring to it as a whole. This implies that the human visual system can fuse stereoscopic disparity into 3D information, with no monocular clues. For that reason, the MIT group chose to use random dot stereograms as test data for their systems.

In their first system [Marr and Poggio 1976], a *cooperative* computational model was developed. It is a relaxation scheme which employs the uniqueness and continuity constraints. A point is assigned multiple disparities which inhibit each other. Local collections of similar disparities, on the other hand, excite one another.

Although the system was successful, it was rejected as a model for human stereopsis. Instead, a system based on varying spatial scale was proposed [Marr and Poggio 1977, 1979, Marr 1982], implemented [Grimson 1981] and refined [Poggio 1984, Grimson 1984]. In this later approach, they suggested that the human visual system uses a coarse-to-fine scheme to detect and fuse disparities, and is accomplished by a convergent eye movement. Matches established at a large spatial scale were used to guide the matching analysis at a finer scale, resulting in less false targets and more accuracy.

The convolution employed a *2D Laplacian of a Gaussian* mask. It is a distribution function shaped like a Mexican hat, which assigns weights to the neighborhood of each pixel in the image: as the distance increases the importance of the neighboring pixel decreases. It is (approximately) a band-limited operator which blurs the image by an amount dependent on its scale, given by

the size of its central diameter. The convolution was performed on a LISP machine, speeded up by special hardware designed for that project.

It was believed, at that time, that the human visual system uses four spatial channels, which are roughly one octave apart. Hence, their coarse-to-fine analysis used four different scales with a size ratio of approximately 1:2.

The information recorded about a ZC includes its location, size and orientation of the segment in which it was contained. These descriptions were computed for each of the eight images. In Grimson's implementation, the orientation at a point on a ZC segment was computed as the direction of the gradient of the convolution values across that segment, and recorded in increments of  $30^\circ$ . Horizontally oriented segments of the ZC contours were ignored because they coincided with the scan line, and hence, did not have a well defined disparity.

Marr and Poggio [1979] have shown statistically that given a ZC, the probability of another ZC of the same sign occurring within half the central diameter is less than 5% (hence the usage of successive filters which are one octave apart).

The matching at each scale proceeded independently. After a ZC was selected in one image, its corresponding ZC in the other image was searched for. The search is conducted in a region with a disparity range the size of the central diameter of the mask, and centered at the expected disparity obtained at the previous, coarser filtering step (the initial disparity may be taken as zero).

If only one match with the same sign and approximate orientation was found, it was taken to be a valid match. If more than one match was found.

the continuity constraint was applied comparing the disparity of each candidate match to the dominant disparity in the immediate neighborhood. If still more than one match prevailed, they were carried over to be disambiguated at a finer scale. If no match was found, the search area was doubled reducing the probability of obtaining a good match to 50%.

#### **2.8.1.2. Mayhew and Frisby (University of Sheffield)**

This team, for the most part, followed up on the MIT line of work. They argued [1981] that, in human stereopsis, the activity at each scale is not independent, and suggested that cross-channel activity is taking place. They also claimed that features used for the matching process include peaks (maxima in the second derivative) as well as ZCs.

The rest of the systems described below disassociate themselves from human stereopsis, allowing the exploration of a variety of camera configurations.

#### **2.8.1.3. Moravec (Stanford University)**

Moravec's study [1980] is concerned with autonomous vehicle navigation. His aim was not to build a depth map, but to obtain information about the vehicle movement.

A sequence of images was obtained in increments of one meter, which were then analyzed to deduce the vehicle movement. He employed in his work the *interest operator* which he had developed earlier and is widely used in the field. Features were selected and then searched, using his *binary correlator*, in a coarse-to-fine process. A low resolution correlation determined a best match

location for a selected feature, and reduced the area in the second image that must be searched at the next higher resolution.

The correlation, which was repeated for each resolution level, was reported to have had about 10% error rate. To reduce false matches, a sequence of 9 equally spaced images was used. A feature in one image was located in the other 8 views. The distances computed from the 36 possible image pairings, at each resolution, were statistically analyzed to provide the best estimate.

#### **2.8.1.4. Bernard and Thompson (University of Minnesota)**

They proposed [1980] a relaxation technique to solve the correspondence problem. Moravec's interest operator was used to select point-like features in each image. Each such point is assigned a list of possible matches based on a threshold value representing a maximal disparity. Each possible match in each list is given a weight probability, indicating the "goodness" of the match. The probabilities are then refined based on the consistency of the disparity values of its neighbors. At each iteration, the probabilities below some threshold value are discarded, while others above another threshold are accepted as valid. This process is repeated until a steady state is reached, which is reported to be around ten iterations.

#### **2.8.1.5. Hannah (Stanford University, Lockheed)**

Hannah [1974,1980] has developed a similar system to that of Moravec. He modified the interest operator to consider ratios of variance in the four directions, as well as intensity values over larger area. This modification ren-

dered a better selection of features to be matched.

#### **2.8.1.6. Gennery (Stanford University)**

In his autonomous vehicle navigation system [1980], Gennery employed cross-correlation, as well as Moravec's interest operator, to deduce a depth map. The search for corresponding matches proceeded from left to right along an epipolar line, using local context of previous matches to suggest tentative match sites.

#### **2.8.1.7. Henderson, Miller and Grosch (Control Data Corporation)**

This group developed [1979] the *broken segment matcher* to construct 3D modeling of man-made scenes for aerial imagery. It combined edge and area based techniques, with the edges serving to bound regions in which correlation is based on image intensities.

The matching is based on intersections of edges and epipolar lines in the two images. Edge match information is propagated from line to line and edited automatically as some edges end and others begin.

The system assumes rectilinear structures, implying that all edges are straight and surfaces have one of three orthogonal orientations. In a later report produced by CDC [DeGryse and Panton 1980], the system was described to be "noisy", "fragmented" and "unstable".

#### **2.8.1.8. Arnold (Stanford University)**

Arnold proposed [1978, 1983] a feature based edge matching scheme. Edges are matched based on orientation, side intensities and the continuity of



two adjacent edges. In its early version, the system used unlinked edge elements (edgels). It succeeded in correctly correlating about 90% of the edgels in the image. In its latest version, this was extended to link the edgels. It used a Viterbi dynamic programming algorithm [Forney, 1973], which employs a finite-state Markov process.

The camera geometry used, in order to constraint the search space, is epipolar geometry with coplanar image planes (see § 2.3).

#### **2.8.1.9. Baker (Stanford University)**

Baker has implemented [Baker and Binford 1981, Baker 1982] a stereo system which combines several ideas used in other systems, and employs an edge matching scheme.

The camera modeling uses epipolar geometry with coplanar image planes. The images are convolved with a *low-pass* linear filter. The search technique is based on a Viterbi algorithm and engages a coarse-to-fine process in which resolution is doubled at successive steps.

An edge is defined to have a *position, contrast, slope* and *intensity* to either side (splitting the edge to a left and a right halves). *Links* are kept to nearest neighbors below, above and to the sides. Edges are matched by comparing their slope, side intensities, relative disparity obtained at a lower resolution, and interval compression implied by the correspondence.

The continuity constraint is used as a *connectivity* constraint in a *cooperative* procedure across epipolar lines, which removes edges that violate surface continuity. Another edge matching process is applied in an attempt to match unassigned edges, which are bound by pairs of matched edges. A final

intensity correlation produces the desired depth map.

#### **2.8.1.10. Ohta and Kanade (CMU)**

This system [1983] improves on Baker's [1982] edge based system by interacting the cooperative procedure with the matching process. This implied a 3D search space, consisting of the row (scanline) and the column position in each image.

First, isolated edges are detected from the image intensity profile. Then, the isolated edges form connected edges. An *inter-scanline* search is used to match the connected edges and to maintain consistency between the rows.

The system is reported to be effective in matching the isolated edges, but slows down considerably as the number of edges increases.

#### **2.8.1.11. Lowrie and Crowley (CMU)**

This system [Lowrie 1984] uses coplanar epipolar geometry as well. The convolution employs a set of 1D band-pass filters, similar to Marr's DOG filter [Marr and Hildreth 1980], which they named *the 1D difference of low-pass (DOLP) transform* [Crowley and Stern 1982].

They use peaks in the second derivative, rather than ZCs, as features for the matching process. This choice was made for the sake of matching features of surfaces rather than those of edges. The positive and negative peaks, at multiple resolutions, were connected to form *edge-paths*.

The search and matching proceed similarly to that of Ohta and Kanade's [1983] system.

## 2.8.2. Motion Systems

### 2.8.2.1. Regan, Beverley and Cynader (Dalhousie University)

Psychologists gave the perception of motion-in-depth early attention [Wheatstone 1838, Gibson 1950, 1966, 1979, Schiff 1965], but the most extensive work was conducted recently by Regan et al. They produced a series of reports [1973, 1978, 1979a, 1979b] in which they suggest the existence of a motion-in-depth channel, which is distinct from the sideways motion channel and from the position in depth channel. They brought forward some psychological evidence that this channel is fed by two types of channels: *stereoscopic motion* channels and *changing size* channels. The stereoscopic motion channels are fed by the relative velocities between the left and right images of a moving object. The changing size channels are channels which are distinct from the channels sensitive to static size.

They noted that changing size is a monocular cue and is available to one eye alone. For that reason pilots and baseball players, who have lost vision in one eye, can continue their activities effectively.

Schwartz [1980b] argued against these findings, suggesting that the retinoscopic *complex logarithmic mapping* (CLM) of the striate cortex (area 17) dismisses the need for channels sensitive to changes in size. This subject matter will be elaborated on in chapter 3.

### 2.8.2.2. Lee (university of Edinburgh)

Lee's work [1974, 1976, 1980] is concerned with the time it takes for an approaching observer to collide with an object positioned on his path, as in the case of a driver who is attempting to brake.

He argued that the absolute velocity at which the camera approached an object and the absolute distance between them, cannot be recovered from the velocity of the optical flow alone. What can be computed is their ratio, which in turn yields the time to collision.

### **2.8.2.3. Jain (General motors, Ann Harbor University)**

Jain [1983a] put forward algorithms to directly compute the focus of expansion (FOE). He employed them to assist in the analysis of optical flow, and separate stationary objects from dynamic ones [1984]. He suggested [1983b] to apply CLM to the images with their FOE placed at the center. This approach is based largely on Schwartz' studies [1977, 1980a, 1981, 1982] and Weiman and Chaikin's investigation [1979] of this transformation.

In a recent publication, O'Brien and Jain [1984] proposed a stereo system similar to the one developed in this project. In that system, the camera modeling consists of a camera moving and gazing along its optical axis, with two (or more) frames being recorded at given spatial intervals. The environment in which the camera moves may consist of both stationary and dynamic objects (see chapter 3).

The search space is 1D advancing radially from the FOE outwards. O'Brien and Jain claimed that as a consequence of the camera modeling and the CLM: "correspondence of the points from one stereo picture map to the other is extremely simple". Although the search space is 1D and directional, matches still must to be obtained. The problem is simplified, but definitely not eliminated.

## CHAPTER 3

### THE GEOMETRICAL ASPECTS OF COAXIAL STEREO

#### 3.1. CAMERA CONFIGURATION

The camera configuration of coaxial stereo (fig. 3.1) consists of two cameras which are situated in space one behind the other. Their optical axes are collinear and their roll angle is the same. This forms a unique case of epipolar geometry (see § 2.3) in which all epipolar planes contain the optical axes of the cameras, and are perpendicular to the image planes. This implies that the intersection of an epipolar plane with the two image planes form two parallel lines. If we superimpose a conventional cartesian grid onto the image planes, with the origin at their intersection with the optical axis, the two intersection lines will form the same angle with the X-axis. Hence, two corresponding image points lie along corresponding radial lines.

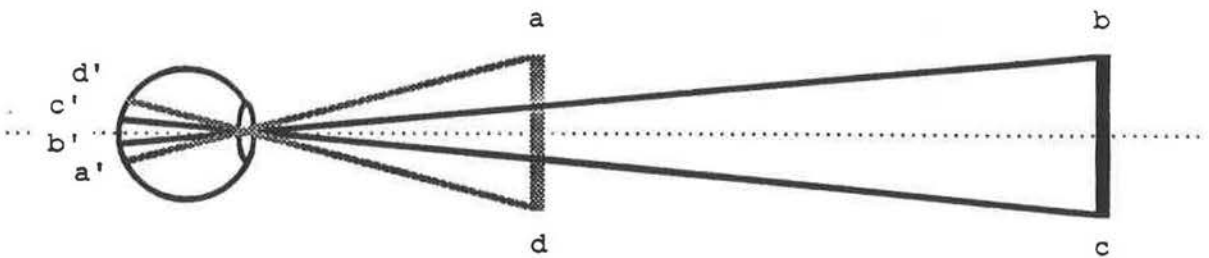


Figure 3.1: the geometry of coaxial stereo.

The coaxial camera configuration is equivalent to having a single camera which moves along its optical axis by a known interval. This is referred to as

*motion-in-depth* (see § 2.7). As the camera approaches the fixation point, the angle at which any other point in the scene is conceived, widens (fig. 3.1). The rate at which this angle increases gives rise to a stereoscopic disparity.

Motion-in-depth projects rectilinear optical flow upon the image plane (fig. 3.2). The radial lines which extend outward (as the camera moves forward) from the *focus of expansion* (FOE) [Ballard and Brown 1982, Prazdny 1980, Williams 1980, Lawton 1982, Jain 1983, 1984] are the one dimensional lines along which one seeks to resolve the correspondence problem. The direction of the optical flow implies that for each point projected onto a close image plane, its corresponding point projected onto a farther image plane is always closer to the FOE. The epipolar geometry implies that the projection of a scenic point moves along a radial line. To eliminate the effect that the distance of an image point from the fixation point has on the depth computations, the transformation described below is applied.

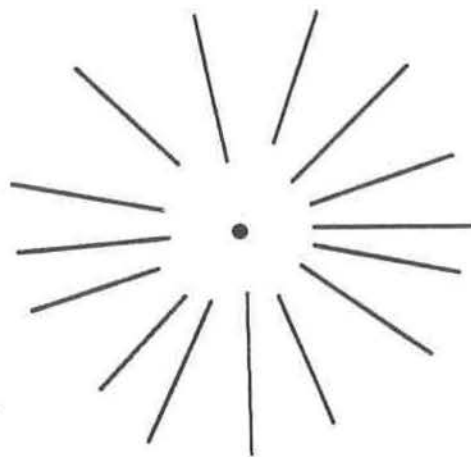


Figure 3.2: FOE

## 3.2. COMPLEX LOGARITHMIC MAPPING

### 3.2.1. Introduction

The size and rotation invariance of *complex logarithmic mapping* (CLM) appealed to several recent vision researchers, in both biological [Schwartz 1977,

1980a, 1981, 1982, Cavanagh 1978, 1981, Casasent and Psaltis 1976] and computational [Funt 1976, Weiman and Chaikin 1979, Sandini and Tagliasco 1980, Schenker et al 1981, Jain 1983, O'Brien and Jain 1984] image analysis.

Schwartz, in a series of papers, has suggested that the retinoscopic mapping of the striate cortex (area 17) performed by the human visual system, is a CLM. Furthermore, he suggested the existence of the same mapping in three other areas of the brain which perform visual analysis tasks. He also pointed out [1981] the projection invariance of CLM. By projection he meant the sequence of projective changes which a stimulus fixed in the environment would undergo as an organism approaches a fixation point. This holds true [Jain 1983, 1984, Lancia and Nicholson 1979] only if the directions of motion and gaze are collinear.

Cavanagh disputed some of Schwartz standings, stating that the size and rotation invariances hold only when measurements are made with respect to the origin of the CLM. This implies that the origin of the CLM should coincide with the FOE.

### 3.2.2. The Mathematical Aspects of CLM

Conventionally, images are sampled in cartesian coordinates. We view the cartesian image plane as a complex plane, with the X-axis being the real axis and the Y-axis being the imaginary axis (fig. 3.3).

A complex number is defined as

$$z = x + iy \quad (1)$$

where  $i = \sqrt{-1}$ . The length of the vector from the origin to  $z$  is denoted by

$|z|$  and referred to as the *modulus*. The angle that the vector forms with the X-axis is called the *argument* and is denoted by  $Arg(z)$ . The complex number  $z$  can be specified in polar form as

$$z = re^{i\theta} = r(\cos\theta + i\sin\theta) \quad (2)$$

where  $r = |z|$  and  $\theta = Arg(z)$ .

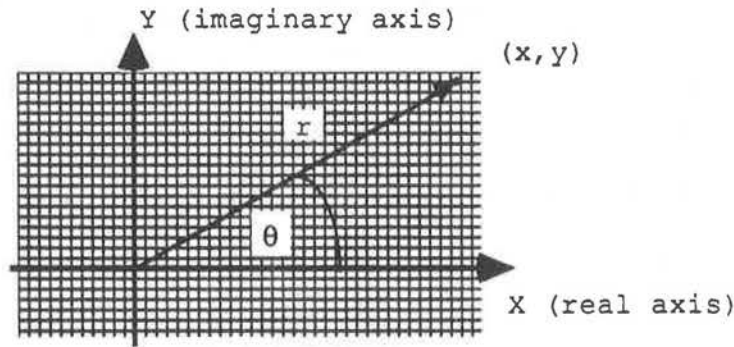


Figure 3.3: complex image plane,  $z=x+iy$

Under conventional cartesian coordinates

$$r = \sqrt{x^2 + y^2} \quad (3)$$

and

$$\theta = \tan^{-1} \frac{y}{x}. \quad (4)$$

Functions which involve complex numbers,  $w = f(z)$ , are called *mapping functions*. The  $z$ -plane (complex cartesian coordinates) is their domain and the  $w$ -plane (complex polar coordinates) their range.

With real variables, CLM is given by

$$w = \ln z \quad (5)$$



and with complex variables by

$$w = u(z) + i v(z). \quad (6)$$

From (2) and (5) we can write

$$w = \ln(re^{i\theta}) = \ln r + i\theta \quad (7)$$

Hence, the mapping given by (5) can be written as

$$u(r, \theta) = \ln r \quad (8)$$

$$v(r, \theta) = \theta \quad (9)$$

A scene point  $P_0(x_0, y_0, z_0)$  and its projection upon the image plane  $P_1(x_1, y_1, 1)$  (fig. 3.4), are given by

$$x_1 = \frac{x_0}{z_0} \quad (10)$$

$$y_1 = \frac{y_0}{z_0} \quad (11)$$

The  $(x_1, y_1)$  plane is assumed to be parallel to the  $(x_0, y_0)$  plane.

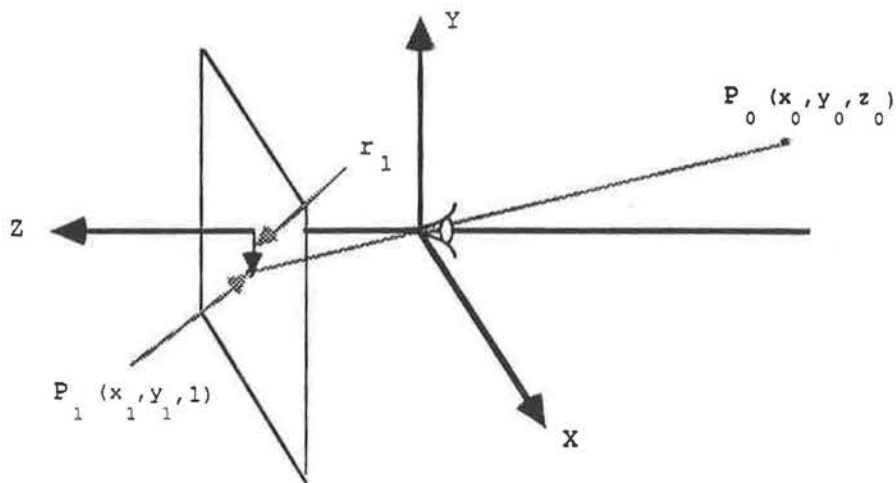


Figure 3.4: 3D projection space

Motion-in-depth of the camera results in

$$\frac{\partial r_1}{\partial z_0} = \frac{\partial \sqrt{x_1^2 + y_1^2}}{\partial z_0} \quad (12)$$

From (10), (11) and (12), we can write

$$\frac{\partial r_1}{\partial z_0} = -\frac{\partial \sqrt{x_0^2 + y_0^2}}{\partial z_0} \frac{1}{z_0} = -\frac{\partial r_0}{\partial z_0} \frac{1}{z_0} = -\frac{r_1}{z_0} \quad (13)$$

The CLM horizontal displacement can be expressed as

$$\frac{\partial u}{\partial z_0} = \frac{\partial u}{\partial r_1} \frac{\partial r_1}{\partial z_0} \quad (14)$$

From (8) we have

$$\frac{\partial u}{\partial r} = \frac{1}{r_1} \quad (15)$$

and from (14), (13) and (15) we obtain

$$\frac{\partial u}{\partial z_0} = -\frac{1}{z_0} \quad (16)$$

The CLM vertical displacement can be expressed as

$$\frac{\partial v}{\partial z_0} = \frac{\partial v}{\partial \theta} \frac{\partial \theta}{\partial z_0} = 0 \quad (17)$$

It can be observed (17) that the 'looming' of a camera generates no vertical displacement of  $P_1$  upon the image plane. Equation (16) implies that from the horizontal displacement and from the (known) movement of the camera, the depth of the point  $P_0$  can be determined.

### 3.2.3. Geometrical Features of CLM

The attraction of CLM is in its various geometrical features. Concentric circles map into vertical lines, and rotation in the  $z$ -plane corresponds to vertical displacement in the  $w$ -plane. Logarithmic spirals map into inclined straight lines. Radial lines map into horizontal lines, while magnification corresponds to horizontal displacement in the  $w$ -plane. This implies that in an environment of stationary objects, if a camera move along its optical axis, all projected images are invariant in size and exhibit horizontal displacement only. This fact plays an important role in simplifying the extraction of depth from motion-in-depth. Note that this is a feature of polar space and is independent of the logarithmic part of CLM.

Weiman and Chaikin [1979] pointed out that congruent square cells in the  $z$ -plane, are transformed in the  $w$ -plane into curvilinear bounded cells which are mutually similar (fig. 3.5). The inverse of CLM in which rectangular regions are mapped into annular regions is known as the *exponential mapping* [Kober 1952, Churchill 1960], and guarantees the mapping to be *conformal* [O'Brien and Jain 1984].

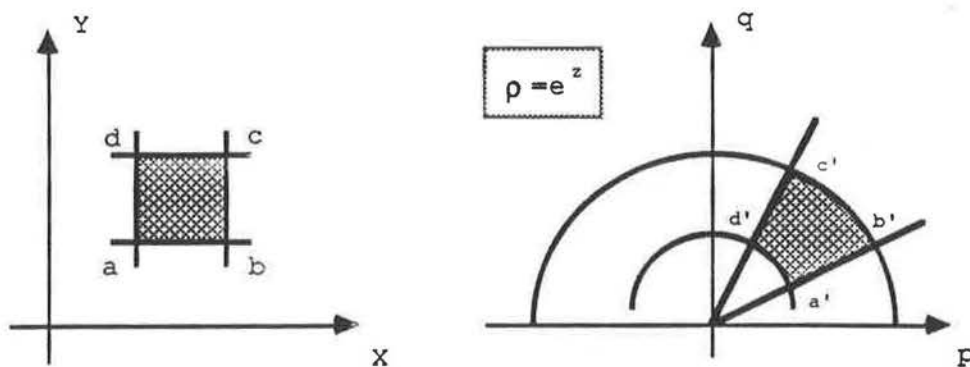


Figure 3.5: the exponential mapping

### 3.2.4. Practical Deficiencies of CLM

Mathematically, when mapping an image from cartesian space into polar space, each point in one space corresponds to exactly one point in the other. But when storing these points and their intensity values in a finite state machine, they must be quantized into picture elements (pixels).

There are several difficulties one must consider when mapping an image from cartesian space to polar space (fig. 3.6). O'Brien and Jain [1984] pointed out that a direct interpolation, where the intensity value of each pixel in the cartesian image is assigned to a corresponding pixel in the polar image, produces discontinuities and result in a broken image. An inverse interpolation, where each pixel in the polar image is set to an interpolated intensity value of the corresponding position in the cartesian image, renders much smoother results.

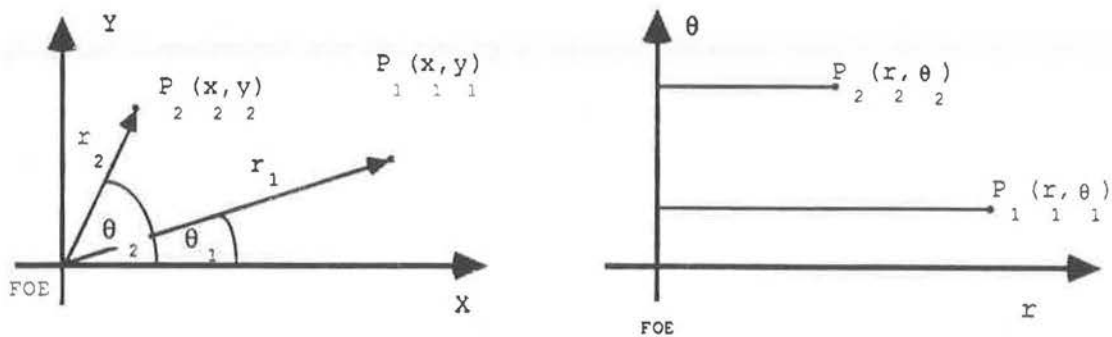


Figure 3.6: cartesian to polar mapping

Cartesian space is conventionally tessellated by a square grid which implies that all pixels are of even size and shape. In polar space, pixels preserve shape only in a retina-like mapping [Sandini and Tagliasco, 1980,

Funt, 1976]. The pixels *width* always grow in size as their distance from the origin increases. The rate at which it grows is constant, since it depends only on the interval chosen between two consecutive  $\theta$ 's. This brings about two related problems. High resampling density is generated around the origin (FOE), which is maximized as we get closer to the origin. In the periphery, a problem of opposite nature emerges. The density of new sampled points decreases as we approach the boundary of the polar image.

The rate at which the *depth* of a polar pixel changes depends on the scheme selected for  $r$ . The two schemes considered here are logarithmic (exponential) spacing and even spacing. In the first case the depth of the pixels grows exponentially as we move away from the origin, and in the second one it remains the same. A logarithmic mapping enhances both the high density problem around the origin and the low density problem at the periphery. Hence, the uniform resolution of a cartesian image is lost when it is transformed into CLM space. Instead, we obtain a retina-like image, with very high resolution at the center and low resolution in the periphery [Sandini and Tagliasco, 1980].

An even interval mapping, on the other hand, avoids both these problems in the depth direction. Furthermore, selecting an interval equal in size to a cartesian image pixel, keeps the resolution of the two images closely related. Thus, it was implemented. Since the logarithmic mapping is desired for simplifying the computation of depth from disparity, it is postponed to the stage where the actual depth is computed. Notice that additional computational savings are made as the logarithmic transformation is applied only to selected feature points, rather than to all pixels of the image.



# CHAPTER 4

## SCALE-SPACE IMAGES OF ONE DIMENSIONAL FUNCTIONS

### 4.1. INTRODUCTION

Scale-space images are used in this project as a representation for point-like features which are extracted and matched to obtain the local disparities.

In this chapter we take a slight digression from the main subject to discuss scale-space images (SSIs) of 1D functions. Some new results on the structure of such SSIs are presented.

#### 4.1.1. The Scale-Space Image Concept

The concept of SSIs was first introduced by Stansfield [1980] but was not pursued at that time. It was reintroduced by Witkin [1983], who made some interesting observations about the nice behavior of contours in a SSI.

A SSI of a 1D function is a complete coarse-to-fine description of that function. It is obtained when a function is convolved with the second derivative of a Gaussian filter, at constantly increasing scales. The reason for convolving a signal with the second derivative of a Gaussian filter is to enhance abrupt value changes. At each scale level the convolution produces a smoothed function whose degree of smoothness is expressed by features called *zero-crossings* (ZCs) (see § 2.4). The accumulation of these ZCs, extracted at different scale levels, forms nicely behaved contours which are referred to as

*scale-space contours.*

There are two types of contours which may appear in a complete SSI of a continuous 1D function: open contours and the more common closed contours. The major differences between the two lies in the fact that closed contours are of finite height and have a single peak, while open contours extend to infinity and have no peaks.

