Purdue University
Purdue e-Pubs

Department of Electrical and Computer Engineering Technical Reports

Department of Electrical and Computer Engineering

7-1-1985

Sequential Detection of Linear Features in Two-Dimensional Random Fields

Paul H. Eichel
Purdue University

Edward J. Delp
Purdue University

Follow this and additional works at: https://docs.lib.purdue.edu/ecetr

Eichel, Paul H. and Delp, Edward J., "Sequential Detection of Linear Features in Two-Dimensional Random Fields" (1985). *Department of Electrical and Computer Engineering Technical Reports.* Paper 548.
https://docs.lib.purdue.edu/ecetr/548

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

Sequential Detection of Linear Features in Two-Dimensional Random Fields

Paul