#### **4.1.2. Some New observations**

In this chapter, we present some new observations about the structure of contours in a SSI of 1D functions.

One observation made by Witkin [1983] was the hierarchical structure of a SSI to which he referred to as "interval trees". Mokhtarian and Mackworth [1984] based their matching algorithm on this hierarchical structure, stating that for their application scale-space contours may never intersect each other. Yuille and Poggio stated [1983a] in their investigation into the properties of these contours, that it is never empirically observed that three open contours, commencing at a fine  $\sigma$  level, merge into one such contour at a coarser scale ( $\sigma$ ) level. We will show that scale space contours may in fact intersect each other. But before doing so, we discuss some of the conditions that give rise to open contours, look at some of their properties and their relations with their corresponding functions.

There are two primitive signals which generate two primitive types of open scale-space contours. They are the Dirac delta function and the single step function (which we assume to be continuous). The single step function generates the *spike* contour which is a straight line extending upward and



composed of ZCs that are independent of  $\sigma$ . The Dirac delta function produces the 45° *rabbit-ears* contour which is composed of two straight lines diverging symmetrically about the function's origin in a 45° angle.

All contours begin to climb in the fine to coarse  $(x, \sigma)$  plane as open contours. Most of these open contours join one of their nearest open neighbors, form a closed contour and cease to exist as  $\sigma$  increases in value. Others just continue their infinite way up the  $(x, \sigma)$  plane. In some special cases, three open contours will merge into one and form a closed contour which is intersected by an open contour. That open contour may then either proceed alone, join with another open contour and form a closed contour (which intersects another closed contour) or merge with another two open contours into one, and so on and so forth.

#### 4.1.3. Approach and Assumptions

The five assumptions made by Yuille and Poggio [1983a] about the Gaussian filter, guide this investigation as well. They are as follows:

- filtering is linear and shift-invariant and, hence, a convolution;
- the filter has no preferred scale length;
- the filter recovers the whole image at sufficiently small scales;
- the position of the center of the filter is independent of  $\sigma$ ;
- the filter goes to zero as  $x \rightarrow \infty$  and as  $\sigma \rightarrow \infty$ .

We make the following assumptions about the 1D functions:

- a function is continuous;
- the domain of a function is infinite;

- a function is monotonically asymptotic to a constant value as  $x \rightarrow \pm\infty$  (but not necessarily the same at both  $+\infty$  and  $-\infty$ ).

In other words, although functions are continuous and infinite, they may have arbitrary value fluctuations only inside some defined domain interval. Outside that interval, when smoothed with a Gaussian filter with a large enough  $\sigma$ , the function's values must change monotonically and it becomes asymptotic to some constant value as  $x \rightarrow \pm\infty$ . The *sin* and *sinc* functions are examples of signals that do not satisfy these constraints.

A function has four values which effect the behavior of contours in its SSI. They provide us with a means by which functions can be categorized, and are as follows:

- a) the initial value of a function:  $init = \lim_{x \rightarrow -\infty} f(x)$ .
- b) the final value of a function:  $fin = \lim_{x \rightarrow +\infty} f(x)$ .
- c) the maximum value of a function:  $fmax \geq f(x)$ .
- d) the minimum value of a function:  $fmin \leq f(x)$ .

In case (1) we attend to signals whose initial and final values are equal. In case (2) we attend to signals whose initial and final values are not equal. To simply our analysis we assume that  $fin \geq init = 0$  and that  $fmax > 0$ .

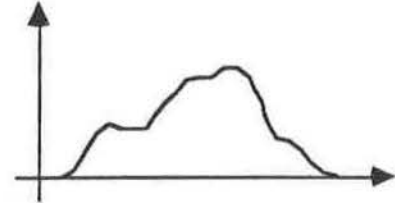
## 4.2. ON THE GENERATION OF OPEN CONTOURS

### 4.2.1. Case 1: signals where $fin = init$

This category is concerned with signals whose initial and final values are

equal. If we assume for a moment that the signals wrap around, then they can be viewed as closed planar curves. Mokhtarian and Mackworth [1984] observed that the SSI of such curves contain only closed contours. Contours that appear to be open are assumed to represent curves about which only partial information is available. Since we are interested not in wrap-around functions but in infinite functions, the "openness" of the functions allows for open contours to be valid scale-space contours.

**4.2.1.1. Case 1.1: signals where  $f_{min} = init = 0$ .**



The most primitive signal which complies with the given constraints is the *impulse* or *Dirac delta* function (fig. 4.1a),  $\delta(x)$ , which is given by

$$\begin{cases} \delta(x) = 0 & x \neq x_0 \pm \epsilon \\ \int_{-\infty}^{\infty} \delta(x) = \int_{x_0 - \epsilon}^{x_0 + \epsilon} \delta(x) = 1 \end{cases} \quad (18)$$

where  $\epsilon$  is an arbitrarily small number greater than 0.

The impulse is the identity function under convolution. This is expressed by

$$\delta(x) \otimes f(x) = \int_{-\infty}^{\infty} \delta(t) f(x-t) dt = f(x-t) |_{t=0} = f(x) \quad (19)$$

Thus, convolving  $\delta(x)$ <sup>1</sup> with the second derivative of a Gaussian filter renders

1. Although  $\delta(x)$  is not a continuous function, we could treat it as a concept in the theory of distribution [Gupta, 1966, Rees et al, 1981] in order to enhance our level of mathematical rigor. But this would only complicate the notations used while producing the same results. Thus, for practical reasons, we treat the impulse as a continuous function.

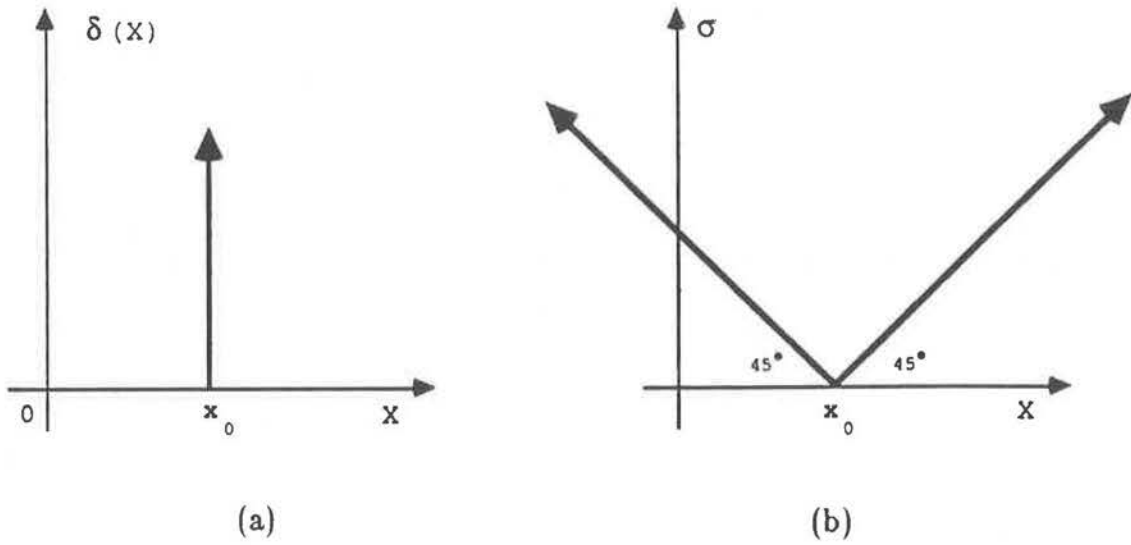


Figure 4.1: The Dirac delta ( $\delta$ ) function (a) and its scale-space image (b), composed of the *rabbit-ears* contour.

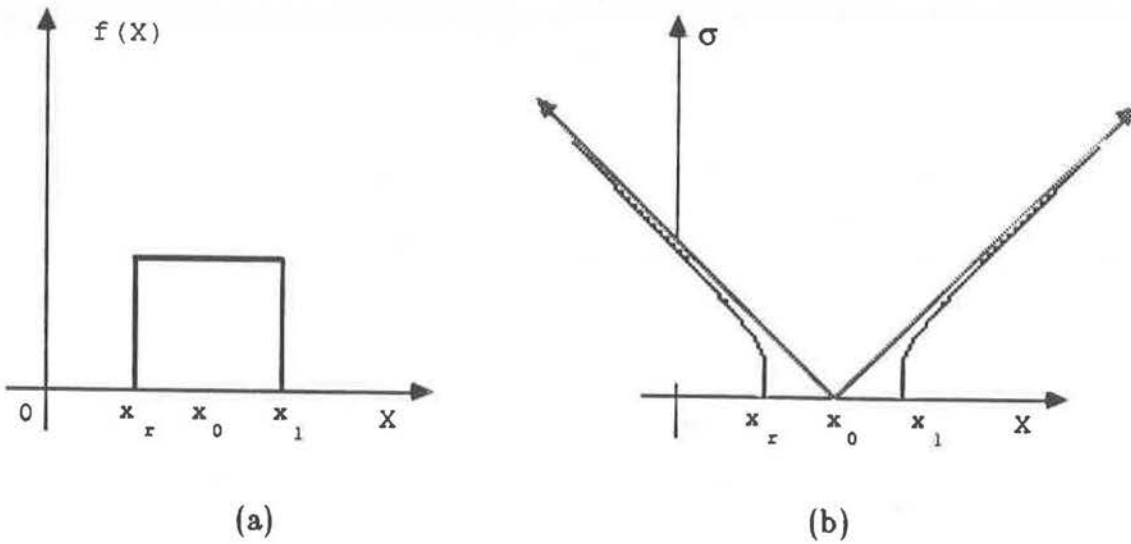


Figure 4.2: A pulse function (a) and its scale-space image (b).

the filter itself, centered at the impulse function position. The accumulation of the filter's ZCs, extracted at incrementing  $\sigma$  values, forms the SSI in figure 4.1b: two straight open contours that extend diagonally in a  $45^\circ$  angle (when the  $\sigma$  interval is 1) in each direction. This pair of open contours is referred to as the  $45^\circ$  *rabbit-ears* contour.

Notice that if we have at  $x \neq x_0 \pm \epsilon$

$$\delta(x) = c,$$

where  $c$  is a constant, then the same SSI is generated. Signals such that

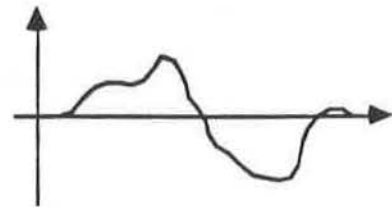
$$f(x) = \begin{cases} c & x < x_l \\ c & x > x_r \\ v & v > c, x_l \leq x \leq x_r \end{cases}$$

(fig. 4.2a), contain in their SSI two open contours which eventually converge to be asymptotic to an imaginary  $45^\circ$  rabbit-ears contour, whose origin is midway between  $x_l$  and  $x_r$  (fig. 4.2b). Furthermore, we prove that signals in case (1.1) contain in their SSI exactly two open contours. The complete proof is given in appendix III, case A1.

#### 4.2.1.2. Case 1.2: signals where

$f_{\min} < \text{init} = 0$ .

We can distinguish here two sub-cases: functions that approach  $\pm\infty$  with the same sign and those which approach  $\pm\infty$  with opposite signs.



Functions that approach  $\pm\infty$  with the same sign resemble case (1) above. They have either two or four open contours in their SSI. This is proved

and discussed in appendix III, case A3.

Functions that approach  $\pm\infty$  with opposite signs have either three or five open contours in their SSI. This is proved and discussed in appendix III, case A4. A simple example that illustrates this sub-case is given in figure 4.3. It shows a signal composed of one cycle of a squared wave (fig. 4.3a). Its SSI is composed of exactly three open contours. Two open contours are asymptotic to an imaginary  $45^\circ$  rabbit-ears contour, and the third one extends straight up midway between the "ears" (fig. 4.3b). It is referred to as the *spike* contour. If the above wave signal is asymmetric (fig. 4.4a) the open contours bend towards the side of the signal, relative to the location where it crosses the zero, which encloses a smaller area under its curve (fig. 4.4b). However, the basic pattern of three diverging open contours is preserved. This is proven in appendix III, case A2.

While in case A4 of appendix III we show that functions may contain five open contours in their SSI, we have not empirically observed any one function that generates more than five open contours in its SSI.

#### **4.2.2. Case 2: signals where $f_{in} > f_{init} = 0$**

This category is concerned with signals whose initial and final values are not equal in value. We concentrate our discussion on functions whose values at any given location are bounded by their initial and final values.

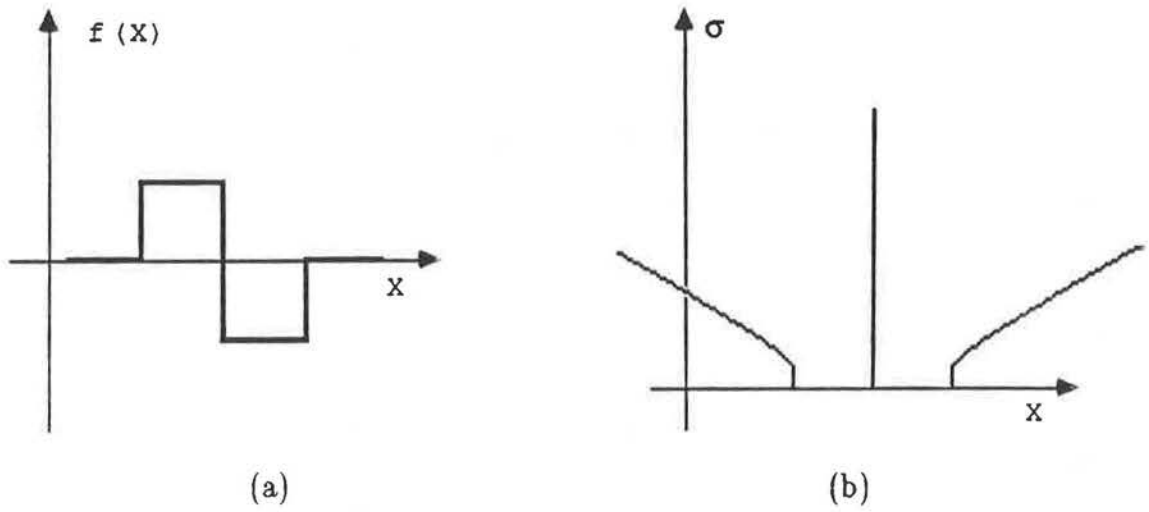


Figure 4.3: A single cycle wave function (a) and its scale-space image (b).

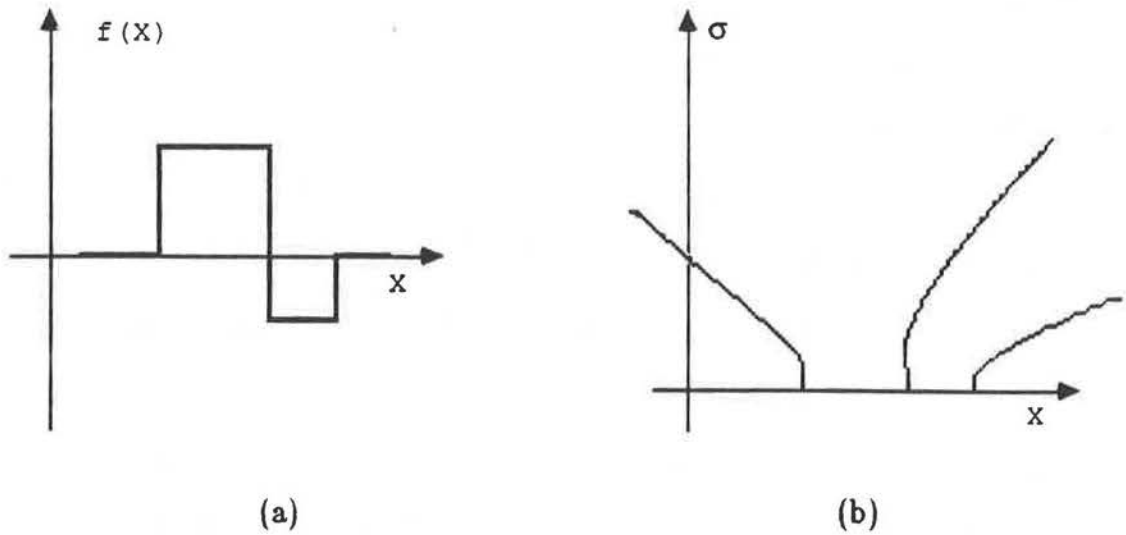
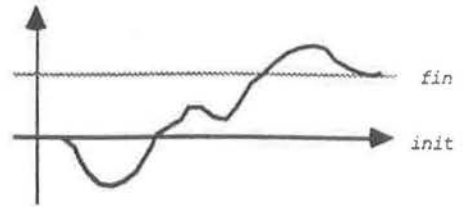


Figure 4.4: An asymmetric single cycle wave function (a) and its scale-space image (b).

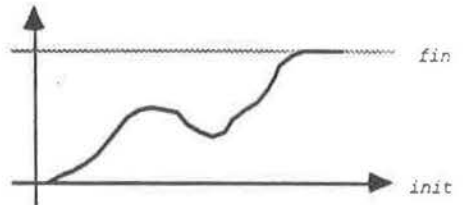
**4.2.2.1. Case 2.1: signals where  $f_{max} > f_{in}$  and/or  $f_{min} < init$**

This case is similar to cases (1.1) and (1.2) above. The imbalance between the initial and final values will force all contours to bend towards the side of the  $(x, \sigma)$  plane that correspond to that end in the generating function which show a smaller value change.



**4.2.2.2. Case 2.2: signals where  $f_{max} = f_{in}$  and  $f_{min} = init$**

All such functions have exactly one open contour in their SSI. This is proven in appendix III, case A5. This contour is either a spike contour, or an open contour which eventually converges to be asymptotic to an imaginary spike contour, arising midway between  $x_l$  and  $x_r$ , where  $f(x) = init$  for all  $x < x_l$ , and  $f(x) = fin$  for all  $x > x_r$ .



We confine our discussion to step signals, although most observations apply to signals containing both steps and troughs. As in the case of the Dirac delta function, we treat this basic signal as a continuous function. The mathematical definitions of step functions are given in appendix I.

Convolving a single step signal (fig. 4.5a) with a Gaussian filter produces a SSI consisting of just the spike contour (fig. 4.5b). The SSI of a double step signal contains one open contour and one closed contour. If the two steps are of uneven size, the open contour will be on the side of the SSI as is the larger



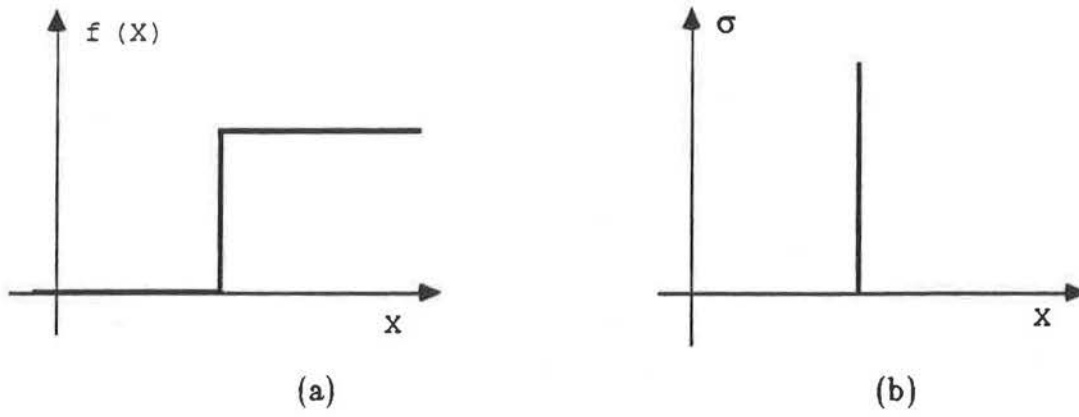


Figure 4.5: A single step function (a) and its SSI (b).

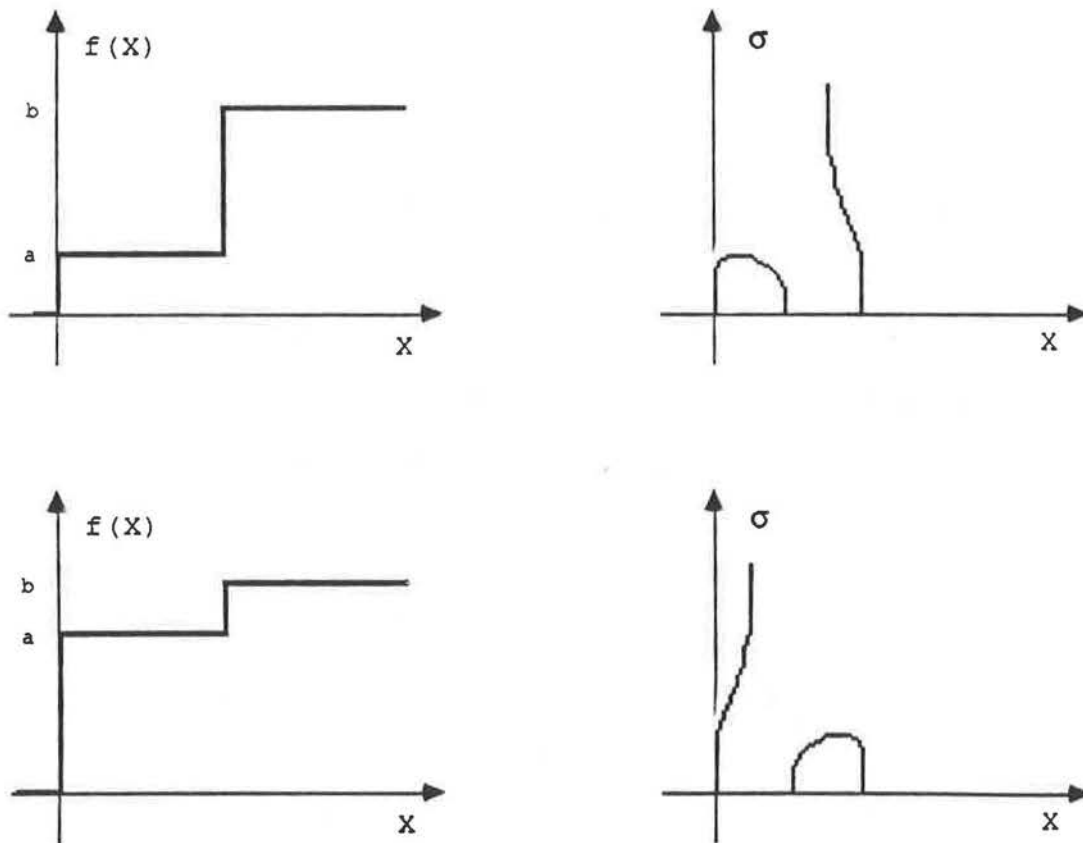


Figure 4.6: Two double step functions of uneven steps. This is to illustrate the change in position of the closed contour with respect to the open one, as the two steps change in size.

step in the generating function (fig. 4.6). If the steps are of even size, the open contour will intersect the closed contour in the middle (fig. 4.7). This corresponds to the situation where the open contour crosses over from one side of the closed contour to its other, as the smaller step becomes the larger one. We will elaborate on this phenomenon in § 4.3.

In general, multi step signals, say of  $n$  steps, generate SSIs containing one open (spike) contour,  $n-1$  closed contours and  $2n-1$  ZCs at a fine  $\sigma$  level (i.e. before any closed contours are formed). The positioning of the spike contour on the X-axis of the  $(x,\sigma)$  plane, depends on the dimensions of the different steps and may intersect any closed contour as it moves about the X-axis.

### **4.3. CROSSING CONTOURS**

We have established the validity of open contours in a complete SSI of several one dimensional functions. These contours, as we have seen, tend to change their position on the X-axis of the  $(x,\sigma)$  plane as values in the generating function change their relative value. In doing so, they may "skip over" some closed contours. In such instances when an open contour moves from one side of a closed contour to its other, an intersection of the closed contour by the open contour occurs.

#### **4.3.1. The Even Double Step signal**

A most primitive signal that illustrates this phenomenon is the even double step (EDS) signal. It is composed of two steps of equal size (fig. 4.7a) and generates a SSI containing a symmetric closed contour which is intersected in

the middle by a spike contour (fig. 4.7b). This instantiates a state of equilibrium between the two possible cases of an uneven double step signal (fig. 4.6). Notice that this particular SSI appears in the list of SSIs that "are never empirically observed when the filter is a Gaussian", in a paper by Yuille and Poggio [1983a] that outlines scale-space theorems for ZCs.

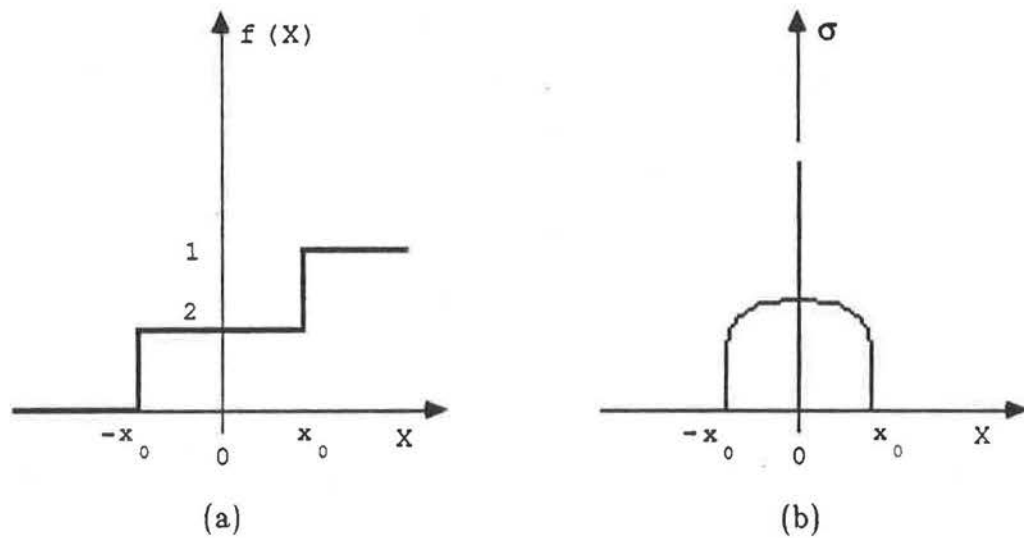


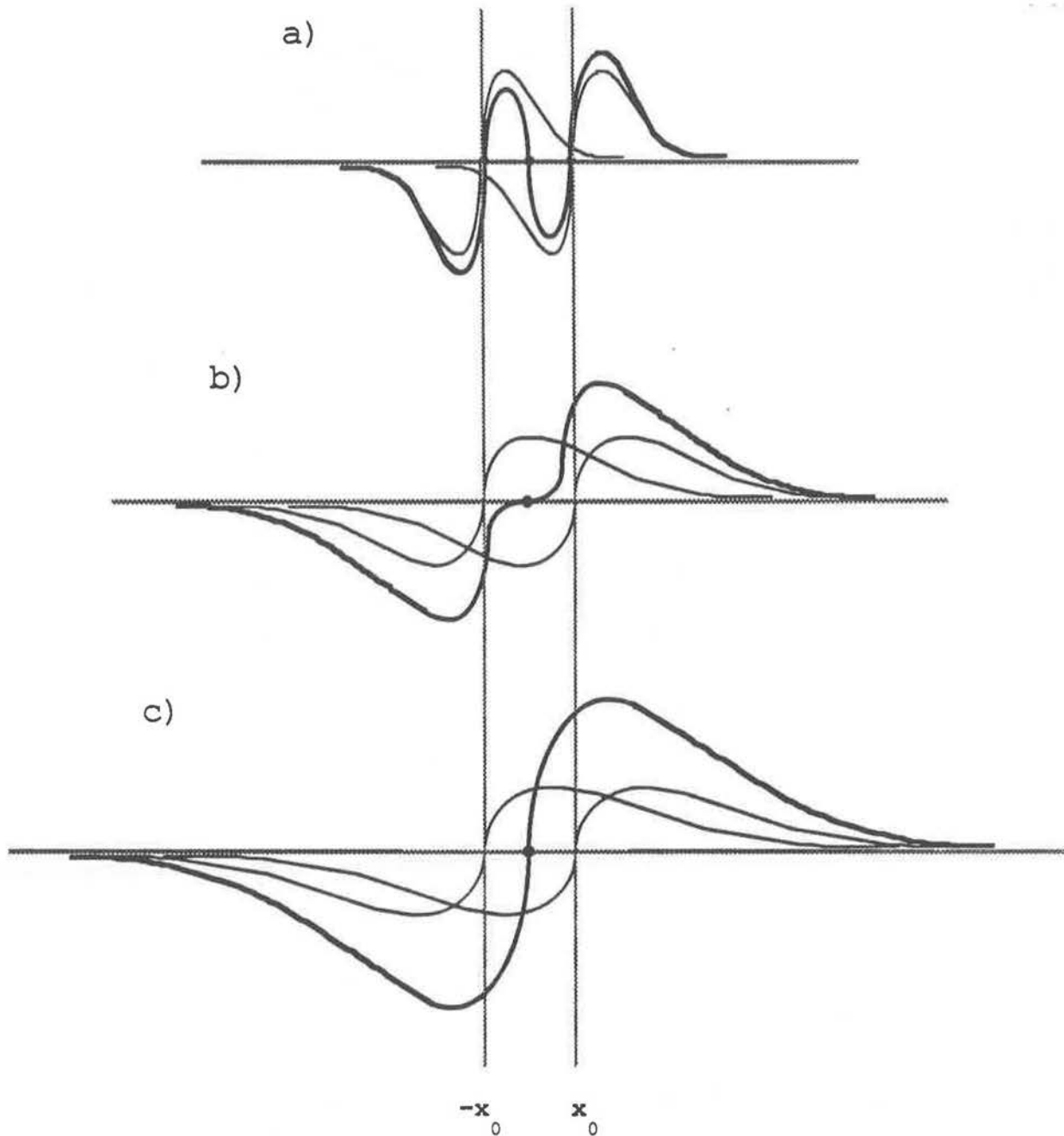
Figure 4.7: An even double step function (a) and its scale-space image (b).

The validity of the SSI of an EDS signal, can be proven as follows:

**proof**

Let the one dimensional function be  $f(x)$ , the one dimensional Gaussian filter  $g(x,\sigma)$  and the convolved function  $h(x,\sigma)$ . From (A1.6) in appendix I we can write

$$h(x,\sigma) = g'(x+x_0) + g'(x-x_0). \tag{20}$$



— the smoothed individual steps  
— their sum

Figure 4.8: The three stages of constructing a SSI of an even double step function:

- (a) the three zero-crossings at a fine  $\sigma$  ( $\sigma < x_0$ );
- (b) the three zero-crossings merge into one ( $\sigma = x_0$ );
- (c) only one zero-crossing emerges when  $\sigma$  is large ( $\sigma > x_0$ ).

The scale-space contours are generated by

$$h(x, \sigma) = 0. \quad (21)$$

Figure 4.8 shows three critical convolution stages, in the formation of this SSI.

From (20), (21) and (A2.2) in appendix II, we can write

$$0 = - \left( \frac{(x+x_0)}{\sigma^2} e^{-\frac{(x+x_0)^2}{2\sigma^2}} + \frac{(x-x_0)}{\sigma^2} e^{-\frac{(x-x_0)^2}{2\sigma^2}} \right) \quad (22)$$

$$= -\frac{1}{\sigma^2} e^{-\frac{(x^2+x_0^2)}{2\sigma^2}} \left( (x+x_0)e^{-\frac{2xx_0}{2\sigma^2}} + (x-x_0)e^{-\frac{2xx_0}{2\sigma^2}} \right) \quad (23)$$

hence

$$0 = x \left( e^{\frac{-x}{\sigma^2}} + e^{\frac{x}{\sigma^2}} \right) + x_0 \left( e^{\frac{-x}{\sigma^2}} - e^{\frac{x}{\sigma^2}} \right) \quad (24)$$

and

$$0 = x \cosh \left( \frac{x}{\sigma^2} \right) - x_0 \sinh \left( \frac{x}{\sigma^2} \right). \quad (25)$$

Thus, the equation

$$x = x_0 \tanh \left( \frac{x}{\sigma^2} \right) \quad (26)$$

represent the SSI of  $f(x)$ .

One solution for equation (26) is  $x=0$  which corresponds to the spike contour in the SSI. The other two solutions correspond to the closed contour in the SSI. They are shown graphically in figure 4.9. The ZCs are the points of

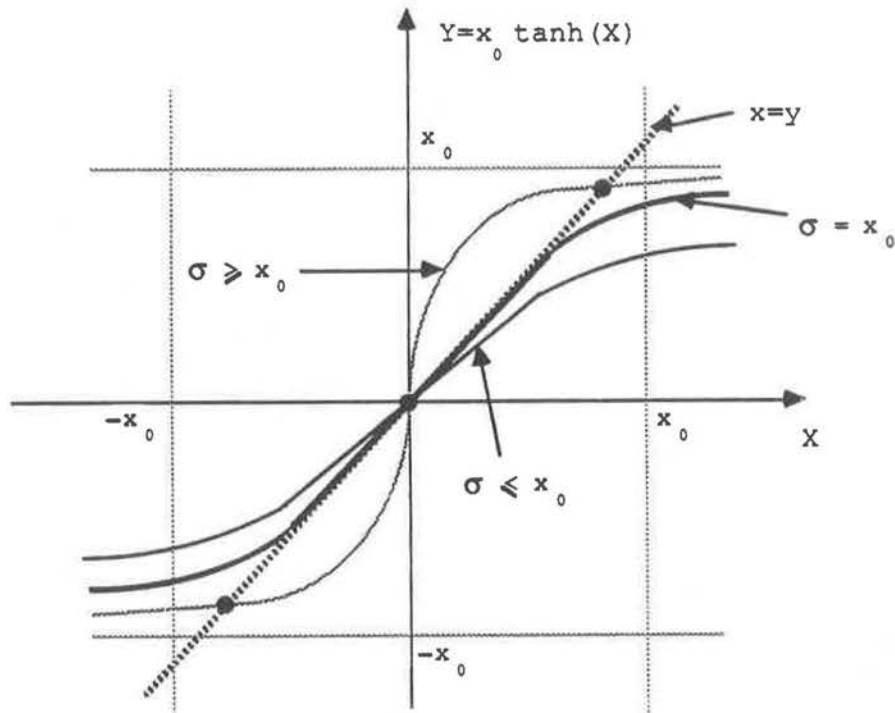


Figure 4.9: The smoothness degree of a convolved even double step function.

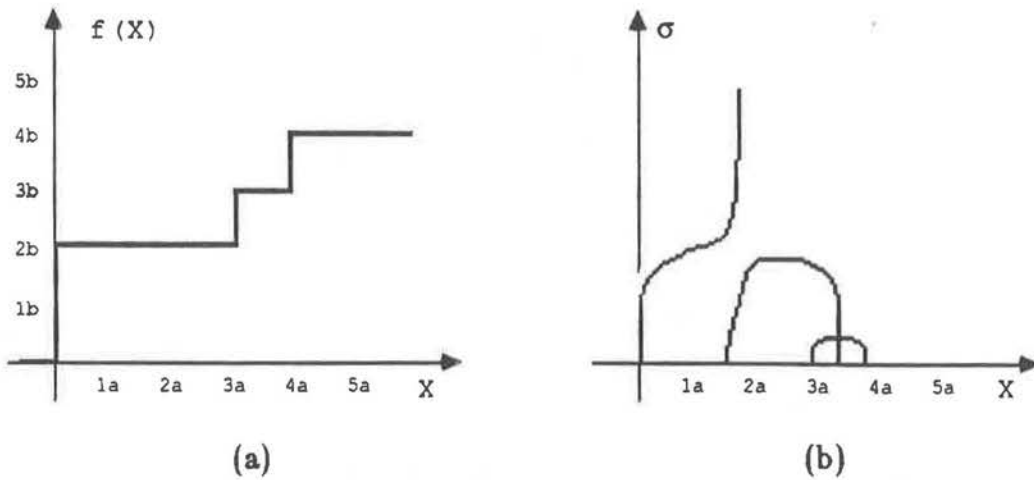


Figure 4.10: A triple step function containing an EDS (a) that generates crossing closed contours in its SSI (b).

intersection between the various curves and the line  $y=x$ . It can be observed that if  $\sigma \geq x_0$  only one ZC is generated, while for  $\sigma < x_0$  three ZCs are generated.

#### **4.3.1.1. Multi step signals which contain one or more EDSs**

Interesting SSIs can be generated by functions containing EDSs. We refer to situations where each segment of a function which contains an EDS, forms crossing contours without any interference by either neighboring steps or smaller steps situated between the two steps of the EDS.

Take, for example, a triple step signal composed of one step which is  $3a$  units wide and  $2b$  units high, and of an EDS each of size  $a$  by  $b$  (fig. 4.10a). It generates a SSI (fig. 4.10b) containing one open contour and two closed contours which intersect each other!

The validity of this SSI can be proven as done for the EDS signal.

The signal in figure 4.11a contains a pair of EDSs which are sufficiently separated. It produces a SSI shown in figure 4.11b the validity of which can be proven as before.

This of course can be extended to an infinite variety of such pyramids that can be generated by a variety of signals containing one or more EDSs.

#### **4.3.2. Wave signals**

The phenomenon of crossing contours is not limited to just signals containing EDSs, but might occur in the SSI of any signal which generates one or more open contours.

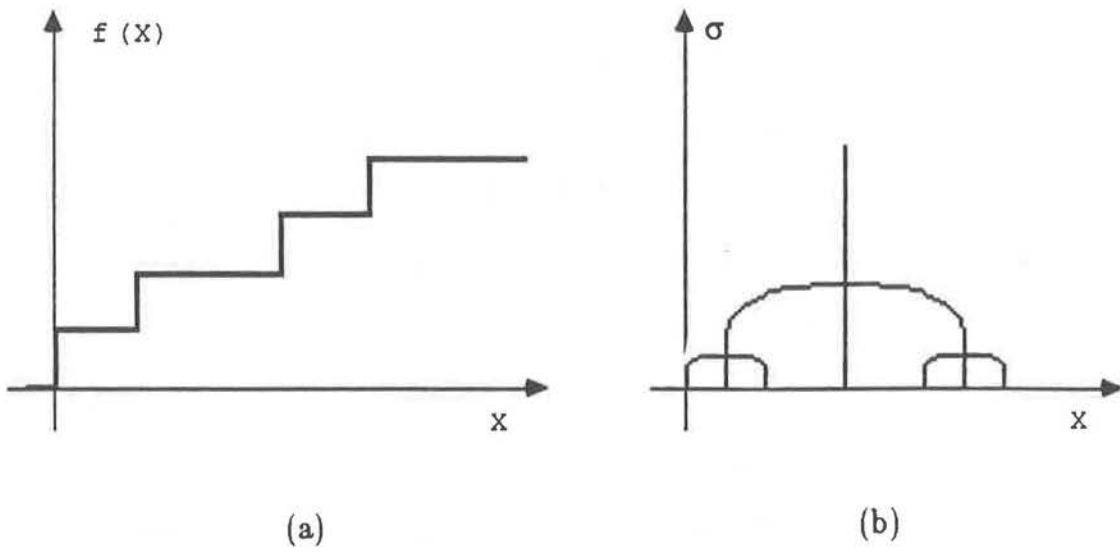


Figure 4.11: A function composed of two EDSs (a) and its SSI (b).

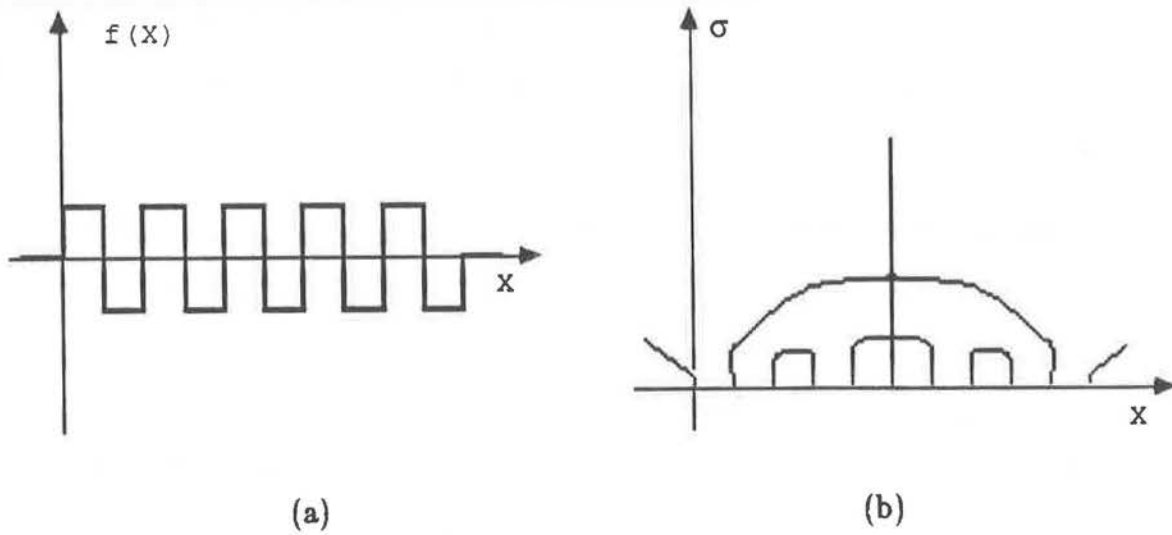


Figure 4.12: A wave function of five cycles (a) and its SSI (b).



The SSI of a squared wave signal which is composed of a single cycle is given in figure 4.3b. It contains a spike contour and a rabbit-ears contour. A signal with two cycles, generates an additional closed contour in the middle of the SSI which is intersected by the spike contour.

In general, a wave signal of  $n$  cycles, generates a SSI which contains three open contours,  $n-1$  closed contours and  $2n+1$  ZCs at a fine  $\sigma$  level. If all the cycles are complete, are of even size and form a continuous wave, then one open contour is the spike contour positioned in the middle of the SSI, and the other two are of the rabbit-ears contour, one on each side of the SSI. One closed contour is the parent of all the other closed contours, which are distributed symmetrically under its umbrella. This parent contour is always intersected by the spike contour. When the number of cycles is odd so is the number the children closed contours. The middle child contour is intersected by the spike contour. A squared wave signal of five cycles (figures 4.12a and 4.12b) is given as an example.

As in the case of the even double step signal, a variety of crossing contours can be generated with different combinations of this signal. A wave signal containing two even double cycles (fig. 4.13a) and its SSI (fig. 4.13b) are given as an example.

### 4.3.3. Other cases

Another group of signals which render similar results are signals composed of strings of pulses with intermittent sign change. One pair of such pulses (fig. 4.14a) produces a SSI (fig. 4.14b) similar to that of a two cycled wave signal. The only difference is in the curvature and height of the closed contour. A sig-

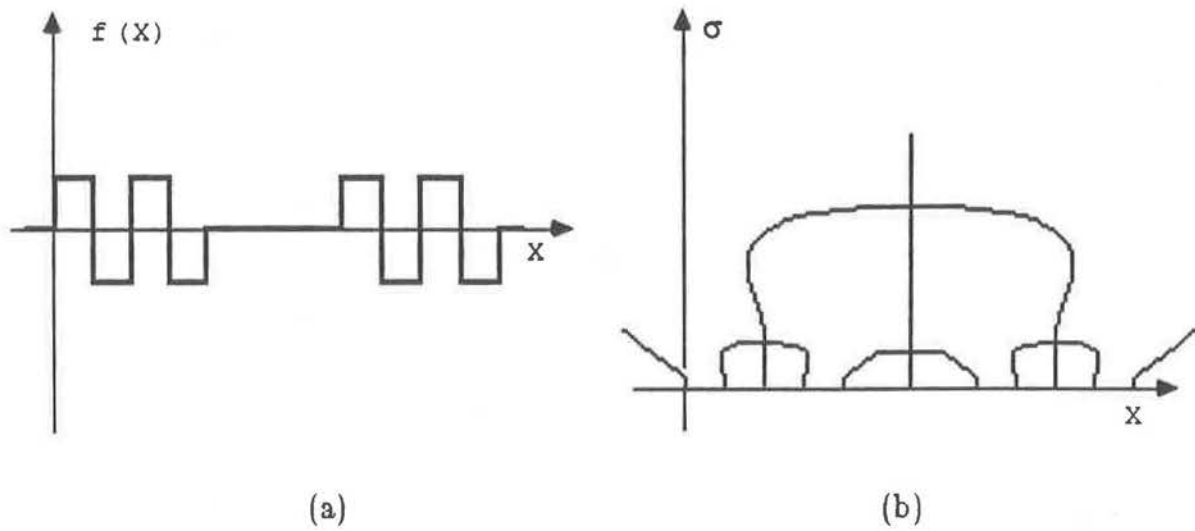


Figure 4.13: A function composed of two even double cycle wave signals (a) and its SSI (b).

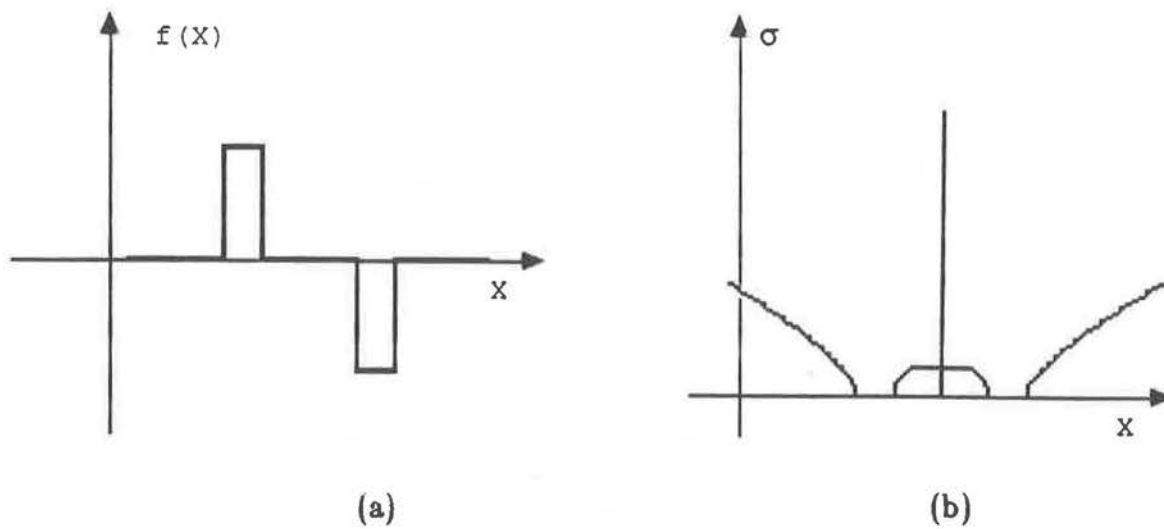


Figure 4.14: A function composed of two pulses of opposite signs (a) and its SSI (b).

nal of several pulses with intermittent sign change and its SSI are shown in figures 4.15a and 4.15b.

So far we have seen crossing contours which are formed by spike type open contours. Open contours belonging to a rabbit-ears contour, can also cross a closed contour. We have seen that a pulse signal yields a SSI containing a pair of diverging open contours (fig. 4.2b). Two pulses of equal sign generate an additional closed contour, whose placement in the SSI depends on the size ratio between the two pulses and their distance apart (fig. 4.16). If the pulses are of equal or almost equal values, the closed contour will be situated between the two open contours. If one is sufficiently smaller than the other, then the closed contour appears on the side of the SSI as is the smaller pulse in the signal. So again, there is a state of balance in which an open contour crosses a closed one. An example is given in figure 4.17.

#### 4.4. BINARY SCALE-SPACE IMAGES

One should note that locally, when two contours cross each other, the point of juncture is always formed by four lines. This suggests that SSIs can be viewed as binary images (fig. 4.18). Should this information be made explicit, it might prove to be the necessary additional information needed to confront difficulties created by the phenomenon of crossing contours. It may also resolve the breaking down of the fingerprints theorems [Yuille and Poggio, 1983b], when a spike contour is present in the SSI. For example, a non-binary SSI containing just a spike contour proposes two ambiguous step signals that may generate it. Should the SSI be binary, the unique and correct signal can

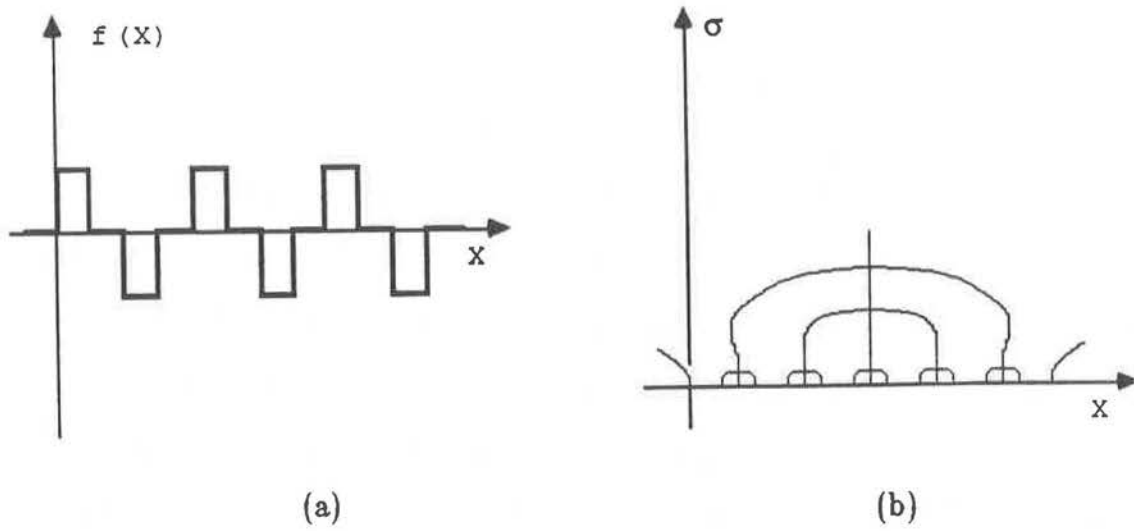


Figure 4.15: A multi-pulse function of intermittent sign change (a) and its SSI (b).

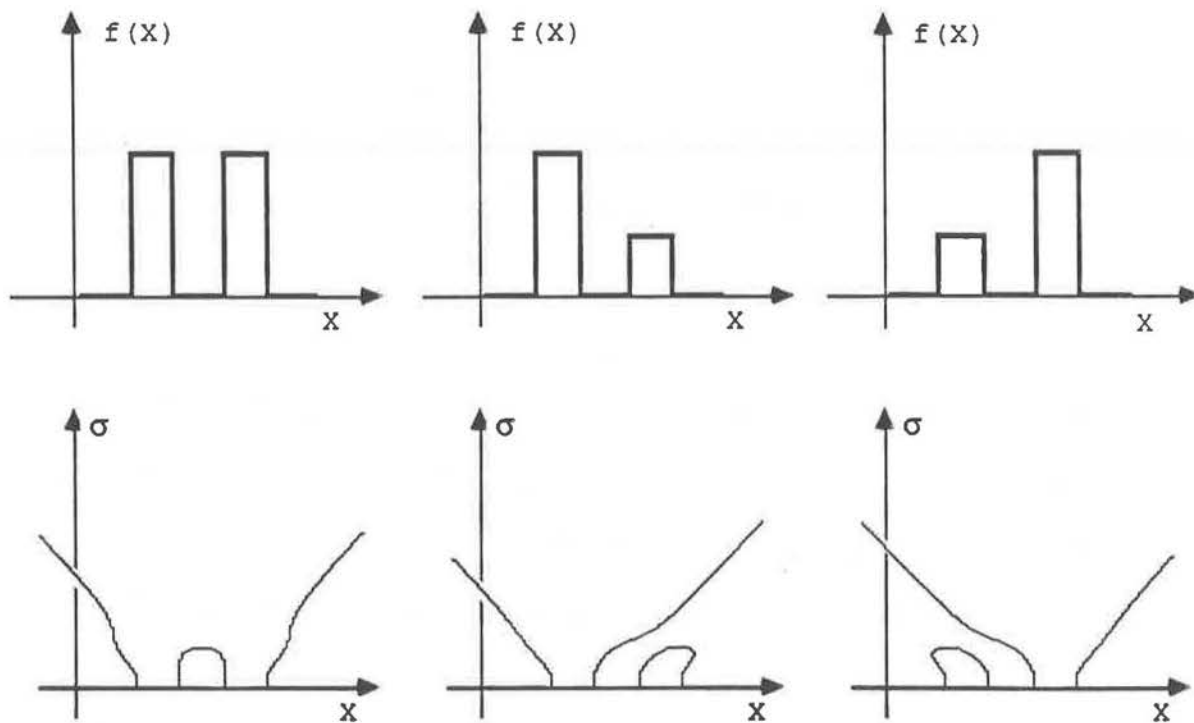


Figure 4.16: A signal composed of two pulses of various relative energy and their corresponding SSIs. This is to illustrate the positioning of the closed contour with respect to the open contours as these energies change.

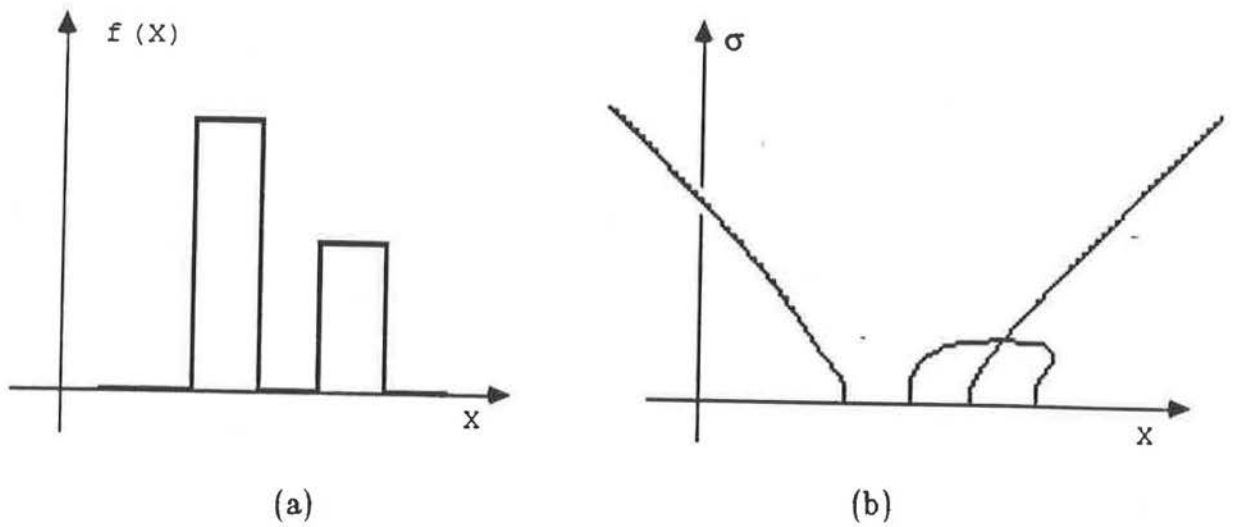


Figure 4.17: A function composed of uneven pulses (a) that generates in its SSI (b) a rabbit-ears contour that crosses a closed contour.

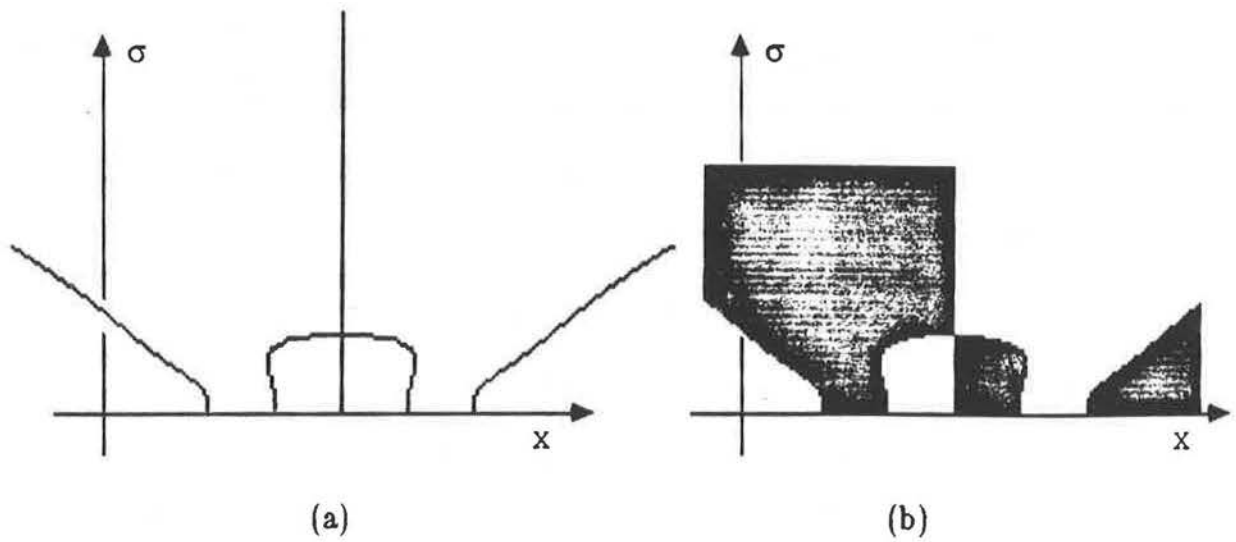


Figure 4.18: The scale-space image of an even double cycled wave signal (a) and its binary form (b).

be recovered. The same can be observed for all SSIs that contain a straight spike contour and are symmetric about this contour.

Since a SSI is formed by individual ZCs, whose direction is available at computation time, and since all the ZCs that form a particular contour have compatible direction, a binary SSI can be easily computed. This idea requires more work and is currently being studied.

#### 4.5. CONCLUSION

In this chapter, we have analyzed the structure of SSIs of some selective basic signals, concentrating on the formation and behavior of open contours.

We proved that 1D functions contain in their SSIs at least one and empirically observed no more than five open contours. Signals which obey  $\lim_{x \rightarrow -\infty} f(x) = \lim_{x \rightarrow +\infty} f(x) = c$  where  $c$  is a constant, contain in their SSI at least two diverging open contours. If the values of a signal at any given location stay either above or below  $c$  there are exactly two open contours in the SSI. These contours converge to be asymptotic to an imaginary 45° rabbit-ears contour.

In the category of signals where  $\lim_{x \rightarrow -\infty} f(x) \neq \lim_{x \rightarrow +\infty} f(x)$ , we concentrated on step signals which at some coarse  $\sigma$  value contain in their SSI just one open contour. This contour is either the spike contour or converges to be asymptotic to an imaginary spike contour. There can be only one such contour in a SSI.

We proved the validity of the crossing contours phenomenon. We ela-

borated on the even double step signal whose SSI, at a fine  $\sigma$  level, contains three open contours which merge into one, at a coarser  $\sigma$  level. In doing so, it forms a closed contour which is intersected by an open contour. This gives rise to other step signals which in turn can generate in their SSI closed contours that intersect each other. We have also explored other signals, such as wave signals, which generate crossing contours.





# CHAPTER 5

## SCALE-BASED MATCHING

### 5.1. INTRODUCTION

Mokhtarian and Mackworth [1984] have pointed out that the invariant properties of SSIs entail the desirability of carrying out the matching process in this space domain. Their invariance under uniform scaling is essential for coaxial stereo matching, while their invariance under translation is essential for conventional stereo matching. Furthermore, in a coaxial stereo system, the invariance of SSIs under rotation may prove very helpful in detecting and correcting camera rotation between frame taking.

Yuille and Poggio have shown [1983b] that an intensity profile of an image can be recovered from its SSI (up to an equivalent class of linear variation). In other words, such a SSI is a complete description of an intensity profile and of its intensities inter-relationships. Thus, matching SSIs is preferred to matching at specific scales. Each such scale represent a horizontal slice of the SSI and contains only partial information. Moreover, when selecting a specific scale, it is not clear which scale is "best" since it may vary for different intensity profiles.

### 5.2. APPROACH CONSIDERATIONS

Pairs of corresponding rows in the polar images are matched in sequence. A SSI is built for each row. The left edge of a SSI corresponds to the FOE.

The right edge corresponds to the frame boundary of the respective radial line in the cartesian image.

The goal of the matching process is to establish correspondence between contours in two SSIs. In coaxial stereo we have, inherently, an established correspondence for one point in the scene: the fixation point (the FOE in the images). In other words, the left edges of both SSIs always match. This presents a great advantage that most other stereoscopic systems lack. Hence, one can start the matching process at this point and proceed with all other scale-space contours in sequential order.

However, such an approach would weaken the *generality* of the matching process since it cannot be applied to most other visual expert systems, and conventional stereo in particular. Another deficiency of this approach is in the degree of *robustness*. In the case of highly textured images, especially around the fixation point, the process will begin by matching small contours which may prove unreliable. Furthermore, such contours appearing in the SSI of the closer image (the closer SSI, here on) may not show in the SSI of the farther image (the farther SSI, here on).

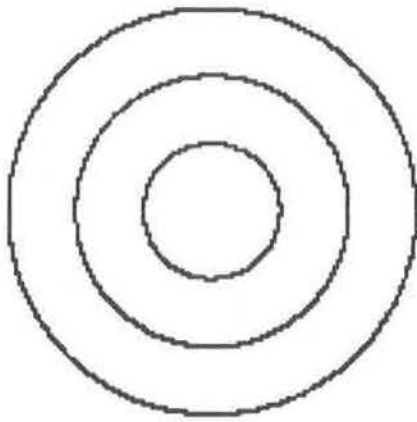
Hence, we took the approach of matching the contours top-down. The process starts with matching the highest closed contours in each SSI, which usually represents the sharpest intensity change in the images. Notice, that by starting with the highest closed contour in the closer SSI, a match for it must exist in the farther SSI.

### **5.3. THE DIFFICULTIES IN ESTABLISHING CORRESPONDENCE**

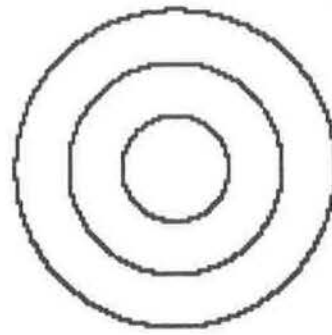
In an ideal situation, all contours will match left to right in sequential

order. However, one must be aware of the following potential problems:

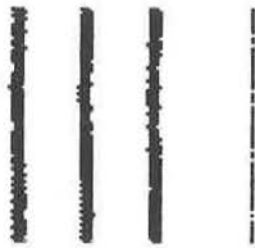
- (a) As an observer approaches an object in the scene, more and more details of that object are revealed and recorded on the retinal image. This may result in extra small contours in the closer SSI.
- (b) Some features observed at one distance may be occluded at a different range (usually a closer range). This will result not only in less features in that SSI, but in a drastic change in its structure. Thus, straight forward SSI matching is no longer possible. However, it might be possible to estimate the potential for changes the SSI might undergo. This problem is not addressed in this project and is left for future work.
- (c) A scenic area projected onto a fixed size image plane is bounded, among other things, by the distance separating it from the camera. As the camera moves away from the scene, more area is projected onto the image plane. This results in additional information in the farther image, which translates into extra contours on the right hand side of its SSI. This problem is resolved by directing the matching process from the closer SSI to the farther one.
- (d) The formation of closed contours is highly sensitive to changes in the balance of the image intensities. Figure 5.1 illustrates this problem. The synthetic images in (a) and (b) show three flat annular shapes as viewed from two distances. Their polar transformations are shown, respectively, in (c) and (d). The SSIs shown in (e) and (f) are of the first rows in the polar images. Although these rows correspond, the structure of their SSIs is different. Hence, two images of the very same scene, taken from different distances, may in fact render SSIs that cannot be directly matched.



(a)



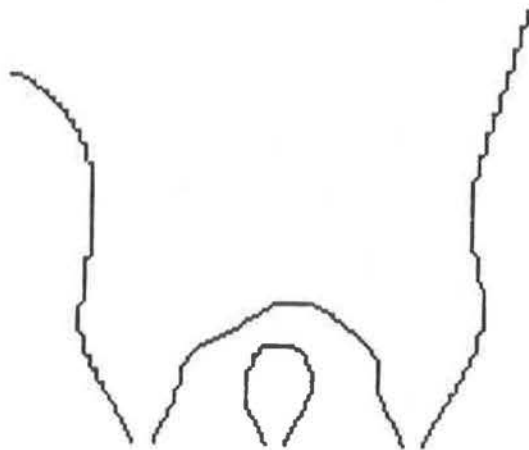
(b)



(c)



(d)



(e)



(f)

Figure 5.1: An illustration of the sensitivity of scale-space contours to the balance of intensities in the image.

- (e) As a result of problem (d) above, contours have a tendency to move about the X-axis of the SSI. This is especially true for spike contours. In such cases (fig. 5.2), a careful analysis of the SSIs is required. This problem as well is left for future work.
- (f) A consequence of problem (e) above is the possible formation of crossing contours. A broad discussion of this phenomenon is given in chapter 4. To deal with this problem, we suggest a *two-tone binary SSI* (fig. 3.18). The contours in this type of a SSI are the peripheral lines that enclose two types of regions: those that render convolution values less or equal to 0, and those with values greater than 0. This approach eliminates the problem of crossing contours by changing the status of crossing contours to distinguishable touching contours. We have not integrated this suggestion into our current system.

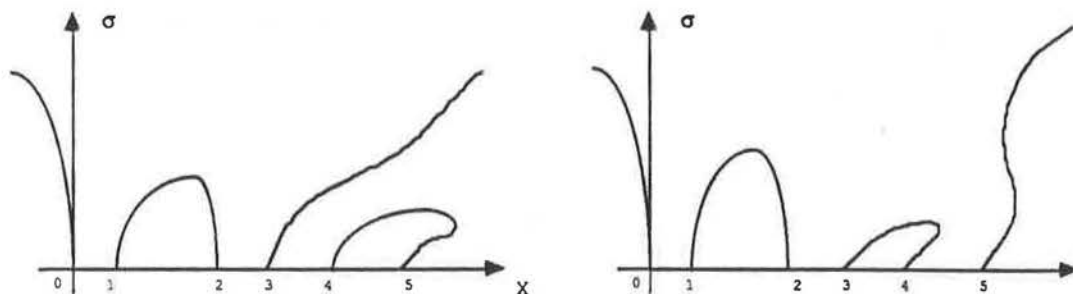


Figure 5.2: movement of an open contour.

these two SSIs exemplify the problems involved when a spike contour is moving about the X-axis. The numbers marked on the X-axis reflect actual matching points. While matches between points 0, 1 and 2 can be obtained by direct contour matching, it is not the case for points 3, 4 and 5.

#### 5.4. THE MATCHING ALGORITHM

A 1D SSI is a coarse-to-fine hierarchical representation of the intensity profile of a 1D image. An imaginary super-contour exists at the root of this tree. This contour contains all the real contours in the SSI. Each real contour has zero or more children, i.e., contours which exist inside it, and zero or more siblings, i.e., contours which share the same parent. Every closed contour has a peak, a right branch and a left branch. Open contours have only one branch. While rabbit-ears contours may appear to have a peak, a spike contour can proceed upward indefinitely, and is truncated at an arbitrary height (above the rest of the contours).

In principle, our algorithm is similar to that implemented by Mokhtarian and Mackworth [1984]. It is a modified Uniform Cost algorithm [Nilsson, 1971]. It finds the lowest cost match between contours in the two SSIs. In most cases, the two tallest closed contours in each SSI will match. An exception will occur when a higher contour exists on the right hand side of the farther SSI (see §5.3 (b)). The sequential order of the contours and several parameters corresponding to uniform scaling and translation are used for the measurement of the "goodness" of a match.

##### 5.4.1. Establishing the Initial Matching Nodes

The algorithm starts out by selecting the highest closed contour in the closer SSI, and pairs it up with all closed contours of the farther SSI, in height order. Each such pair is exploited and evaluated in terms of the cost of the match. There are two scale space transformation parameters mapping one contour to another which need to be computed for each node. These two are the

scale  $k$  and shift  $d$  parameters. The relationship between the old coordinates,  $x$  and  $\sigma$ , and the new coordinates,  $x'$  and  $\sigma'$ , is as follows:

$$\begin{aligned}x' &= kx + d \\ \sigma' &= k\sigma\end{aligned}$$

Only one pair of points is needed to compute  $k$  and  $d$ . These can be computed using the coordinates of the peaks of the two contours since peaks provide a pair of points on the contours which correspond to each other. Since for open contours peaks are not reliable, these parameters are computed for the x-coordinate of their single branch.

The contours of the farther SSI are always mapped onto those of the closer SSI. As a result, contours don't shrink and matching errors are better accounted for. The same set of parameters is used to match the next pair of contours when a node is expanded.

The cost of match between two contours is defined as the average distance between them after one of them has been transformed. The average distance between two contours is the average of the distances between the peaks, the right branches and the left branches. To reduce the computational cost of finding the lowest cost node, an initial cost is assigned to each node. This "penalty" is based on prior knowledge about the system (and thus, will be different for conventional stereo). Specifically, we make use of the following two observations:

- (a) In an ideal case, all contours will match in sequential (along the X-axis) order. Thus, if the two candidates do not have the same sequential order, the node is penalized. The size of this portion of the initial cost is a linear function of the gap in sequence.

- (b) In most cases, the magnification in height of corresponding closed contours is roughly the same ratio as that between their positions along the X-axis. If the two ratios differ, a penalty, directly proportional to that difference, is added to the initial cost.

Open contours are ignored at the initial state of creating the queue of nodes. However, if the closer SSI does not contain a closed contour, the system will try to match existing open contours.

#### **5.4.2. Expanding a node**

The algorithm proceeds by finding the lowest cost node from the queue of initial nodes and expanding it. The cost the expanded node is computed and added to the previous cost. Then the new node is added to the queue and this process is repeated until the lowest cost node can not be expanded any more. The correct match is assumed to be the one indicated by this node.

In order to expand a node, the next pair of contours in the SSIs to be matched must be selected. This is done in sequential order, both directions of the top contour of the node. As the contours are paired up, they are edited as follows:

- (a) Check for corresponding parents. If this is not the case, it is assumed that the contour of the closer SSI represents extra detail and thus is ignored. One should note that this can also occur due to occlusion which is not allowed in this project.
- (b) Having established that their parents correspond, check if the contours are of compatible types. If they are not, two possibilities are considered:



- (i) As in case (a), we have an occurrence of extra detail in the closer SSI. The same action is taken.
- (ii) It is the case where a spike contour is moving about the X-axis of the SSI (see §5.3 (e)). An appropriate error message is printed.
- (c) Having established that their parents and type correspond, check if the peak ratio of the candidate pair is below some threshold value in comparison with the top pair of the node. If so, the candidate contour from the closer SSI is considered extra detail and is ignored.

If none of the above is encountered, the two candidate contours are accepted as a pair and the next two in line become the new candidates. When all the contours in the closer SSI are accounted for, the individual pairs are matched and the cost of the match is added to the node total.

Once the lowest cost node is selected, the individual disparities can be extracted and the depth computed.

#### **5.4.3. Attending to Skipped Open Contours**

We recall that in the stage of setting up the initial nodes only closed contours, if any exist, are considered. Hence, we must retrieve those skipped open contours, if any, pair them up, extract the appropriate disparities and compute the depth.



# CHAPTER 6

## IMPLEMENTATION

### 6.1. COMPUTER SYSTEM

The programs which realize the underlying theory were implemented on a VAX\* 11/780, running a UNIX† 4.2 BSD Operating system.

### 6.2. A COLLECTION OF PROGRAMS

The system is implemented as a collection of independent programs, each of which performs a task logically separating it from the others. The obvious advantage is that any of the programs can be run, tested, debugged and modified without having to touch any of the other programs. Programs should have well-defined input and output so that it is easy to make one program interact with another. Moreover, these programs can be available as public programs on the computer system on which they were implemented. Any other user of the system who is working on a topic which requires similar tasks to be carried out, might be able to use one or more of these programs.

### 6.3. PROGRAMMING LANGUAGES

The programming languages C and LISP were used to implement all the programs which realize this thesis. C was chosen for programs which have

---

\* VAX is a Trademark of DEC.

† UNIX is a Trademark of Bell Laboratories.

numeric flavor: transforming the images into polar or complex logarithmic space, constructing the SSIs and building a representation for a SSI. Since a large portion of the computation time is spent doing convolutions (constructing a SSI), it seemed that C would provide the desired efficiency. It is also a language which is well-supported by UNIX.

Since the matching algorithm used has a more symbolic flavor, that portion of the system was implemented in Franz LISP, a dialect of LISP running on UNIX.

## **6.4. SYSTEM DESCRIPTION**

### **6.4.1. System Flow**

The system is engaged in four major tasks (fig. 6.1):

- (1) Transforming the images from cartesian space to either complex logarithmic or polar space.
- (2) Constructing a SSI for each row in a transformed image, and build a representation for it.
- (3) Matching the SSIs of corresponding rows.
- (4) Computing depth from the extracted disparities.

### **6.4.2. Image Transformation**

As pointed out in chapter 3, it is desired to transform the original images, which are conventionally sampled in cartesian space, into polar space. This task is carried out by program *polar*. Optionally, the original images can be

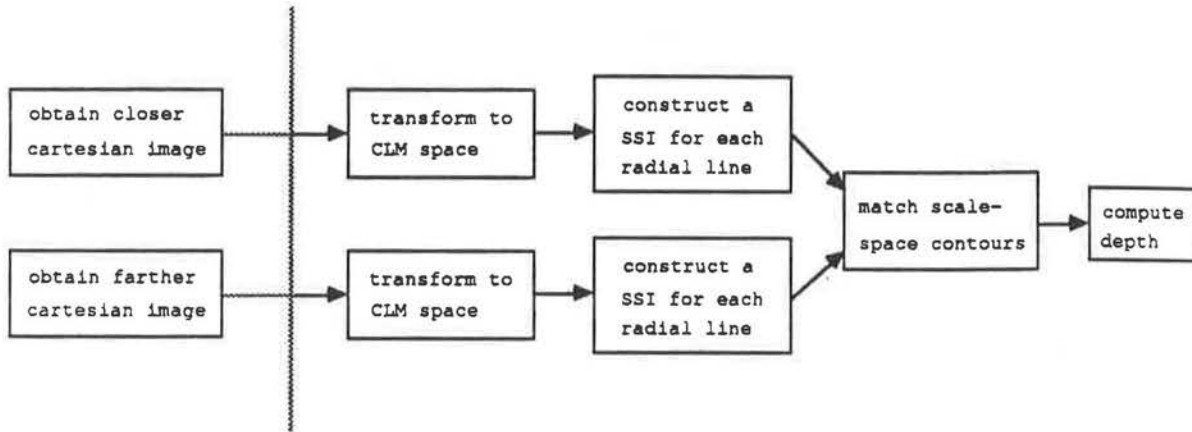


Figure 6.1: the system flow

- 1) the raw cartesian images are transformed into polar space.
- 2) a SSI is constructed for each radial line (row in the polar image).
- 3) corresponding SSIs are matched.
- 4) depth is computed for matching features.

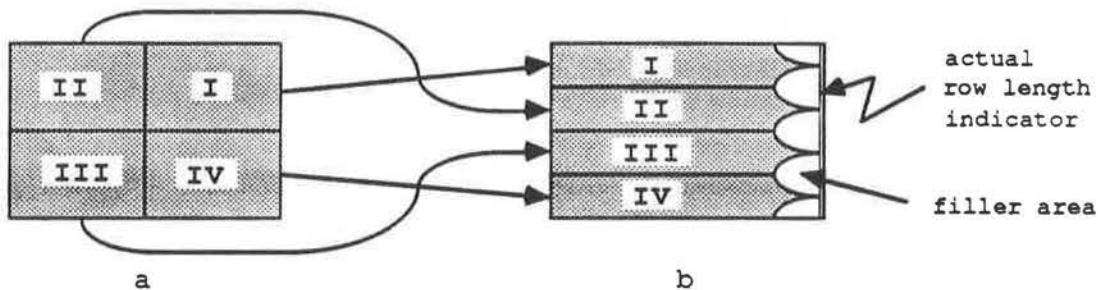


Figure 6.2 : Cartesian-to-Polar mapping.

A rectangular image in cartesian space maps into a fin-like image in polar space. The 4 spines correspond to the 4 corners. The center of the cartesian image corresponds to the left hand column of the polar image. At the end of each polar row an extra pixel is added to indicate its actual length.

transformed into complex logarithmic space, a task carried out by program *clm*.

A transformed image has a standard image format with the radial distance,  $r$ , aligned with the conventional X-axis and the angle,  $\theta$ , with the conventional Y-axis. Since a rectangular cartesian image does not map into a rectangular image in polar space (fig. 6.2), the polar rows do not have uniform length. Hence, each row in the polar image is given an extra pixel, attached to its end, indicating its actual length. The number of columns in the output image is a function of the resampling interval along the radial lines, and the number of rows depends on the angle interval separating two neighboring radial lines. This angle can be controlled by the user (the default is  $1^\circ$ ).

For reasons given in §3.2.4, the logarithmic part of the transformation is postponed until after correspondence between features has been established. Then, it is applied to these features. Hence, our system employs the program *polar* to perform the transformation task. The sampling interval along a radial line is taken to be the size of an input pixel. This keeps the resolution of the output image closely related to that of the input image. Figure 6.3 gives an example of these two options of transforming a cartesian image.

Two optional interpolation schemes are provided to the user. The first one utilizes cubic convolution which introduces some noise around sharp edges but gives good results elsewhere. The second option, which is pursued in this project, follows the interpolation scheme proposed by O'Brien and Jain[1984]. In this scheme, a window of 3 by 3 pixels around the point in polar space is superimposed on the corresponding location in the cartesian image. The portions of the cartesian pixels covered by the window are added up and divided

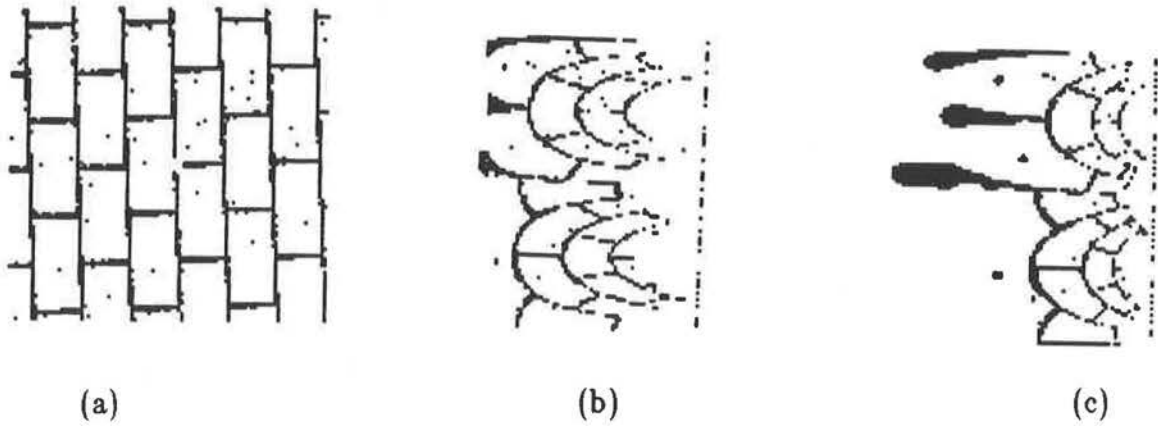
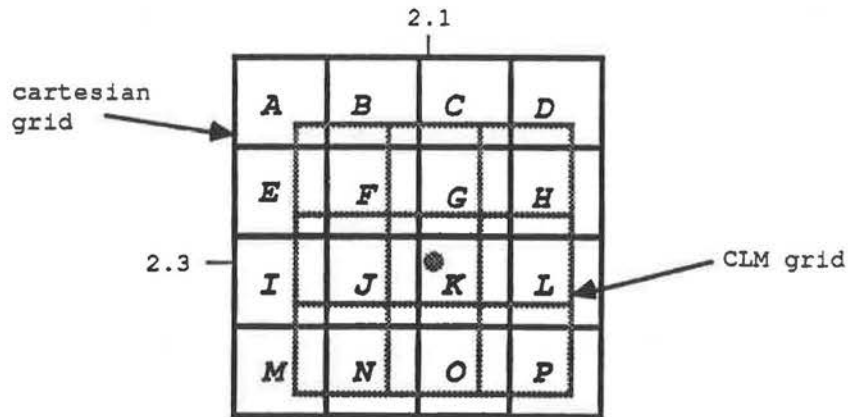


Figure 6.3: CLM versus polar transformations. The cartesian image is shown in (a), its polar transformation in (b) and its CLM in (c).



$$I_{CLM} = (.4) (.2) I_A + (.2) (I_B + I_C) + (.6) (.2) I_D + \dots + (.6) (.8) I_P$$

Figure 6.4: Jain's interpolation scheme.

by the window size. Fig 6.4 gives an interpolation example.

In polar space the width of a pixel grows as its distance from the FOE increases (see chapter 3). When it grows to be larger than two cartesian pixels, an adjustment can be optionally made: each neighboring pixel along a line perpendicular to the median is computed and given a weight. A one dimensional Gaussian distribution function is employed to assign these weights.

### 6.4.3. Constructing SSIs

This phase of the system consists of four programs, each of which is assigned a different task.

#### 6.4.3.1. Program *scale-space*

Program *scale-space* is the most substantial one of the four. It takes as input a 1D image (function) or a specified row in a transformed image, and produces two SSIs in a standard image format. To obtain a SSI, a Gaussian mask with a scale  $\sigma = 1.0$  is convolved with the function and locations of ZCs are found. The value of  $\sigma$  is then repeatedly increased by a small increment,  $\Delta\sigma$  (controllable by the user), and the new filters are convolved with the function. In order to increase the efficiency of the program the filter is convolved with the function only in neighborhoods where ZCs were previously detected. This decreases the CPU requirements of the program significantly.

ZCs are extracted from a 1D convolved image. If the intensity of a pixel in the convolved image is zero and its neighbors are of opposite sign, then the location of the ZC is well defined. But, for two neighboring pixels which carry intensities of opposite signs, the position of the resulting ZC needs to be



interpolated. In this project, two SSI pixels are designated for each input pixel. This minimizes the growth in size of the SSI. The extent of the interpolation has a direct effect on the accuracy of the computed depth. Therefore, in a production system, a finer interpolation scheme must be applied.

One SSI produced is an unsigned binary image which shows, for varying values of  $\sigma$ , the compounded locations of the ZCs. The second SSI has the following additional information:

- the ZCs are signed to indicate their direction.
- their magnitude, which indicates the slope of the convolved function, is specified.

Convolving a 1D image with a Gaussian filter brings up the issue of how to evaluate pixels near the function's end points, for which the filter extends beyond the function domain. There are three basic assumptions one can make about these unknown intensities:

- (a) Assume they are zero. Making such an assumption on both ends of the function introduces three major side effects:
  - two artificial ZCs are placed at the frame boundaries. These ZCs will (usually, but not always) compound into a rabbit-ears contour.
  - all other contours are forced to be closed (see chapter 4), assuming that all image intensities are positive.
  - the shape of these contours can be drastically distorted, depending on the intensity value at each end point of the function. This can bring about some difficulties in obtaining a correct match by the matching algorithm used in this project. However, it will have no effect on the positioning of the contours on the X-axis which is vital

to the depth computations.

- (b) Use the intensity of each end point, repeatedly. This ensures that no new edges are introduced and minimizes the SSI distortion. This is the course taken by our system.
- (c) Assume that the function wraps-around. Although this option is not applicable to our domain, it is added for completeness.

One should notice, that when convolving a row of a polar image, intensities beyond the end point representing the FOE (the left hand side of the row) are known. The system allows the user to take advantage of this information, and only when the periphery of the original image is reached, should one of the above options be selected.

Other option available in the program to the user are:

- control of the increment value for  $\sigma$ ,  $\Delta\sigma$  (default is 1.0).
- set the initial scale (default is 1.0).
- extend the SSI on either side (default is 0).
- control the height at which a spike contour is truncated (default is 20 rows above the next highest contour).
- control the height at which a rabbit-ears contour is truncated (default is 120 rows above the next highest contour).

Since a SSI is built in a coarse-to-fine procedure, the image is, aesthetically, upside-down. Thus, it should be flipped before using it as input to other programs.

#### 6.4.3.2. Filling Gaps in a SSI

Gaps can appear in contours when the change in its slope is large. This happens specially near the peak of closed contours and it is an undesirable feature since a contour must be connected everywhere in order to be recognized correctly. One solution is to use very small increments for  $\sigma$  but this method runs into serious problems with CPU and storage requirements. The solution adopted was to fill such gaps. This ad hoc task is carried out by program *ss-fill-gaps*. A gap is filled by simply joining two loose ends. A loose end is a 1 pixel in a SSI which complies with one of the following conditions:

- has no neighboring 1 pixels.
- has one 1 neighboring pixel with the exception of the first and last rows.
- has two 1 neighboring pixels with whom it forms an L shape and positioned at one end of this shape.

When a loose end is discovered, a search for another loose end starts in a very small neighborhood (two pixels away). This neighborhood is gradually expanded if no other loose ends are found. The search is abandoned when a threshold value of the neighborhood size is reached and no loose ends found.

#### 6.4.3.3. Building a Representation for a SSI

Program *ss-rep* produces a representation for a binary SSI in a hierarchical, top-down order. It starts with the highest contour in the SSI and ends with the smallest. The following information about each contour in the SSI is computed and stored in the representation:

- x-peak*      The height of the peak of the contour.
- y-peak*      The x-coordinate value of the peak of the contour.

- x-rbase* The x-coordinate of the base of either a right branch of a closed contour or the left 'ear' of a rabbit-ears contour.
- x-lbase* The x-coordinate of the base of either a left branch of a closed contour or a spike contour or the right 'ear' of a rabbit-ears contour.
- partial* 0 if the contour is closed; 1 if the it is a left 'ear' of a rabbit-ears contour; 2 if the it is a right 'ear' of a rabbit-ears contour; 4 if it is a spike contour.
- seq* The sequential order of the x-coordinate of the left branch of the contour (lbase). If no left branch then its right one (rbase).
- parent* The number of the contour on top of this contour. If no such contour exists, super-contour is assumed to be the parent (contour number 1).
- children* The contours which are inside this contour.
- y-min* The minimum x-coordinate value of any point on the contour.
- y-mar* The maximum x-coordinate value of any point on the contour.
- x-rb* The heights of the points on the right branch of the contour as traveled from the peak to the end of that branch.
- y-rb* The x-coordinate value of the points on the above right branch.
- x-lb* The heights of the points on the left branch of the contour.
- y-lb* The x-coordinate value of the points on the above left branch.

Additional information about the SSI is recorded in the super contour:

- corr-seq-num* A list of corresponding contour numbers (hierarchical order) and their sequential order.
- last-cont* The total number of contours in the SSI.

<i>log</i>	1 if the image transformation is logarithmic; 0 if equally spaced.
<i>angle</i>	The angle interval used in the image transformation.
<i>xyctr</i>	The cartesian coordinates of the FOE.

The output representation is written in LISP format to be readable by the matching routines.

#### **6.4.3.4. Constructing A Two-Tone Binary SSI**

Program *binary-ss* takes the signed SSI produced by *scale-space*, and computes a different SSI: all pixels preceding a negative ZC (inclusive) are set to 1, and those preceding a positive ZC (exclusive) are set to 0. For an example, see figure 4.18. This SSI is proposed for dealing with the problem of crossing contours (see §5.3(f)). Currently, it is not used in the system.

#### **6.4.4. Matching**

Program *ss-match* matches the scale-space representations (SSR) of corresponding rows in the closer and farther SSIs. Initially, it creates a queue of nodes which correspond to all possible matches between the highest closed contour in the closer SSR and all closed contours in the farther SSR. A match value is computed for each node and always the node which has the lowest cost of match of all is removed from the queue, expanded one step and added to the queue again. This process is repeated until no further expansion is possible. See chapter 5 for more details about the matching algorithm.

A node carries certain information which makes it unique and makes it

possible to expand that node if it is selected. That information is the following:

- contours*            The numbers of the two contours (one from the closer SSR, one from the farther SSR) whose possible match is being considered.
- seq*                 The sequential order of the x-coordinate of the left branch of the above contours. If no left branch then its right one.
- match*              The cost of match for this node so far.
- parameters*        The parameters that take contours in the farther SSR to the contours in the closer SSR. In addition, parameters needed to extract disparities are specified.
- second-pairs*      A list of other paired contours of the expanded node, their sequential order and their corresponding parameters. Since only closed contours are considered at the initial stage, once a match is established, skipped open contours, if any, and their corresponding parameters are added to this list.
- cpath*              The path from the root contour of the closer SSR to the last contour in that SSR which was matched in the last time this node was expanded.
- fpath*              The path from the root contour of the farther SSR to the last contour in that SSR which was matched in the last time this node was expanded.
- cskip-path*        The path of open contours of the closer SSR that have been skipped so far, and not yet been retrieved and added to *second-pairs*.
- fskip-path*        The path of open contours of the farther SSR that have been skipped so far, and not yet been retrieved and added to

- second-pairs.*
- cnum*            The number of the contour pairs that have been matched inside this node so far.
- cc-relevant*    The path of contours of the closer SSR yet to be matched and added to *second-pairs*.
- fc-relevant*    The path of contours of the farther SSR yet to be matched and added to *second-pairs*.

#### 6.4.5. Computing depth

Program *coaxial-stereo* takes the minimum cost node selected by *ss-match* and computes the depth for each pair of matching contours. The depth is computed from the parameters associated with each pair for each branch of the matching contours. Closed contours have two such points and open contours one.

If the transformation of the images was logarithmic, the depth is computed directly from the x-coordinate shift of a projected scenic point from its position in the farther SSI to that in the closer SSI (see chapter 3). The distance by which the camera has traveled is taken to be a single distance unit ( $\Delta z_0=1$ ). A factor  $\frac{\Delta z_0}{2}$  is subtracted, since the calculated depth is that of the average distance between the object and the two camera positions. If the transformation was not logarithmic, a logarithmic transformation is applied to the selected projection points, and then, the depth is computed as above.

Optionally, for polar transformation, the depth can be computed by triangulation, given the distance the camera has moved between the frames and its focal length.





# CHAPTER 7

## EXPERIMENTS

### 7.1. INTRODUCTION

The lack of access to suitable imaging equipment, capable of complying with the particulars of coaxial stereo geometry, forced us to construct several synthetic images in order to test the system. Appendix IV provides the formulae used to build these images.

Note that the depth-map of a scene is always computed for the closer image. In all of the described experiments, an angle interval of  $5^\circ$  is used when transforming images from cartesian to polar space. Hence, all generated polar images have 72 rows.

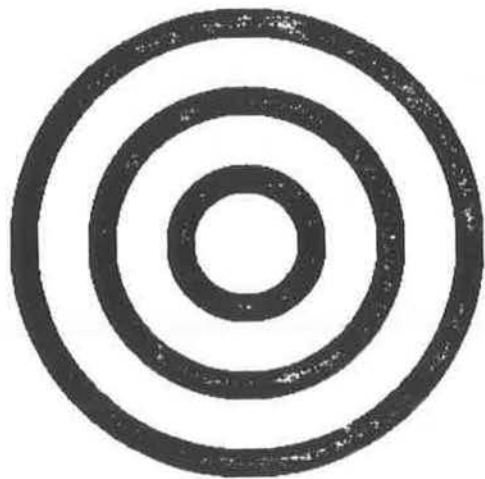
### 7.2. A TWO DIMENSIONAL SCENE

We chose to start with a set of 2D images (fig. 7.1), which allows us to carry out an error analyze of the system. A spatial resolution of 384 by 384 was used for these images.

The scene is composed of three rings each of which is 5cm wide. A gap of 10cm separates the rings. The inner rim of the first ring is placed 10cm from the center. The image in figure 7.1a is taken from a distance of 80cm, the one in 7.1b from 100cm and that in 7.1c from 120cm. The results of transforming these images into polar space are shown in figure 7.2.



(a)



(b)



(c)



Figure 7.1: the cartesian images of the 2D scene.

Figure 7.2: the polar images of the 2D scene.

### 7.2.1. Experiment #1

Images 7.2a and 7.2b were selected for the first experiment. Figure 7.3 shows the ZCs at scale 1.0 selected for the closer image. The intensities of these pixels were set to the computed depth. The resultant depth-map computed for the scene is illustrated in figure 7.4. Figure 7.4a shows the depth-map with a tilt of  $90^\circ$ , and 7.4b with a tilt of  $40^\circ$ .

The SSIs of all the radial line in this particular scene are, theoretically, identical. Variations occur due to resolution limitations in plotting the original images. Figure 7.5 shows a typical set of SSIs of two corresponding radial lines. The depth values computed after matching these SSIs are summarized in table 7.1.

SCALE		SPACE		computed depth		actual depth		error (%)	
contour num	contour seq	branch		branch		branch		branch	
close	far	close	far	left	right	left	right	left	right
2	2	1	1	78	NIL	80	NIL	2.5	NIL
5	3	4	4	80	NIL	80	NIL	0	NIL
4	5	3	3	80	79	80	80	0	1.25
3	4	2	2	80	79	80	80	0	1.25

Table 7.1: the results of matching the SSIs in experiment #1 (see text).

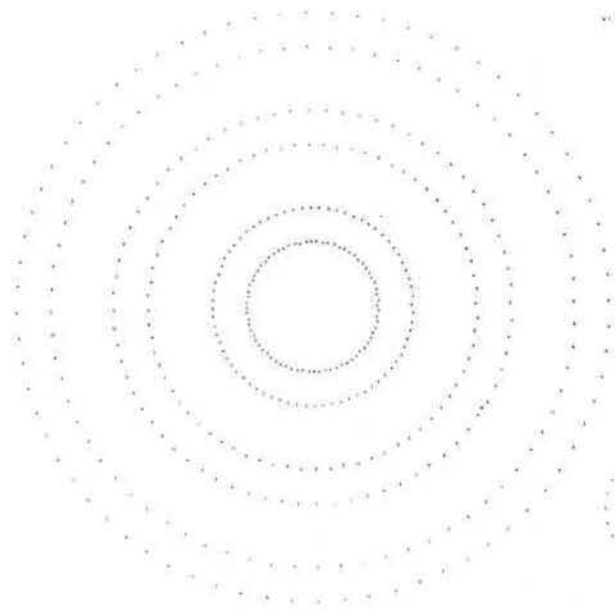


Figure 7.3: the ZCs extracted from the closer image.

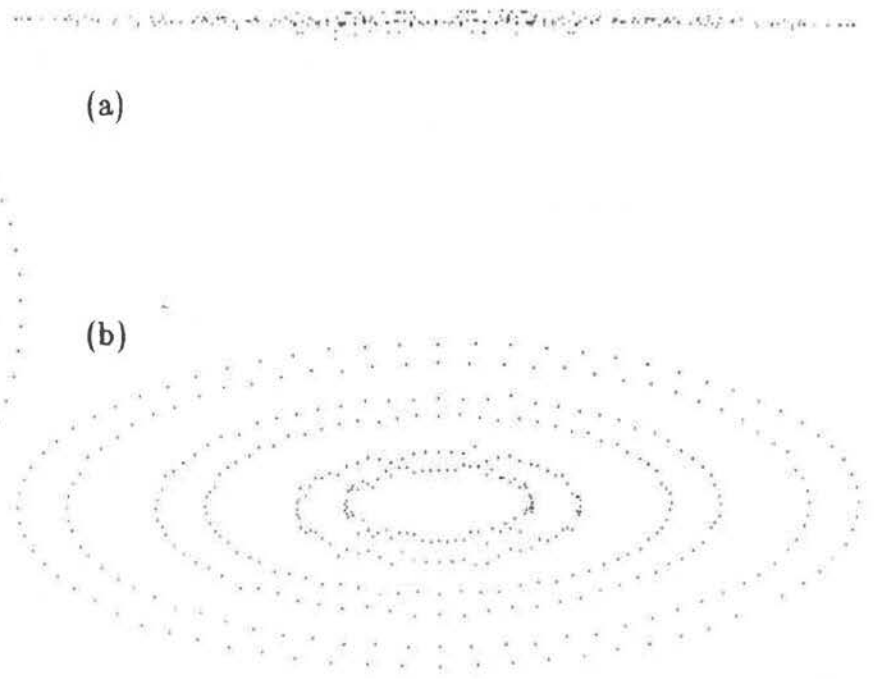


Figure 7.4: the depth map tilted 90° in (a) and 40° in (b).

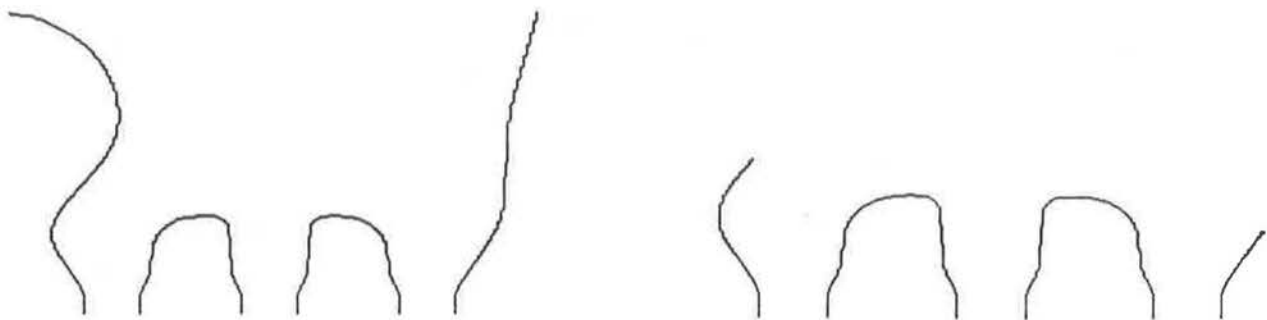


Figure 7.5: a typical set of SSIs.

### 7.2.2. Error Analysis

Picture elements must be quantized when stored in a digital machine. As a result of this limitation, the original images are plotted with a  $\pm \frac{1}{2}$  pixel accuracy. Since depth is computed from the shift in pixels' position, an error may occur. The error magnitude is inversely proportional to the image resolution. The image transformation to polar space is done inversely (see § 3.2.4 and § 6.4.2). Errors can be introduced due to the smoothing effect of the interpolation performed on the intensity values of pixels in the cartesian image. However, this is difficult to measure or estimate and, thus, is not taken into account in this analysis. Another error may take place because of the poor interpolation ratio used in setting the ZCs locations when the SSI is constructed. This error does not add up to the other errors and, hence, does not effect the range of possible errors.

Since each pixel in a radial line is represented by two columns of pixels in the SSI, it implies that each scale-space branch is allowed to move about the X-axis no more than  $\pm 1$  pixel. Hence, when a small shift in position translates to a large change in depth (close to the FOE) a large error can occur.

Let us go back to the example given in figure 7.5 and table 7.1. If we take the first pair of contours in the table (both of sequence 1), they correspond to a pixel placed 40 pixels away from the center in the closer image and 32 pixels away in the farther image. In the corresponding SSIs this translates to the pair (80,64), and computes to a depth of 80cm (79.63). This is also the actual depth. To compute the whole range of possible error in the computed depth, we allow each branch to move 1 pixel away from each other. Thus, to obtain the error range for this example, we compute the depth for the

pair (81,63) which is 70cm (69.58) and for the pair (79,65) which is 93cm (92.53). Hence, the error range for this location, which is near the FOE, is quite large: from +16.25% to -12.5%, a total of 28.75% of the actual depth.

As we move away from the FOE, this improves at a rate of  $\frac{1}{r}$ , where  $r$  is the distance from the FOE. Let us take the contour sequenced 4 in the above example. This contour should be placed 360 pixels to the right of the origin in the closer SSI and 288 pixels in the farther one. This again renders a depth of 80cm (79.63). The maximum error will occur for the pairs (361,287) which gives a depth of 77cm (77.19), and the pair (359,289) with a depth of 82cm (82.21). So the errors allowed are reduced significantly to range from +2.5% to -3.75%, a total of 6.25% of the actual depth.

### 7.2.3. Experiment #2

Images 7.2a and 7.2c were selected for the next experiment, in which we examine the impact of increasing the distance from which the farther image is taken.

As in experiment #1, we will analyze the same two pairs of contours. The first contour should be placed 53.63 pixels to the right of the origin in the farther SSI. It is placed 54 pixels to the right and, hence, we get an error in the computed depth: 82cm (81.77). The smallest possible depth is computed from the pair (81,53): 74cm (74.30), and the largest from the pair (79,55): 90cm (90.46). Hence, compared with experiment #1, the error allowed is reduced to range from +12.5% to -7.5%, a total of 20% of the actual depth. Still bad, but nevertheless, better.

The other contour (sequence 4) is placed 242 pixels to the right of the farther SSI (should be at 241.32 pixels). This translates to a depth of 81cm (80.71). The smallest possible depth is computed from the pair (361,241): 79cm (78.99), and the largest from the pair (359,243): 82cm (82.496). So again, the error allowed is reduced to range from +2.5% to -1.25%, a total of 3.75% of the actual depth.

Hence, moving the camera farther away to obtain the second image reduces the possible error range. Notice that the improvement is similar for features close to the FOE and for those farther away from it.

#### **7.2.4. Experiment #3**

Images 7.2b and 7.2c were selected for this experiment, in which we examine the impact of selecting a deeper point in the scene relative to the two camera positions.

Again the same analysis is followed. The first pair of contours are set at (64,54). The depth rendered is 108cm (107.72). The smallest possible depth is computed from the pair (65,53): 88cm (97.99), and the largest from the pair (63,55): 137cm (137.27). So, the error allowed ranges from +37% to -12%, a total of 49% of the actual depth. This is a considerable increase compare to the results of experiment #1.

For the other contour (sequence 4) we have the pair (288,242) which translates to a depth of 105cm (104.93). The smallest possible depth is generated by the pair (289,241): 100cm (100.11), and the largest by the pair (287,243): 110cm (110.18). Again, the error allowed is increased to range from +10% to 0%, a total of 10% of the actual depth.

Hence, by comparing this experiment and the first one, we can conclude that as the depth of an object increases so does the range of possible errors. The increase is similar for regions close to the FOE and those farther away from it. A comparison with the second experiment suggests that a larger distance between the two camera position yields much better results.

### **7.2.5. Conclusions**

To minimize the possible error range, one should:

- maximize the resolution of the original images.
- minimize the distance of the closer camera position.
- maximize the distance between the two camera positions.

## **7.3. THREE DIMENSIONAL SCENES**

### **7.3.1. Round Cake Scene**

For the next experiment a round cake composed of four layers was constructed. The images are 350 by 350 pixels. The scene dimensions and viewing axis are illustrated in figure 7.6.

One image was taken from a distance ( $d$ ) of 60cm and the other from 80cm. The images and their corresponding polar transformations are shown in figures 7.7 and 7.8 respectively. A typical set of SSIs is shown in figure 7.9 and the depth computed by matching these SSIs is summarized in table 7.2. A complete depth map is given in figure 7.10. The depth map is tilted  $90^\circ$  in figure 7.10a,  $30^\circ$  in figure 7.10b and  $45^\circ$  in 7.10c.



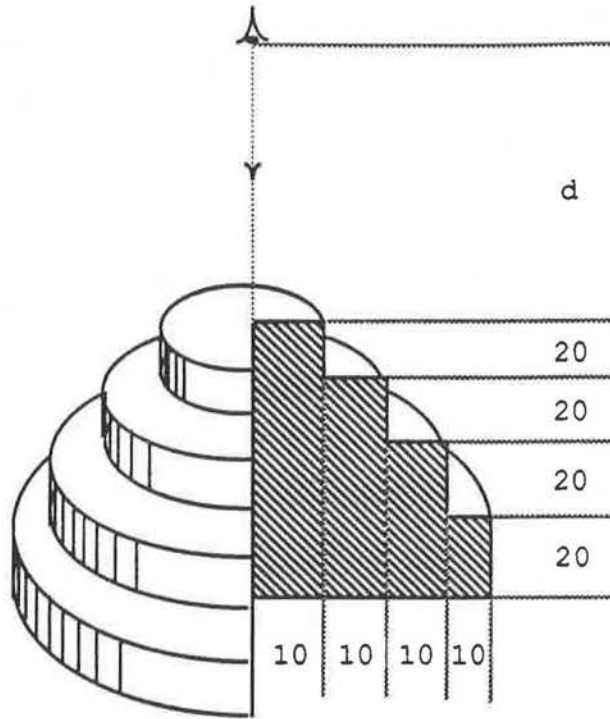
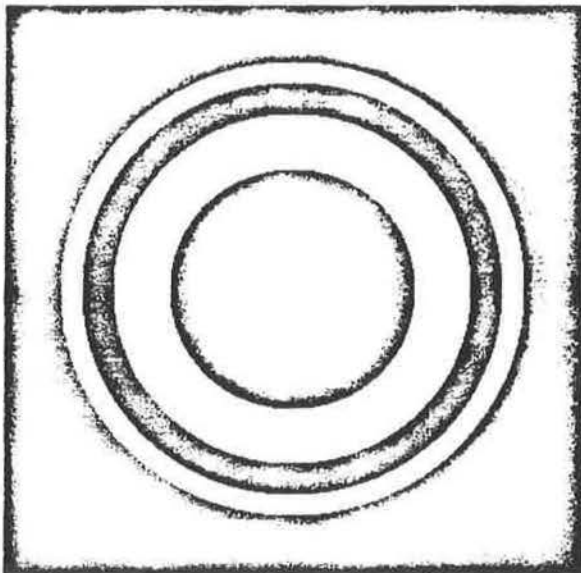
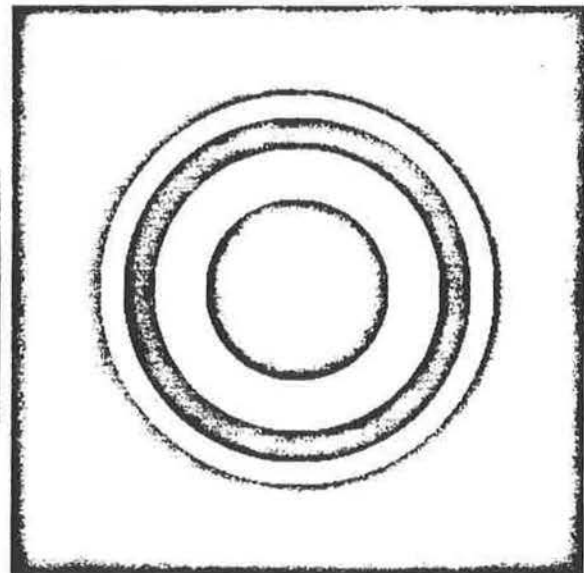


Figure 7.6: round cake scene



(a)



(b)

Figure 7.7: the cartesian images of the round cake scene.



Figure 7.8: the polar images of the round cake scene.

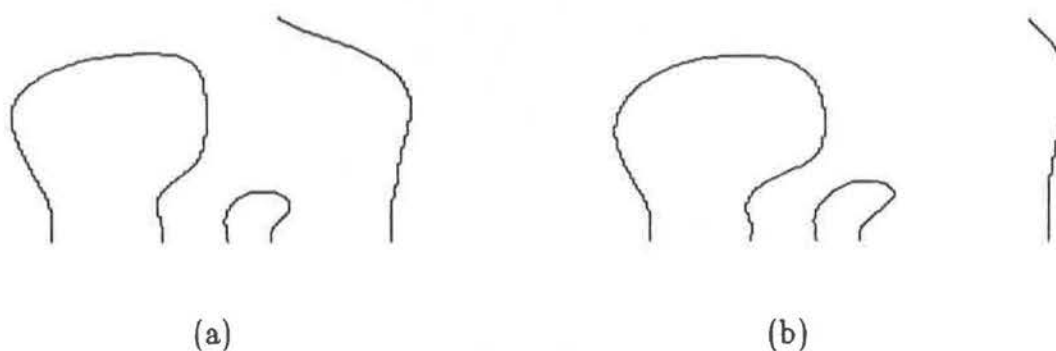


Figure 7.9: a typical set of SSIs.

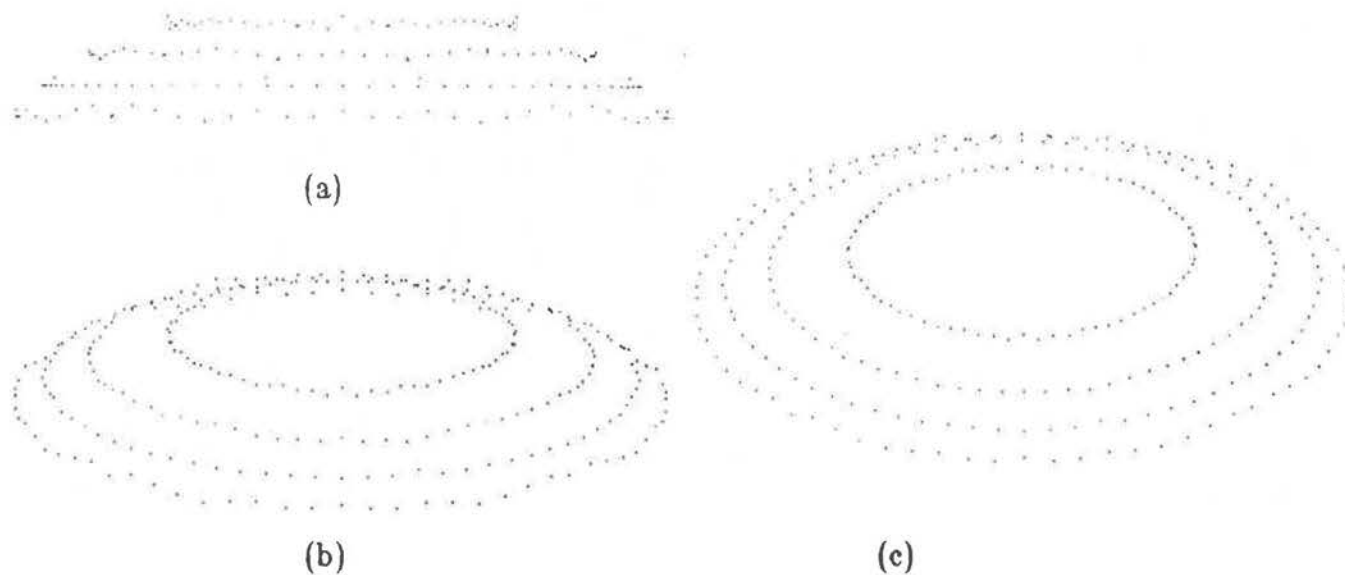


Figure 7.10: the depth map tilted  $90^\circ$  in (a),  $45^\circ$  in (b) and  $30^\circ$  in (c).

An interesting side result can be observed in this particular experiment. The SSIs in figure 7.9 show five ZCs at a fine  $\sigma$  value (i.e., before any closed contours are formed). The radial lines that gave rise to these SSIs contain only four intensity changes. This phenomenon characterizes step functions (see chapter 4) and the 'extra' ZC is referred to as *the phantom zero-crossing*.

SCALE		SPACE		computed depth		actual depth		error (%)	
contour num		contour seq		branch		branch		branch	
close	far	close	far	left	right	left	right	left	right
2	2	3	3	NIL	NIL	0	NIL	0	NIL
4	4	2	2	101	119	100	120	1	0.83
3	3	1	1	62	82	60	80	3.33	2.5

Table 7.2: the results of matching SSIs generated in the round cake scene experiment (see text).

This ZC proved to be phantom in our system as well: as the camera moved along its optical axis no shift along the X-axis was recorded in the position of the branch containing these ZCs. This interprets to a depth value of zero, and having no depth means being phantom.

### 7.3.2. Inverted Square Cake Scene

An inverted square cake was constructed for this experiment. Its dimensions are illustrated in figure 7.11.

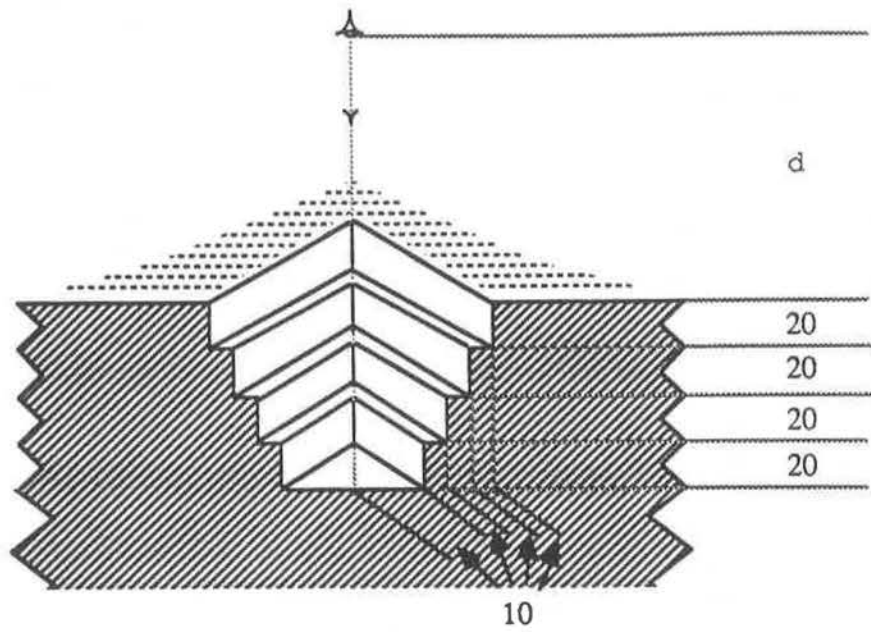
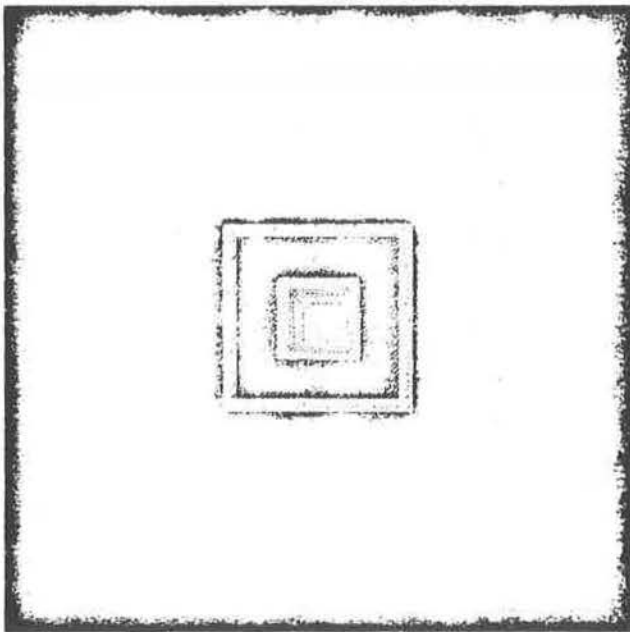
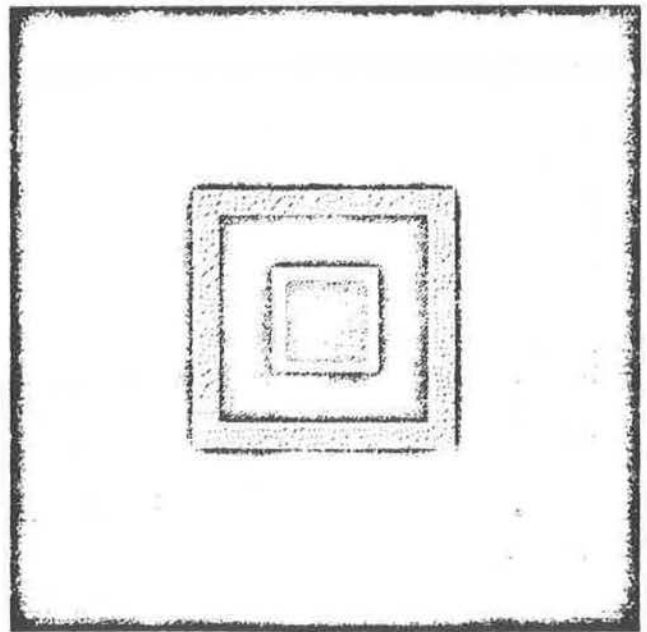


Figure 7.11: square cake scene



(a)



(b)

Figure 7.12: the cartesian images of the square cake scene.

In the previous experiments round objects were used because a circle, whose center is at the FOE, transforms into a straight line in polar space. However, any arbitrary shaped objects can be processed by the system just as well. This scene gives an example to this effect.

Following the concluding remarks of the analysis of the 2D scene (in § 7.2), we worsen the conditions in this experiment compared to the previous one. That is, edges located closer to the FOE are set to have a greater depth. Hence, the error range in this region is quite large, while in the periphery it is quite small.

SCALE		SPACE		computed depth		actual depth		error (%)	
contour num		contour seq		branch		branch		branch	
close	far	close	far	left	right	left	right	left	right
2	2	4	4	60	NIL	60	NIL	0	NIL
4	4	2	2	113	110	100	100	13	10
5	5	1	1	NIL	125	NIL	120	NIL	4.16
3	3	3	3	82	84	80	80	2.5	5

Table 7.3: the results of matching SSIs generated in the square cake scene experiment (see text).

The pair of input images are assumed to be taken from 60cm (fig. 7.12a) and 90cm (fig. 7.12b). Their corresponding polar images are shown in figure 7.13. The SSIs of the first row in each polar image (angle 0°) are given in



Figure 7.13 the polar images of the square cake scene.

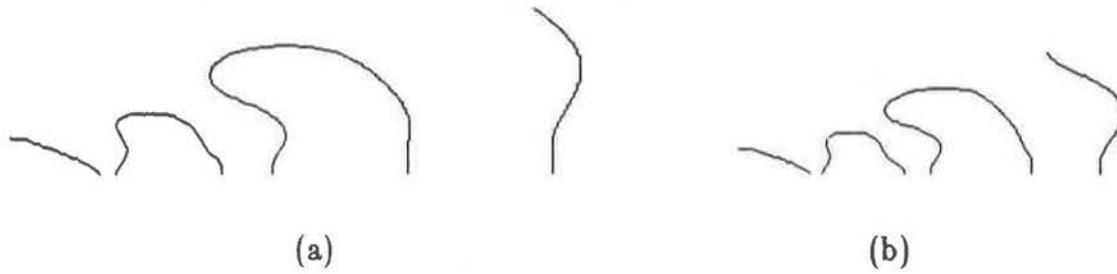


Figure 7.14: a typical set of SSIs.

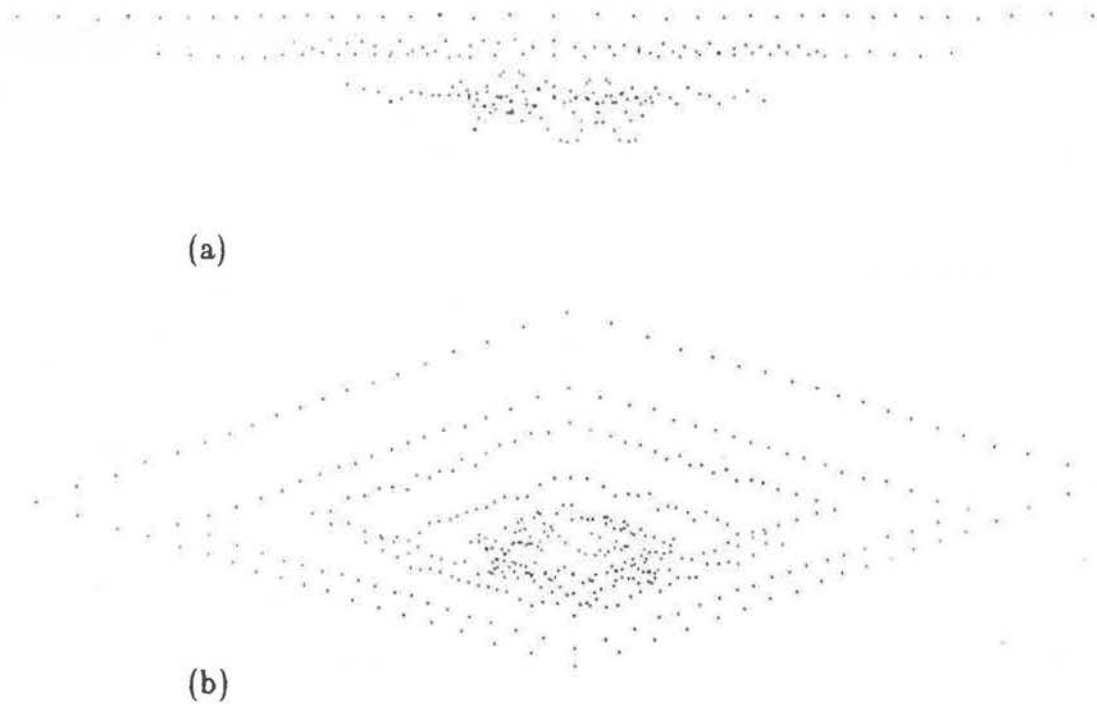


Figure 7.15: the depth map tilted 90° in (a) and 45° in (b).

figure 7.14, and the results of their match-up is summarized in table 7.3. The depth map is depicted in figure 7.15 in two tilted displays, one at  $90^\circ$  and the other at  $45^\circ$ .

### 7.3.3. Occlusion and Loss of Information

For this experiment the scene illustrated in figure 7.16 was constructed. Images are taken from one of each of the following distances: 90cm, 60cm, 40cm, 30cm (fig. 7.17). Their corresponding polar transformations are shown in figure 7.18. The scene contains four visible (i.e. not occluded) edges. They give rise to four ZCs in the SSI, at a fine  $\sigma$  level. The edges are denoted as follows:

- E1: the top edge of the filled cylinder.
- E2: the inside bottom edge of the ring.
- E3: the inside top edge of the ring.
- E4: the outside top edge of the ring.

We denote the instantaneous projections of each of the edges upon the image plane by P1, P2, P3, and P4, respectively.

The initial distance of the camera from the nearest point in the scene is 90cm. All four edges are visible (fig 7.17a). As the camera moves along its optical axis towards the scene, the projections of the edges upon the image plane move outwards relative to the FOE ( $\equiv$ optical flow). The projection of points which are closer to the camera generate a faster optical flow velocity than points that are farther away from it. Hence, P1 has the fastest optical flow velocity, P3 and P4 have a slower rate and P2 the slowest. Thus, as the camera motion persists, P1 moves closer to P2. At a distance of 60cm P1 and P2

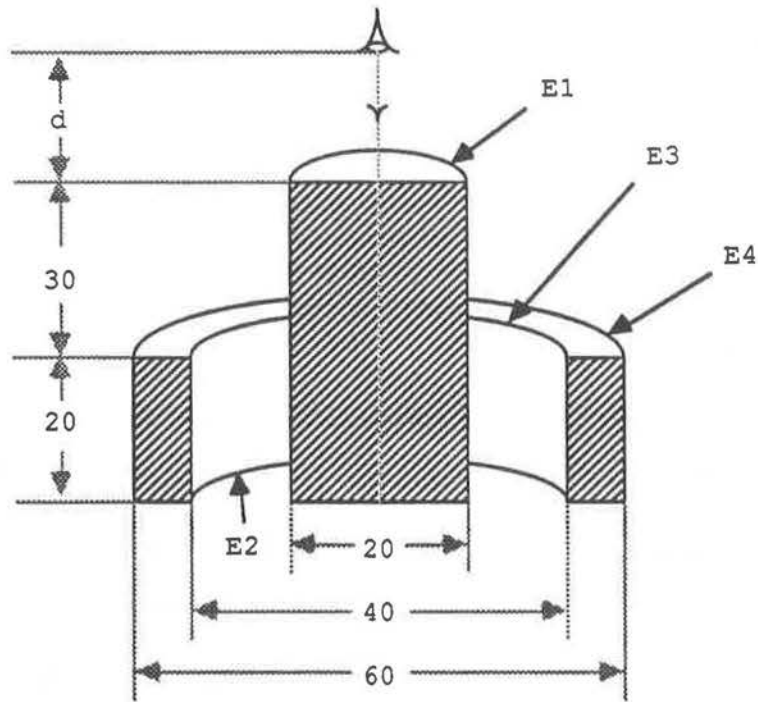


Figure 7.16: a 3D scene containing a cylinder and a ring

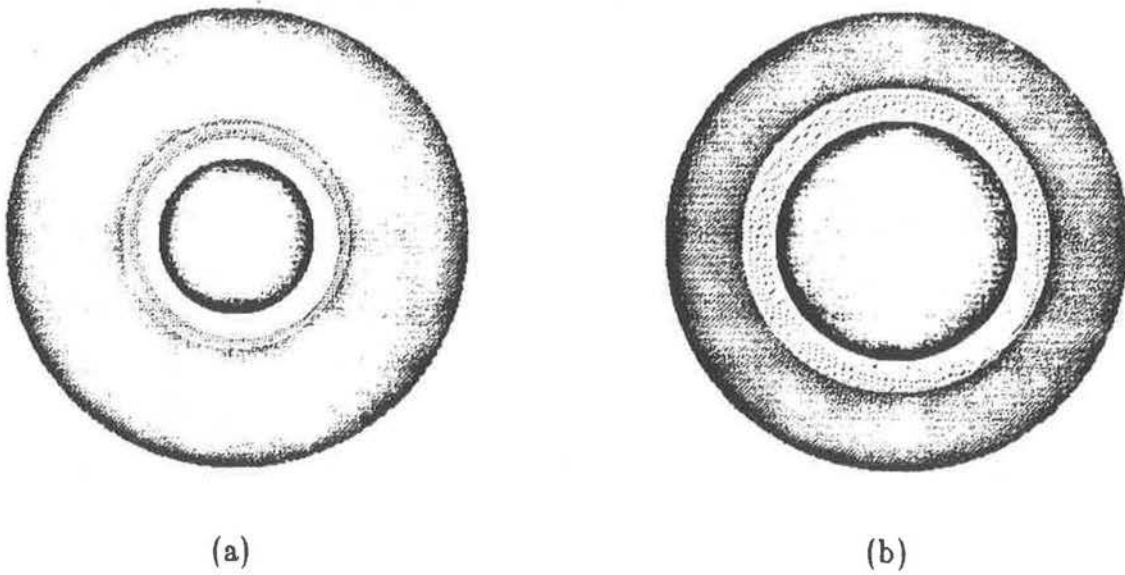
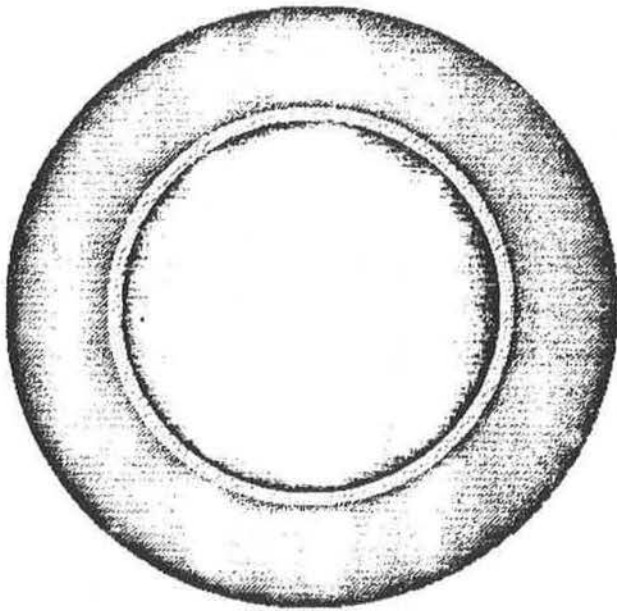
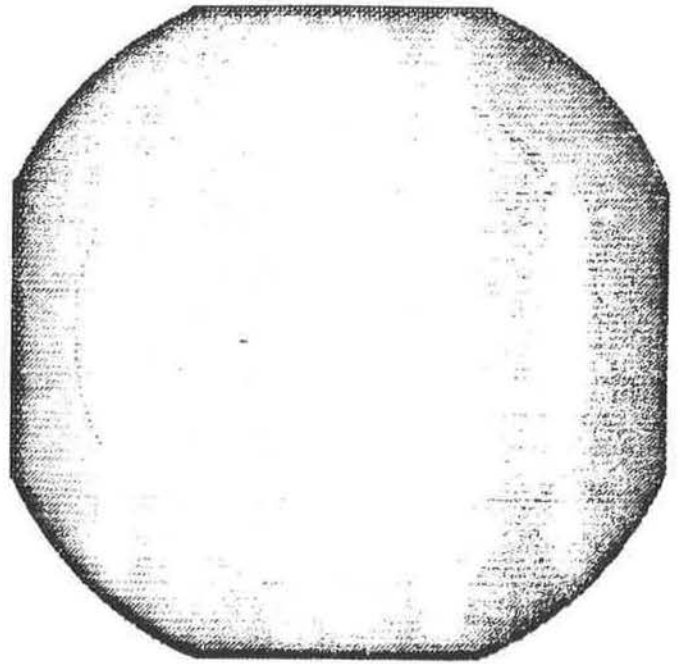


Figure 7.17: the cartesian images of the ring and cylinder scene.



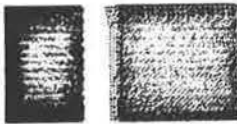


(c)

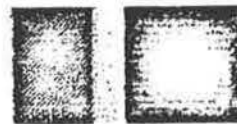


(d)

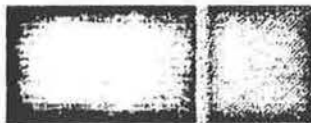
Figure 7.17 (continued)



(a)



(b)



(c)



(d)

Figure 7.18: the polar images of the ring and cylinder scene.

are very close, but still distinguishable (fig. 7.17b). Shortly after that, the two projections overlap and E2 becomes occluded to the viewer. Only three edges remain in view, a situation which is captured in the image viewed from 40cm (fig. 7.17c). As the camera continues to move, P1 closes in on P3 and P4. In the image taken from 30cm (fig. 7.17d), E3 is no longer visible.

As projection points move outwards, they are bound to move out of the image frame at some close distance to the scene. In figure 7.17d we see that E4 is recorded only in the corner regions of the image. So, at angle  $45^\circ$  two edges are showing while at angle  $0^\circ$  only one.

If we look at the SSIs constructed for row 0 in each of the polar images (fig. 7.19), we see a different one for each of them. That is, no SSI can be matched correctly to any of the others. Furthermore, when we compute the SSI for row 9 (angle  $45^\circ$ ) of the image in figure 7.18d, we see yet another SSI that cannot be matched correctly with either of the others.

The SSI in figure 7.19a is not compatible with that in figure 7.19b because the change in the balance of intensities triggered the open contour to move about the X-axis. Figure 7.19c reflects the occlusion of edge E2. Hence, the left branch of the closed contour cannot be matched correctly. The single open contour in figure 7.19d cannot be matched correctly either. In figure 7.19e, it should be noted, one branch in the SSI represents an invisible edge (step function) and, hence, it cannot be matched correctly as well.

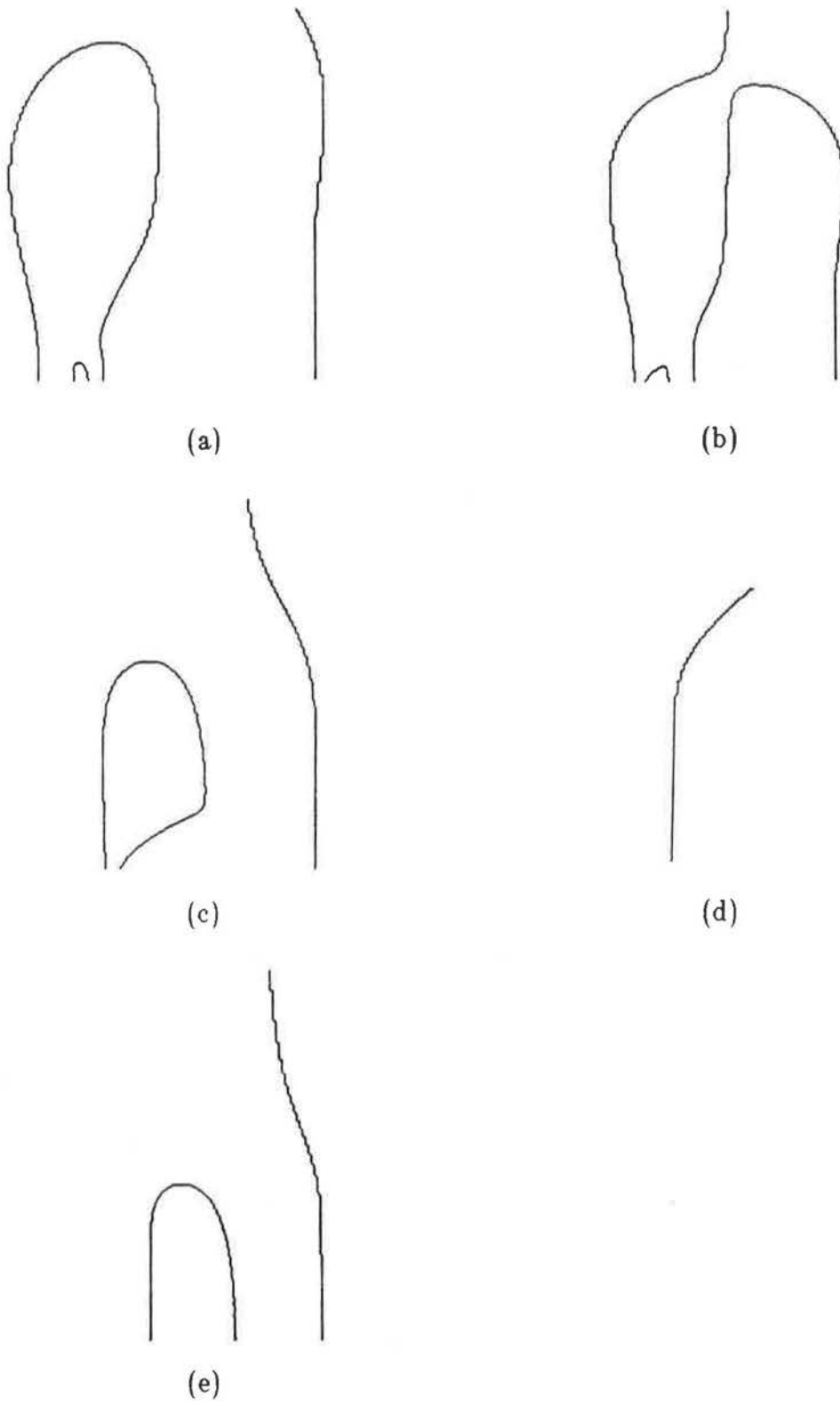


Figure 7.19: the SSIs for the images.



# CHAPTER 8

## EXTENSIONS AND CONCLUSIONS

### 8.1. EXTENSIONS TO THIS PROJECT

The lack of access to suitable imaging equipment, capable of complying with the particulars of coaxial stereo geometry, forced us to construct several synthetic images in order to test the system. Further research will require the usage of real images.

#### 8.1.1. A Different Matching Algorithm

We have used in our implementation a top-down algorithm to match SSIs. To minimize distortions in the shape of the scale-space contours, we assumed a repetition of the end point intensities when the frame of an image is reached.

A different approach, which has a stabilizing effect on the sensitivity of scale-space contours to the balance of intensities in the input image, is worth while investigating. In this approach one assumes zero intensities for locations outside the image frame. Such an assumption introduces two zero-crossings at the end point positions. These zero-crossings form a rabbit-ears contour in the SSI, which can be ignored in the matching process. All other contours are forced to be closed. The shape of these contours may be distorted considerably, but the vital information for the computation of depth remains intact. Closed contours are easier to match but a different matching algorithm needs

to be developed which will tolerate distortions in the shape of the contours.

One may reduce the above distortion by assuming zero intensities for locations outside the end points, while adding some high constant to all image intensities. This may enable the usage of the matching algorithm as is.

### **8.1.2. Application to Other Stereo Systems**

The matching algorithm, as implemented, can be easily adopted to serve conventional stereo systems as well as other visual expert systems.

### **8.1.3. Multiple Input Images**

It would be more efficient to perform a parallel process on several pairs of input images rather than expanding the matching algorithm to handle more than two images at one time.

### **8.1.4. Occlusion**

This problem, which characterizes most computational vision systems, is not addressed in this project. However, it might be possible to estimate the probability of changes in the structure of a SSI when occlusion occurs. The problem can be eased by matching scale-space branches rather than contours.

### **8.1.5. Moving Contours**

The sensitivity of contours in a SSI to the balance of intensities in the image weakens the robustness of the matching algorithm. By viewing each branch of a closed contour as an open contour that terminates at the contour peak, one can improve that robustness. By matching scale-space branches

rather than contours, the problem can be avoided in most cases.

### **8.1.6. Two-Tone SSIs**

This type of SSI, as suggested in chapter 4, can resolve difficulties that the phenomenon of crossing contours can bring about. This suggestion is not integrated in the current system.

## **8.2. SUMMARY AND CONCLUDING REMARKS**

This project tries to shed some light on issues concerning coaxial stereo and the usage of the scale-space concept to address the correspondence problem.

We have examined a general procedure to compute depth from stereo and looked at some existing stereoscopic and related motion systems.

We described the principles and geometrical aspects of coaxial stereo. This geometry provides better physical constraints than other stereo systems do:

- one solution to the correspondence problem is given (FOE), providing a reference point.
- usually, occlusion will occur only as the camera moves towards the scene. This may take exception if the scene contains a hidden surface, which is revealed as the camera moves in (the inner walls of the hole of a torus, for example).
- all the scenic area captured on the closer image is also recorded on the farther one.

The main deficiency of this configuration is that the depth of the scenic point corresponding to the FOE in the images cannot be determined. In addition to that, the probability for error in the computed depth for points in the images located close to the FOE is quite large.

Complex logarithmic space was reviewed. It provides a simple means to compute depth from the extracted disparities, and is independent of the camera specifications. Ways to utilize its advantages and avoid its disgrations were discussed.

We provided an analysis on the structure of scale-space images of one dimensional functions. We showed that such functions contain in their scale-space image between one and five open contours, and proved the validity of crossing contours in such SSIs.

We went on to examine the difficulties involved in establishing correspondence between two SSIs. A top-down matching algorithm was outlined. This algorithm was adopted from Mokhtarian and Mackworth [1984], and was modified for this application.

The last example (§ 7.3.3) shows that, although the basic concepts of coaxial stereo are sound, the top-down scale-space matching algorithm is not always the most appropriate matching algorithm to use. Modifications along the lines of § 8.1.5 (and perhaps others) are required.

We described a system implemented to deduce depth from coaxial stereo. The system employs delayed complex logarithmic mapping and a scale-space matching algorithm.

A series of experiments were conducted with synthetic images. We showed that the potential for errors is by far larger for regions closer to the FOE than



for those located near the frame boundaries. In addition, we concluded that one should minimize the distance between the scene and the closer camera position, and maximize the distance interval between the two camera positions. These are limited by the focal length of the camera, the dimensions of the analyzed scene and the resolution of the images. As is true for most computational vision tasks, higher image resolution reduces the magnitude of occurring errors.



## References

1. R. D. Arnold, "Local Context in Matching Edges for Stereo Vision," *Proc. of APRA Image Understanding Workshop*, Science Applications Inc., Boston, Mass., May 1978.
2. R. D. Arnold, *Automated Stereo Perception*, Ph.D. Thesis, Stanford AI Lab., AIM-351, March 1983.
3. W. H. Baker and T. O. Binford, "Depth from Edge and Intensity Based Stereo," *IJCAI-81 Proc.*, pp. 631-636, Vancouver, B.C., 1981.
4. W. H. Baker, *Depth from Edge and Intensity Based Stereo*, Stanford AI Lab., AIM-347, Sept. 1982.
5. D.H. Ballard and C.M. Brown, *Computer Vision*, Prentice Hall, Englewood Cliffs, New Jersey, 1982.
6. S. T. Barnard and W. B. Thompson, "Disparity Analysis of Images," *IEEE Trans. on PAMI*, vol. 2, no. 4, pp. 333-340, July 1980.
7. S. T. Barnard and M. A. Fischler, "Computational Stereo," *Computing Surveys*, vol. 14, no. 4, pp. 553-572, Dec. 1982.
8. D. Casasent and D. Psaltis, "Position, Rotation, and Scale Invariant Optical Correlation," *Applied Optics*, vol. 15, no. 7, pp. 1795-1799, 1976.
9. P. Cavanagh, "Size and Position Invariance in the Visual System," *Perception*, vol. 7, pp. 167-177, 1978.
10. P. Cavanagh, "Size Invariance: Reply to Schwartz," *Perception*, vol. 10, pp. 469-474, 1981.
11. R. V. Churchill, *Complex Variable and Applications*, McGraw-Hill, New York, 1960.
12. J. J. Clark and P. D. Lawrence, "A hierarchical Image Analysis System Based Upon Oriented Zero Crossings of Bandpassed Images," in *Multiresolution Image Process-*

- ing and Analysis*, ed. A. Rosenfeld, pp. 148-168, Springer-Verlag, Berlin, 1984.
13. J. L. Crowley and R. M. Stern, "Fast Computation of the Difference of Low-Pass Transform," CMU-RI-82-18, The Robotic Institute, Carnegie-Mellon University, Nov. 1982.
  14. D. G. DeGryse and D. J. Panton, "Syntactic Approach to Geometric Surface Shell Determination," *SPIE*, vol. 238, pp. 264-272, 1980.
  15. B.V. Funt, "Whisper: a computer implementation using analogues in reasoning," TR-76-09, Computer Science Dept., University of British Columbia, Vancouver, Canada, 1976.
  16. D. B. Gennery, *Modeling the Environment of an Exploring Vehicle by Means of Stereo Vision*, Ph.D. Thesis, Stanford AI Lab., AIM-339, June 1980.
  17. J.J. Gibson, *The perception of the visual world*, Houghton Mifflin Co., Boston, Mass., 1950.
  18. J.J. Gibson, *The senses considered as perceptual systems*, Houghton Mifflin, Boston, 1966.
  19. J.J. Gibson, *An Ecological Approach to Visual Perception*, Houghton Mifflin, Boston, 1979.
  20. G. L. Gimel'farb, V. B. Marchenko, and V. I. Rybak, *An Algorithm for Automatic Identification of Identical Sections on Stereopair Photographs*, pp. 311-322, *Kybernetika*, 1972.
  21. W. E. L. Grimson, *From Images to Surfaces: A Computer Study of Human Early Visual System.*, MIT Press, Cambridge MA., 1981.
  22. W. E. L. Grimson, *Computational Experiments with a Feature Based Stereo Algorithm*, MIT AI-Memo 762, Cambridge, MA, January, 1984.
  23. S. C. Gupta, *Transform and State Variable Methods in Linear Systems*, John Wiley and sons, New York, 1966.

24. M. J. Hannah, *Computer matching of areas in stereo images*, Ph.D. Thesis, Stanford AI Lab., AIM-239, July 1974.
25. M. J. Hannah, "Bootstrap Stereo," *Proc. of APRA Image Understanding Workshop*, Science Applications Inc., NTIS Report no. AD-A084 746, Collage Park, Md, April 1980.
26. R. L. Henderson, W. J. Miller, and C. B. Grosch, "Automatic Stereo Reconstruction of Man-made Targets," *Proc. of SPIE, Digital Processing of Aerial Images*, vol. 186, pp. 240-248, 1979.
27. E.C. Hildreth, *Implementation of a theory of edge detection*, MIT AI-Memo 579, Cambridge, MA, 1980.
28. R. Jain, "Direct Computation of the Focus of Expansion," *IEEE Trans. on PAMI*, vol. 5, no. 1, pp. 58-64, 1983.
29. R. Jain, "Complex Logarithmic Mapping and the Focus of Expansion," *SIGGRAPH/SIGART Workshop on Motion: Representation and Perception*, Toronto, April 1983.
30. R. Jain, "Segmentation of Frame Sequence Obtained by a Moving Observer," *IEEE Trans. on PAMI*, vol. 6, p. 5, 1984.
31. G. D. Forney Jr., "The Viterbi Algorithm," *Proc. of IEEE*, vol. 61, no. 3, pp. 268-278, 1973.
32. B. Julesz, *Foundations of cyclopean perception*, University of Chicago Press, Chicago, 1971.
33. H. Kober, *Conformal Mapping*, Dover, New York, 1952.
34. J.J. Koenderink and A.J. van Doorn, "Photometric invariants related to solid shape," *Acta Optica*, vol. 27, no. 7, pp. 918-996, 1980.
35. W. B. Lancia and W.Q. Nicholson, "Passive Determination of Three-dimensional Form from Dynamic Imagery," *Proc. of SPIE, Digital Processing of Aerial Images*,

- vol. 186, pp. 240-248, 1979.
36. D. T. Lawton, "Motion Analysis Via Local Translational Processing," *IEEE Workshop on Computer Vision: Representation and Control*, Rindge, New Hampshire, 1982.
  37. D.N. Lee, "Visual Information During Locomotion," in *Perception: Essays in Honor of J.J. Gibson*, ed. H.L. Pick, pp. 250-267, Cornell University Press, Ithaca, 1974.
  38. D. N. Lee, "A Theory of Visual Control of Braking based on Information about Time-to-collision," *Perception*, vol. 5, pp. 437-459, 1976.
  39. D. N. Lee, "The Optic Flow of Field: The Foundation of Vision," *Phil. Trans. Royal Soc. of London*, vol. B290, pp. 169-179, 1980.
  40. A. L. Lowrie, "Multiple Resolution Row Matching for Stereo Vision," Masters Project Report, Dept. of Electrical and Computer Engineering, CMU, April 1984.
  41. D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 194, pp. 283-287, 1976.
  42. D. Marr and S. Poggio, *A Theory of Human Stereo Vision*, MIT AI-Memo 451, Nov. 1977.
  43. D. Marr and T. Poggio, "A Computational Theory of Human Stereo Vision," *Phil. Trans. Royal Soc. of London*, vol. B204, no. 1156, pp. 301-328, 1979.
  44. D. Marr and S. Ullman, *Directional Selectivity and its Use in Early Visual Processing*, MIT AI-Memo 524, 1979.
  45. D. Marr and E. Hildreth, "Theory of Edge Detection," *Proc. Royal Soc. of London*, vol. B207, pp. 187-217, 1980.
  46. D. Marr, *Vision*, W. H. Freeman and Company, San Francisco, 1982.
  47. J. E. W. Mayhew and J. P. Frisby, *Artificial Intelligence*, vol. 17, pp. 349-385, North Holland Publishing Company, 1981.

48. F. Mokhtarian and A. K. Mackworth, "Scale-based Description of Planar Curves and Two-Dimensional Shapes," UBC Technical Report 84-15, 1984.
49. H. P. Moravec, *Obstacles Avoidance and Navigation in the Real World by Seeing Robot Rover*, Ph.D. Thesis, Stanford AI Lab., AIM-340, Sept. 1980.
50. K. I. Mori, M. Kidode, and H. Asada, *An Iterative Prediction and Correction Method for Automatic Stereocomparison*, 2, pp. 393-401, CGIP, 1973.
51. N. O'Brien and R. Jain, "Axial Motion Stereo," in *IEEE Workshop On Computer Vision*, pp. 88-92, April 1984.
52. Y. Ohta and T. Kanade, "Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming," CMU-CS-83-162, Computer Science Dept., Carnegie-Mellon University, Oct. 1983.
53. M. H. Pirenne, *Optics Painting and Photography*, Cambridge Press University, 1970.
54. T. Poggio, "Vision by Man and Machine," *Scientific American*, vol. 250, no. 4, pp. 106-116, April 1984.
55. K. Prazdny, "Egomotion and relative depth map from optical flow," *Biological Cybernetics*, vol. 36, pp. 87-102, 1980.
56. C. S. Rees, S. M. Shak , and C. V. Stanojevic, *Theory and Applications of Fourier Analysis*, Marcel Dekker Inc., 1981.
57. D. Regan and K. I. Beverley, "Electrophysiological Evidence for Existence of Neurons Sensative to Direction of Depth Movement," *Nature*, vol. 246, pp. 504-506, London, 1973.
58. D. Regan and K. I. Beverley, "Looming Detectors in the Human Visual Pathway," *Vision Research*, vol. 18, pp. 515-421, 1978.
59. D. Regan, K. I. Beverley, and M. Cynader, "The Visual Perception of Motion in Depth," *Scientific American*, vol. 241, pp. 136-151, 1979.

60. D. Regan, K. I. Beverley, and M. Cynader, "Stereoscopic Subsystems for position in Depth and for Motion in Depth," *Phil. Trans. Royal Soc. of London*, vol. B204, pp. 481-501, 1979.
61. W. Richards, *Structure from Stereo and Motion*, MIT AI-Memo 731, September, 1983.
62. G. Sandini and V Tagliasco, "an Anthromorphic Retin-like Structure for Scene Analysis," *CVGIP*, vol. 14, pp. 365-372, 1980.
63. P. S. Schenker, K. M. Wong, and E. G. Cande, "Fast Adaptive Algorithms for Low-level Scene Analysis: Application of Polar Exponential Grid (PEG) Representation to High-speed, scale-and-rotation Invariant Target Segmentation," *Proc. of SPIE, Techniques and Applications of Image Understanding*, vol. 281, pp. 47-57, 1981.
64. W. Schiff, "Perception of Impending Collision: A Study of Visually Directed Avoidant Behavior," *Psychological Monographs: General and Applied*, vol. 79, no. 11, Whole No. 604, 1965.
65. E. L. Schwartz, "Spatial Mapping in the Primate Sensory Projection: Analytic Structure and Relevance to Perception," *Biological Cybernetics*, vol. 25, pp. 181-194, 1977.
66. E. L. Schwartz, "Computational Anatomy and Function Architecture of Striate Cortex: A Spatial Mapping Approach to Coding," *Vision Research*, vol. 20, pp. 645-669, 1980a.
67. E. L. Schwartz, "A Quantitative Model of the Functional Architecture of Human Striate Cortex with Application to Visual Illusion and Cortical Texture Analysis," *Biological Cybernetics*, vol. 37, pp. 63-76, 1980b.
68. E. L. Schwartz, "Cortical Anatomy, Size invariance and Spatial Frequency Analysis," *Perception*, vol. 10, pp. 455-468, 1981.



69. E. L. Schwartz, "Columnar Architecture and Computational Anatomy in Primate Visual Cortex: Segmentation and Feature Extraction Via Spatial Frequency Coded Difference Mapping," *Biological Cybernetics*, vol. 42, pp. 157-168, 1982.
70. J. L. Stansfield, *Conclusions from the commodity expert project*, MIT AI-Memo 601, 1980.
71. S. Ullman, *The Interpretation of Visual Motion*, MIT Press, Cambridge, Mass, 1979.
72. S. Ullman, "Analysis of Visual Motion by Biological and Computer Systems," *IEEE Trans. on Computer*, vol. 14, pp. 57-69, 1981.
73. C. Weiman and G. Chaikin, "Log Spiral Grids in Computer Pattern Recognition," *CVGIP*, vol. 10, pp. 197-226, 1979.
74. C. Wheatstone, "Contributions to the Physiology of Vision," *Phil. Trans. Royal Soc. of London*, vol. B 13, pp. 371-394, 1838.
75. T. D. Williams, "Depth from Camera Motion in a Real World Scene," *IEEE Trans. on PAMI*, vol. 2, p. 6, 1980.
76. A. P. Witkin, "Scale-Space Filtering," *IJCAI-83 Proc.*, pp. 1019-1022, Karlsruhe, West Germany, 1983.
77. A. L. Yuille and T. Poggio, *Scaling Theorems for Zero-Crossing*, MIT AI-Memo 722, 1983a.
78. A. L. Yuille and T. Poggio, *Fingerprints Theorems for Zero-Crossings*, MIT AI-Memo 730, 1983b.
79. A. L. Yuille, *Zero-Crossings on Lines of Curvature*, MIT AI-Memo 718, December, 1984.



# APPENDIX I

## Step Signals

A single step signal (fig. A1.1) is a function which is discontinuous at some  $x = x_0$ . It is given by

$$f(x) = \begin{cases} y_1 & x \geq x_0 \\ y_0 & x < x_0 \end{cases} \quad (\text{A1.1})$$

and by its integral property

$$\int_{-\infty}^{\infty} f(x-x_0) v(x) dx = y_1 \int_{x_0}^{\infty} v(x) dx \quad (\text{A1.2})$$

where  $v(x)$  is an arbitrary function.

This definition can be extended to multi step functions, say  $n$  steps, by viewing such functions as a stack of single steps (fig. A1.2) adjacent to each other. Its integral property is then defined as

$$\begin{aligned} \int_{-\infty}^{\infty} f(x) v(x-t) dt &= y_1 \int_{x_0}^{x_1} v(x-t) dt + y_2 \int_{x_1}^{x_2} v(x-t) dt + \\ & y_3 \int_{x_2}^{x_3} v(x-t) dt + \cdots + y_n \int_{x_{n-1}}^{\infty} v(x-t) dt \end{aligned} \quad (\text{A1.3})$$

If  $v(x)$  is taken to be the second derivative of an one dimensional Gaussian filter,  $g''(x)$ , as defined in appendix II, then the resulting convolution is

$$\begin{aligned} h(x, \sigma) &= y_1 \int_{x_0}^{x_1} g''(x-t) dt + y_2 \int_{x_1}^{x_2} g''(x-t) dt + \\ & y_3 \int_{x_2}^{x_3} g''(x-t) dt + \cdots + y_n \int_{x_{n-1}}^{\infty} g''(x-t) dt \end{aligned} \quad (\text{A1.4})$$

$$= y_1(-g'(x-x_1) + g'(x-x_0)) + y_2(-g'(x-x_2) + g'(x-x_1)) +$$

$$y_3(-g'(x-x_3) + g'(x-x_2)) + \cdots + y_n(-0 + g'(x-x_{n-1})) \quad (\text{A1.5})$$

hence

$$h(x, \sigma) = y_1 g'(x-x_0) + (y_2 - y_1) g'(x-x_1) + \\ (y_3 - y_2) g'(x-x_2) + \cdots + (y_n - y_{n-1}) g'(x-x_{n-1}) \quad (A1.6)$$

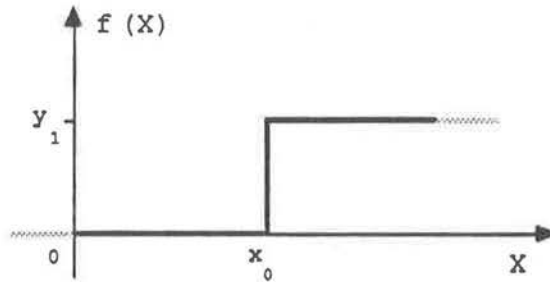


Figure A1.1: a single step signal

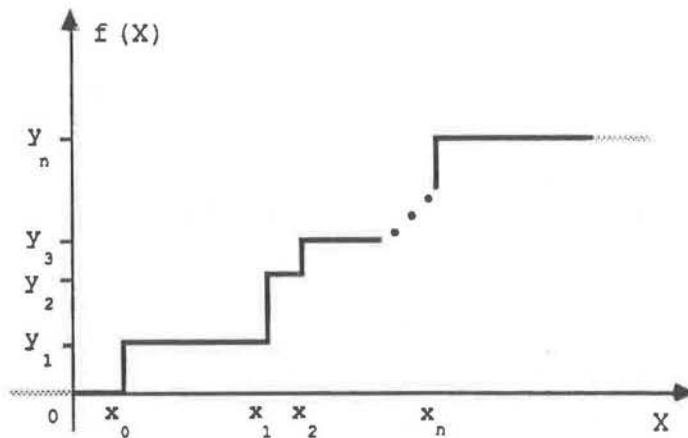


Figure A1.2: a multi step signal

## APPENDIX II

### The Gaussian Filter and its Derivatives

A one dimensional Gaussian is given by

$$g(x, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}} \quad (\text{vii})$$

its first derivative by

$$g'(x, \sigma) = \frac{-x}{\sigma^2} g(x, \sigma) \quad (\text{viii})$$

its second derivative by

$$g''(x, \sigma) = \frac{(x^2 - \sigma^2)}{\sigma^4} g(x, \sigma) \quad (\text{ix})$$

its third derivative by

$$g'''(x, \sigma) = \frac{-(x^3 - 3x\sigma^2)}{\sigma^6} g(x, \sigma) \quad (\text{x})$$

and its fourth derivative by

$$g''''(x, \sigma) = \frac{(x^4 - 6x^2\sigma^2 + 3\sigma^4)}{\sigma^8} g(x, \sigma) \quad (\text{xi})$$

Notice also the following relations

$$g\left(\frac{x}{k}, \sigma\right) = kg(x, k\sigma) \quad (\text{xii})$$

and

$$g(kx, k\sigma) = \frac{1}{k} g(x, \sigma) \quad (\text{xiii})$$



## APPENDIX III

### Proofs Concerning Open Contours

One dimensional signals can be grouped using some features that characterizes them. We prove for particular groups that the contain in their SSI, a particular number of open contours. Five groups are identified and addressed separately. We use the following notations for the features used to group the functions:

- a) the initial value of a function  $init = \lim_{x \rightarrow -\infty} f(x)$ .
- b) the final value of a function  $fin = \lim_{x \rightarrow +\infty} f(x)$ .
- c) the maximum value of a function  $fmax \geq f(x)$ .
- d) the minimum value of a function  $fmin \leq f(x)$ .

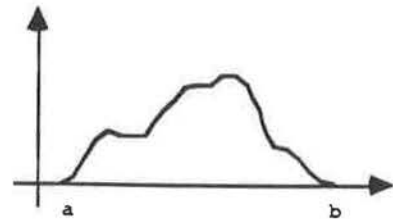
We assume, for convenience, the following:

- $fin \geq init = 0$ .
- $fmax > 0$ .

**Case A1:** Signals where

$$f(x) = \begin{cases} c & x < a \\ c & x > b \end{cases}$$

and  $fmin = fin = init$ , have exactly two open contours in their SSI.



The one dimensional Gaussian function and its derivatives are defined in appendix II. The convolution of a function  $f(x)$  with the Gaussian filter, is

expressed by

$$h(x, \sigma) = f(x) \otimes g(x, \sigma) \quad (\text{A3.1})$$

$$= \int_{-\infty}^{+\infty} f(t) g(x-t, \sigma) dt \quad (\text{A3.2})$$

Suppose that  $f(x) = 0$ , for  $x < a$  and  $x > b$ .

Hence, we can write

$$h(x, \sigma) = \int_a^b f(t) g(x-t, \sigma) dt \quad (\text{A3.3})$$

substituting  $s = x-t$  renders

$$h(x, \sigma) = \int_{x-b}^{x-a} f(x-s) g(s, \sigma) ds. \quad (\text{A3.4})$$

Let

$$c = \max(|a|, |b|) \quad (\text{A3.5})$$

hence

$$h(x, \sigma) = \int_{x-c}^{x+c} f(x-s) g(s, \sigma) ds. \quad (\text{A3.6})$$

We now show that for a large enough  $\sigma$  the 2<sup>nd</sup> derivative of the convolved function,  $h(x, \sigma)$ , has only two zero-crossings:

$$h''(x, \sigma) = \int_a^b f(t) g''(x-t, \sigma) dt \quad (\text{A3.7})$$

$$= \int_{x-b}^{x-a} f(x-s) g''(s, \sigma) ds \quad (\text{A3.8})$$

$$= \int_{x-c}^{x+c} f(x-s) g''(s, \sigma) ds \quad (\text{A3.9})$$

substituting  $u = \frac{s}{\sigma}$  renders

$$h''(x, \sigma) = \sigma \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) g''(\sigma u, \sigma) du \quad (\text{A3.10})$$



from (A2.3) and (A2.7) in appendix II we have

$$g''(\sigma u, \sigma) = \frac{(\sigma^2 u^2 - \sigma^2)}{\sigma^4} g(\sigma u, \sigma) \quad (\text{A3.11})$$

$$= \frac{u^2 - 1}{\sigma^3} g(u, 1) \quad (\text{A3.12})$$

hence

$$h''(x, \sigma) = \frac{1}{\sigma^2} \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u)(u^2-1) g(u, 1) du. \quad (\text{A3.13})$$

Since

$$(u^2 - 1)g(u, 1) = g''(u, 1) \quad (\text{A3.14})$$

and from Taylor's formula we have

$$\begin{aligned} g''(u, 1) &= g''\left(\frac{x}{\sigma}, 1\right) + \left(u - \frac{x}{\sigma}\right) g'''\left(\frac{x}{\sigma}, 1\right) + \left(u - \frac{x}{\sigma}\right)^2 g''''\left(\frac{x}{\sigma}, 1\right) + \dots \\ &\dots + \left(u - \frac{x}{\sigma}\right)^n g^{(n+2)}\left(\frac{x}{\sigma}, 1\right) \end{aligned} \quad (\text{A3.15})$$

we obtain

$$\begin{aligned} h''(x, \sigma) &= \frac{1}{\sigma^2} \left[ g''\left(\frac{x}{\sigma}, 1\right) \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) du + \right. \\ &\quad \left. + g'''\left(\frac{x}{\sigma}, 1\right) \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u)\left(u - \frac{x}{\sigma}\right) du + \dots \right] \quad (\text{A3.16}) \\ &= \frac{g\left(\frac{x}{\sigma}, 1\right)}{\sigma^2} \left[ \frac{(x^2 - \sigma^2)}{\sigma^2} \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) du - \right. \end{aligned}$$

$$\left. - \frac{(x^3 - 3x\sigma^2)}{\sigma^3} \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) \left(u - \frac{x}{\sigma}\right) du + \dots \right] \quad (\text{A3.17})$$

Since  $u = \frac{x-t}{\sigma}$  and  $t = x - \sigma u$ , we can write

$$\int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) du = \frac{1}{\sigma} \int_{-c}^c f(t) dt \quad (\text{A3.18})$$

and

$$\int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(x-\sigma u) \left(u - \frac{x}{\sigma}\right) du = \frac{1}{\sigma^2} \int_{-c}^c t f(t) dt \quad (\text{A3.19})$$

both of which are independent of  $x$ .

Since  $f(t) \geq 0$ , we can write

$$\left| \frac{1}{\sigma^2} \int_{-c}^c t f(t) dt \right| \leq \frac{c}{\sigma^2} \int_{-c}^c f(t) dt = \frac{c/\sigma}{\sigma} \int_{-c}^c f(t) dt \quad (\text{A3.20})$$

hence

$$h''(x, \sigma) = \frac{\sigma g(x, \sigma)}{\sigma^3} \left[ \frac{(x^2 - \sigma^2)}{\sigma^2} - \frac{(x^3 - 3x\sigma^2)}{\sigma^3} O(\epsilon) \right] \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(t) dt \quad (\text{A3.21})$$

$$\approx \frac{(x^2 - \sigma^2)}{\sigma^4} g(x, \sigma) \int_{\frac{x-c}{\sigma}}^{\frac{x+c}{\sigma}} f(t) dt \quad (\text{A3.22})$$

Hence, setting

$$h''(x, \sigma) = 0 \quad (\text{A3.23})$$

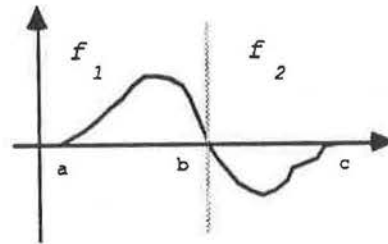
renders zero-crossings at

$$x \approx \pm \sigma. \quad (\text{A3.24})$$

Hence, given a function which is zero outside some finite interval and whose activity within that interval is either all positive or all negative, then convolving it with a Gaussian function with a large enough  $\sigma$ , will result in a smoothed Gaussian-like function. For such a  $\sigma$ , the 2<sup>nd</sup> derivative of this function has two zero-crossings (i.e. the function has two fluctuation points). Note that this entails one zero-crossing in its 1<sup>st</sup> derivative (i.e. the function has one extremum).

**Case A2: Signals where**

$$f(x) = \begin{cases} \text{init} & x < a \\ \text{fin} & x > c \\ 0 & x = b, \quad a < b < c \\ \text{non-0} & \text{elsewhere} \end{cases}$$



and  $f(x)$  at  $a \leq x < b$  is opposite in sign to  $f(x)$  at  $b < x \leq c$ , and obey  $\text{fin} = \text{init}$ , and  $\text{fmin} < 0$ , have exactly three open contours in their SSI.

The function  $f(x)$  can be broken into two:

$$f(x) = f_1(x) + f_2(x). \tag{A3.25}$$

Hence, the convolved function can be expressed by

$$h(x, \sigma) = \int_{-\infty}^{+\infty} [f_1(t) + f_2(t)] g(x-t, \sigma) dt \tag{A3.26}$$

$$= h_1(x, \sigma) + h_2(x, \sigma) \tag{A3.27}$$

From case (A1) above, one can conclude that both  $h_1(x, \sigma)$  and  $h_2(x, \sigma)$ , at a large enough  $\sigma$ , are Gaussian-like functions of opposite signs. Since the single

extremum of  $h_1(x, \sigma)$  is located left of  $b$ , and the that of  $h_2(x, \sigma)$  is right of  $b$ ,  $h(x, \sigma)$  must have a third zero-crossings between these two extrema (fig. A3.1). As  $\sigma$  increases in value, this third zero-crossing (in  $h''(x, \sigma)$ ) moves towards an imaginary rectilinear line midway between  $a$  and  $c$ .

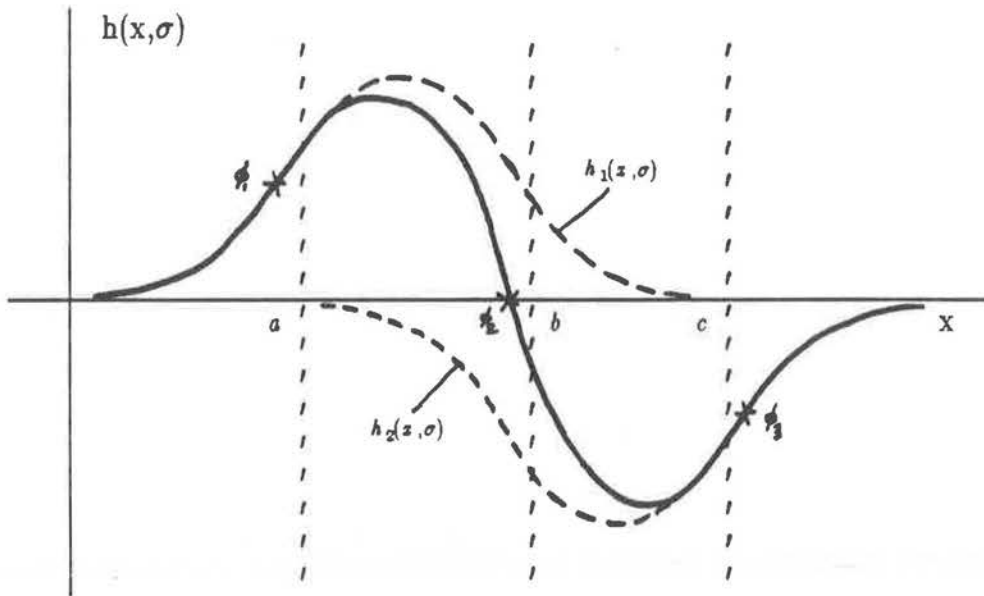
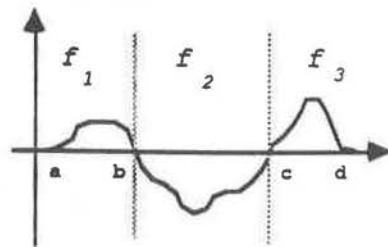


Figure A3.1: The sum of the two smoothed functions  $h_1$  and  $h_2$  renders a function  $h$  which contains exactly three fluctuation points ( $\phi$ ), for a large enough  $\sigma$ .

**Case A3: Signals where**

$$f(x) = \begin{cases} \text{init} & x < a \\ \text{fn} & x > d \\ 0 & x = b, \quad a < b < c < d \\ 0 & x = c, \quad a < b < c < d \\ \text{non-0} & \text{elsewhere} \end{cases}$$



and  $f(x)$  at  $a \leq x < b$  and  $c < x \leq d$ , is opposite in sign to  $f(x)$  at  $b < x < c$ , and obey  $f_{in} = \text{init}$ , and  $f_{min} < 0$ , have two or four open contours in their SSI.

We follow the same approach as in case 3 above:

$$f(x) = f_1(x) + f_2(x) + f_3(x) \quad (\text{A3.28})$$

hence, the convolved function can be expressed by

$$h(x, \sigma) = \int_{-\infty}^{+\infty} [f_1(t) + f_2(t) + f_3(t)] g(x-t, \sigma) dt \quad (\text{A3.29})$$

$$= h_1(x, \sigma) + h_2(x, \sigma) + h_3(x, \sigma) \quad (\text{A3.30})$$

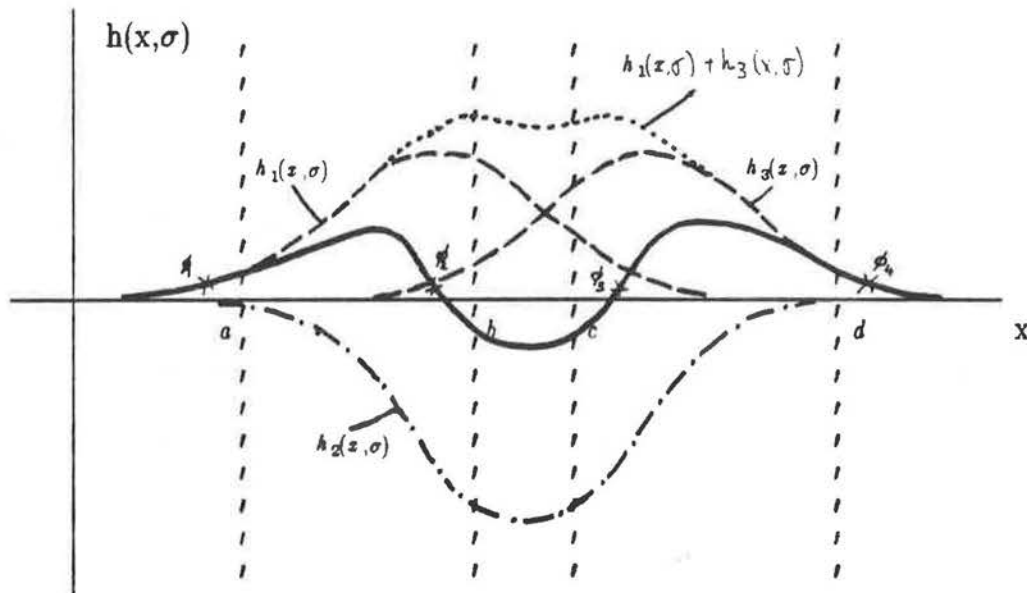


Figure A3.2: The sum of the three smoothed functions  $h_1$ ,  $h_2$  and  $h_3$  renders a function  $h$  which, in this example, contains four fluctuation points ( $\phi$ ), for a large enough  $\sigma$ .

Equations (A3.21) and (A3.22) suggest that the outcome of the convolution operation depends on the area enclosed under the function's graph.

Let

$$c_1 = \max(|a|, |b|) \quad (\text{A3.31})$$

$$c_2 = \max(|b|, |c|) \quad (\text{A3.32})$$

$$c_3 = \max(|c|, |d|) \quad (\text{A3.33})$$

Since at any given time the functions  $f_1$ ,  $f_2$ , and  $f_3$  are convolved with an identical mask, we can expand equation A3.22 to read

$$h''(x, \sigma) \simeq \frac{(x^2 - \sigma^2)}{\sigma^4} g(x, \sigma) \left[ \int_{\frac{x-c_1}{\sigma}}^{\frac{x+c_1}{\sigma}} f_1(t) dt + \int_{\frac{x-c_2}{\sigma}}^{\frac{x+c_2}{\sigma}} f_2(t) dt + \int_{\frac{x-c_3}{\sigma}}^{\frac{x+c_3}{\sigma}} f_3(t) dt \right] \quad (\text{A3.34})$$

$f_2(t) \leq 0$ . Therefore, if

$$\left| \int_{\frac{x-c_2}{\sigma}}^{\frac{x+c_2}{\sigma}} f_2(t) dt \right| > \int_{\frac{x-c_1}{\sigma}}^{\frac{x+c_1}{\sigma}} f_1(t) dt + \int_{\frac{x-c_3}{\sigma}}^{\frac{x+c_3}{\sigma}} f_3(t) dt \quad (\text{A3.35})$$

then  $h''(x, \sigma)$  will be negative, implying that its minimum is negative. This minimum must be between  $b$  and  $c$ . Hence,  $h(x, \sigma)$  will have three extrema points: two maxima and a minimum between them, and the SSI will contain four open contours.

If

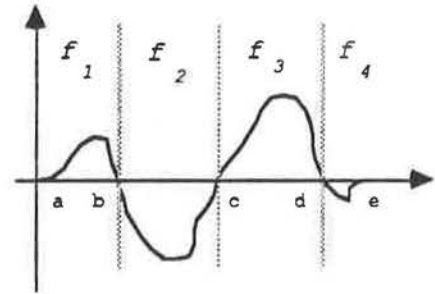
$$\left| \int_{\frac{x-c_2}{\sigma}}^{\frac{x+c_2}{\sigma}} f_2(t) dt \right| \geq \int_{\frac{x-c_1}{\sigma}}^{\frac{x+c_1}{\sigma}} f_1(t) dt + \int_{\frac{x-c_3}{\sigma}}^{\frac{x+c_3}{\sigma}} f_3(t) dt \quad (\text{A3.36})$$

then  $h''(x, \sigma)$  will be positive, implying that its minimum is positive. Hence, a situation conforming case (A1) above is at hand. Thus, the function will have two open contours in its SSI.

The example in figures *e* illustrates the above results. In figure A3.3a we have  $A_1 + A_2 > A_3$  where  $A_1, A_2$  and  $A_3$  are the areas enclosed by the function's curve. The SSI in figure A3.3b is obtained, where two open contours are generated. In figure A3.3c,  $A_1 + A_2 \leq A_3$ . The SSI in figure A3.3d is obtained, where four open contours are generated.

**Case A4: Signals where**

$$f(x) = \begin{cases} \text{init} & x < a \\ \text{fn} & x > e \\ 0 & x = b, \quad a < b < c < d < e \\ 0 & x = c, \quad a < b < c < d < e \\ 0 & x = d, \quad a < b < c < d < e \\ \text{non-0} & \text{elsewhere} \end{cases}$$



and  $f(x)$  at  $a \leq x < b$  and  $c > x > d$  is opposite in sign to  $f(x)$  at  $b < x < c$  and  $d < x \leq e$ , and obey  $\text{fn} = \text{init}$ , and  $\text{fmin} < 0$ , have three or five open contours in their SSI.

Following the same approach we have:

$$f(x) = f_1(x) + f_2(x) + f_3(x) + f_4(x) \quad (\text{A3.37})$$

and

$$h(x, \sigma) = h_1(x, \sigma) + h_2(x, \sigma) + h_3(x, \sigma) + h_4(x, \sigma) \quad (\text{A3.38})$$

Figure A3.4 illustrates the formation of  $h(x, \sigma)$ .

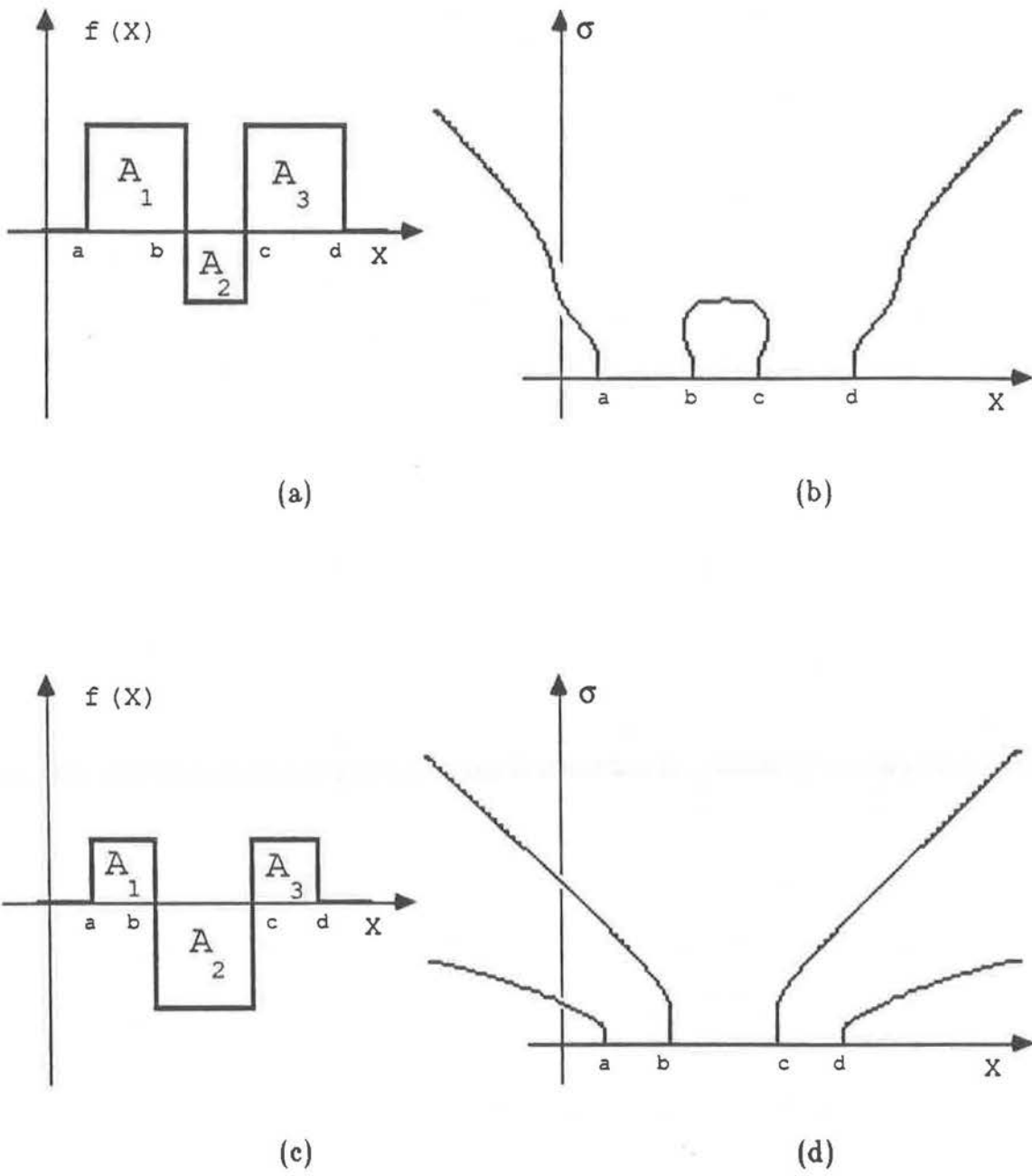


Figure A3.3: A signal that crosses the zero twice, generates two (a,b) or four (c,d) open contours in its SSI.



As is for segment (b,c) in case (A3) above, the area under the function curve for segments (b,c) and (c,d) has a crucial influence on the number of open contours in the SSI of the function. If either

$$|h_2(x,\sigma)| < |h_1(x,\sigma) + h_3(x,\sigma) + h_4(x,\sigma)| \quad (\text{A3.39})$$

or

$$|h_3(x,\sigma)| < |h_1(x,\sigma) + h_2(x,\sigma) + h_4(x,\sigma)| \quad (\text{A3.40})$$

for any  $\sigma$ , we will obtain a function that comply with case (A2) above and, hence, three open contours will be formed. Otherwise, five open contours will be formed. The example in figure A3.5 illustrates this fact.

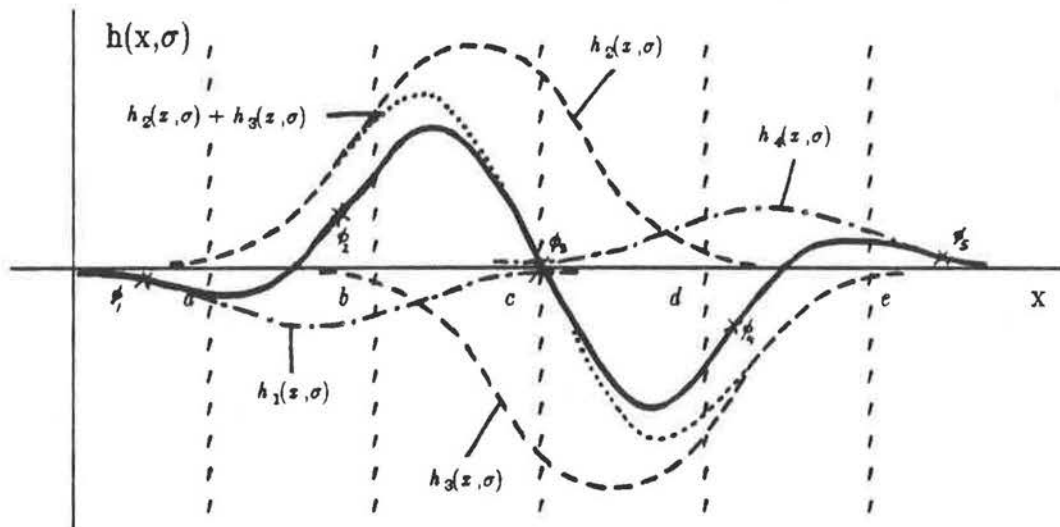
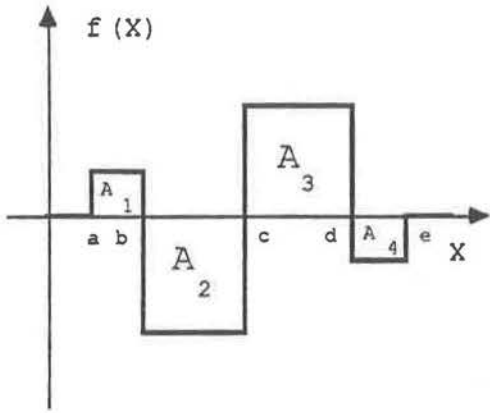
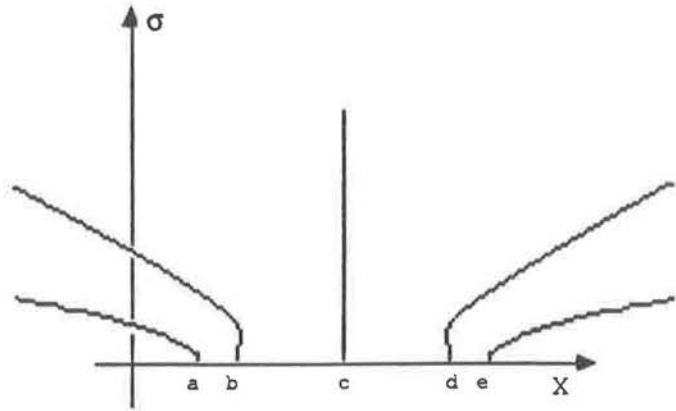


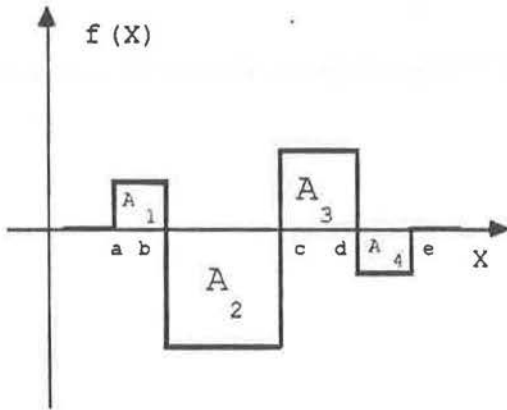
Figure A3.4: The sum of the two smoothed functions  $h_2$ ,  $h_3$  and  $h_4$  renders a function  $h$  which, in this example, contains five fluctuation points ( $\phi$ ), for a large enough  $\sigma$ .



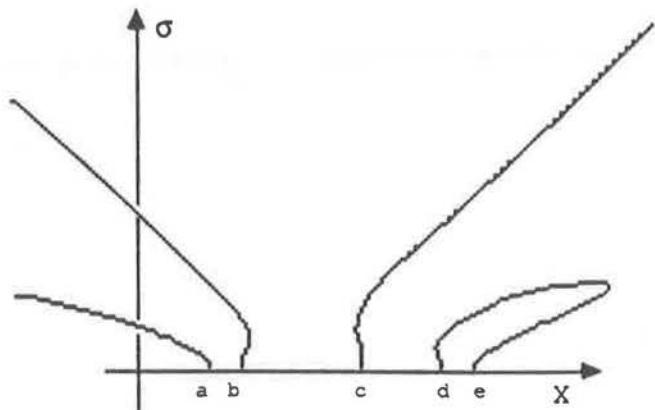
(a)



(b)



(c)

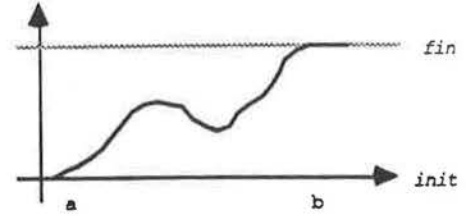


(d)

**Figure A3.5:** A signal that crosses the zero three times, generates three (c,d) or five (a,b) open contours in its SSI.

Case A5: Signals where

$$f(x) = \begin{cases} \text{init} & x < a \\ \text{fin} & x > b \end{cases}$$



and  $f_{\max} = \text{fin} > f_{\min} = \text{init} = 0$ , have exactly one open contour in their SSI.

If for all points in the function's domain its curve has a slope  $s \geq 0$ , then taking its 1<sup>st</sup> derivative produces a function which complies with functions in case (A1) above. Since the 1<sup>st</sup> derivative of functions obeying the constraints of case (A1), convolved with a Gaussian filter, have exactly one zero-crossing, it is also the case for the 2<sup>nd</sup> derivative of functions in this case with  $s \geq 0$  and convolved with a Gaussian filter.

If  $s$  at any point on the curve is negative, then taking its 1<sup>st</sup> derivative produces a function which complies with functions in case (A3) above. Since the function always approaches points  $a$  and  $b$  with a positive slope, a similarity to case (A4) above is ruled out.

Since  $f_{\text{in}} > \text{init}$ , the total length of the function's segments that have positive or zero slope is larger than that of segments with negative slope. Thus, for  $f'(x)$ , the total area under the function's curve is larger for segments with positive or zero slope than that for segments with negative slope. In case (A3) above we proved that such functions have exactly two open contours in their SSI. In other words, for a large enough  $\sigma_0$ , functions in this case will be smoothed to have only positive or zero slope implying exactly one zero-crossing. Hence, for all  $\sigma \geq \sigma_0$  one zero-crossing is obtained compounding to a single open contour in the SSI.



## APPENDIX IV

### Synthetic Image Construction

The *thin lens equation* is used to compute the locations of scene points projected onto an image plane. A lens with a focal length of 50mm is assumed in all calculations. The FOE is always taken to be at the center of the image.

The thin lens equation is given by

$$\frac{1}{f} = \frac{1}{d} + \frac{1}{x} \quad (\text{A4.1})$$

where  $f$  is the focal length,  $d$  is the distance of the object from the lens and  $x$  is the distance of image plane from the lens (see fig. A4.1).

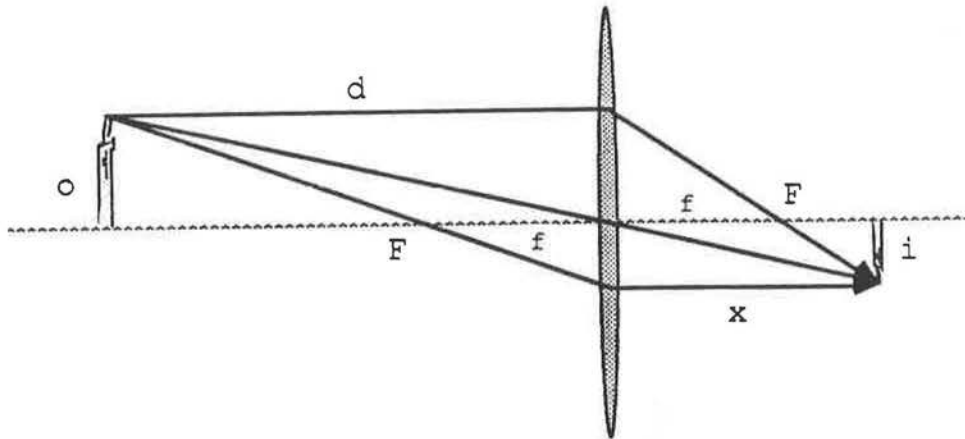


Figure A4.1: standard thin lens geometry

From triangulation we have

$$x = \frac{i}{o} d. \quad (\text{A4.2})$$

Substituting (A4.2) into (A4.1) renders

$$\frac{1}{f} = \frac{i+d}{d i} \quad (\text{A4.3})$$

and

$$i = \frac{o}{\frac{d}{f} - 1} \quad (\text{A4.4})$$

After  $i$  is obtained,  $x$  can be computed using (A4.2).

For the farther image, having  $o$  and  $x$  and knowing the change in  $d$ , the new  $i$  is computed by

$$i = \frac{x}{d} o \quad (\text{A4.5})$$

as in (A4.2).

## APPENDIX V

### Manual pages of a few programs in the public library

This appendix contains the manual pages for programs *binary-ss*, *clm*, *polar*, *scale-space*, *-ss-fill-gaps*, *ss-rep*. These manual pages are available on-line on the LCV's VAX computer.

The programs were developed as part of the work towards an M.Sc. degree and are part of the public library of programs on the above system.

**NAME**

binary-ss - constructs a binary (two-tone) scale-space image from a signed scale-space image.

**SYNOPSIS**

**ss-rep** [input scale-space image] [output scale-space image]

**DESCRIPTION**

*Ss-rep* produces a binary (two-tone) scale-space image from a signed scale-space image as produced by program *scale-space* (see *scale-space* (1-UBC)). All pixels preceding a negative ZC (inclusive) are set to 1, and those preceding a positive ZC (exclusive) are set to 0.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

The scale-space image produced by *scale-space* is up-side-down, and should be flipped before using it as input (see *yflip* (1-UBC)).

The input must be a signed scale-space image.

**SEE ALSO**

*scale-space*(1-UBC), *yflip*(1-UBC)



**NAME**

*clm* - transforms an image from cartesian space to complex logarithmic mapping (CLM) space.

**SYNOPSIS**

*clm* [-a] [input-image] [output image]

**DESCRIPTION**

*clm* transforms an image from cartesian space to complex logarithmic space, which is a polar space with logarithmic spacing along radial lines. The center of the cartesian image is taken to be the origin of the CLM image coordinates. Each row in the CLM image represents a radial line in the cartesian image.

The following options are interpreted by *clm* :

-*a* *alpha*

*alpha* is the angle interval between two consecutive radial lines. The default value is 1.

-*cubic* *cubic* is a flag indicating that cubic convolution should be used for the resampling process. Else, a 4x4 window is used for an extended bi-linear interpolation. The later is the default.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

$360/\alpha$  should be an integer.

The input image should not exceed 512 x 512.

**SEE ALSO**

*polar*(1-UBC), *image*(1-UBC)

**NAME**

**polar** - transforms an image from cartesian space to polar space.

**SYNOPSIS**

**polar** [-a -j] [input-image] [output image]

**DESCRIPTION**

*polar* transforms an image from cartesian space to polar space. The center of the cartesian image is taken to be the origin of the polar image coordinates. Each row in the polar image represents a radial line in the cartesian image.

The following options are interpreted by *polar* :

**-a *alpha***

*alpha* is the angle interval between two consecutive radial lines. The default value is 1.

**-cubic** *cubic* is a flag indicating that cubic convolution should be used for the resampling process. Else, a 4x4 window is used for an extended bi-linear interpolation. The later is the default.

**-j *adj*** if *adj* is none-0, an adjustment is made to the intensity value of pixels when the width of the sector grows to be larger than two cartesian pixels. The adjustment process uses a Gaussian Distribution function. The default value is 0.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

$360/\alpha$  should be an integer.

The input image should not exceed 512 x 512.

**SEE ALSO**

clm(1-UBC), image(1-UBC)

## NAME

scale-space - compute a scale-space image of a one dimensional function.

## SYNOPSIS

**scale-space** [options] [input-image] [signed+magnitude scale-space image] [binary scale-space image]

## DESCRIPTION

*Scale-space* generates two scale-space images (SSIs) in standard image format. The first one is signed and the magnitude of the zero-crossings (slope of the function) indicated. The second one is binary - zero-crossings are marked as 1 and the rest of the image is 0.

The input can be either a 1D image or a 2D polar image (see *polar* (UBC-1), *clm* (UBC-1)).

In the convolution process, values for post boundary pixels must be assumed. There are three possible assumptions for pixels located beyond the image frame (**-bc** and **-be** options):

- 0 - use zeroes as the pixel's value.
- 1 - encore the last pixel.
- 2 - wrap around the image frame.

If the input image is polar, then the left edge of the image frame corresponds to the center of the original cartesian image (FOE). In such a case the intensities of pixels to the left of this edge are known (until the boundary of the cartesian image is reached). This allows us to optionally use these known intensities for the left edge of the polar image (**-bc** option) and then, as before:

- 3 - use zeroes as the pixel's value.
- 4 - encore the last pixel.
- 5 - wrap around the image frame.

The default value for pixels past the right end-point (**-be**) is 1, and for the left end-point (**-bc**) is 4.

The following additional options are interpreted by *scale-space* :

**-sigma**

*sigma* is the initial scale value, a parameter of the Gaussian function. It must be  $\geq \sqrt{2}$ . The default value is 1.0.

**-rrow** *row* is a row in a polar image for which the SSI will be computed. The default value is 0.

**-iint** *int* is the interval in value between two consecutive sigmas. The default value is 1.0.

**-lmaxn1**

if only one zero-crossing is generated at some large sigma value, terminate the process after *n1* such zero-crossings are produced. The default value is 20.

**-dlmaxn2**

if only two zero-crossings are generated at some large sigma value, terminate the process after *n1* such pairs of zero-crossings are produced. The default value is 120.

**-xlncol1**

extend the SSI to the left by *ncol1* columns. The default value is 0.

**-xrncol2**

extend the SSI to the right by *ncol2* columns. The default value is 0.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

The scale-space image produced is up-side-down, and should be flipped before using it (see *yflip* (1-UBC)).

Sigma should not be less than  $\sqrt{2}$ .

The input image should not exceed 360 x 724 (corresponds to a cartesian image of 512 x 512).

The scale-space image can grow to be very large and time consuming (cpu) to compute. This can be controlled by the size of the sigma interval, *-l*, and the truncation values of open contours, *-lmax* and *-dlmax*.

**SEE ALSO**

*polar*(1-UBC), *clm*(1-UBC), *ss-fill-gaps*(1-UBC), *ss-rep*(1-UBC), *binary-ss*(1-UBC), *image*(1-UBC), *yflip*(1-UBC)

**BUGS**

The wrap around option for *-l* has not been thoroughly tested and should be used with caution.

There can be gaps at the top of contours in the scale-space image due to quantization errors. Program *ss-fill-gaps* tries to 'fill' these gaps (see *ss-fill-gaps* (1-UBC)).

**NAME**

**ss-fill-gaps** - attempts to 'fill' gaps in contours of a scale-space image.

**SYNOPSIS**

**ss-fill-gaps** [input scale-space image] [output scale-space image]

**DESCRIPTION**

There can be gaps at the top of contours in the scale-space image due to quantization errors. *Ss-fill-gaps* tries to 'fill' these gaps: for each 'loose-end' found another 'loose-end' is searched for in its nearest neighborhood, which grows gradually to a certain limit.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

The scale-space image produced by *scale-space* is up-side-down, and should be flipped before using it as input (see *yflip* (1-UBC)).

If the gap is too big it may fail to close it.

**SEE ALSO**

*scale-space*(1-UBC), *ss-rep*(1-UBC), *image*(1-UBC), *yflip*(1-UBC)

**BUGS**

There may be situations where legitimate openings are wrongly filled.

**NAME**

`ss-rep` - constructs a LISP representation of a scale-space image.

**SYNOPSIS**

`ss-rep [-i] [input scale-space image] [output file]`

**DESCRIPTION**

`Ss-rep` reads in a binary scale-space image, as constructed by programs `scale-space` and `ss-fill-gaps` (see `scale-space` (1-UBC), `ss-fill-gaps` (1-UBC)), and creates a LISP list which contains information found in that image.

The option `-i` can be used for the input image, else it is read from standard input.

**AUTHOR**

Itzhak Katz

**LIMITATIONS**

contours in the scale-space image should not contain gaps.

A contour should not exceed 500 pixels.

The maximum input image is 1000 x 725.

This list is designed for a LISP program (`ss-match`) and addresses the needs of the *coaxial stereo* matching process. Hence, it may lack information required by other applications.

**SEE ALSO**

`scale-space`(1-UBC), `ss-fill-gaps`(1-UBC)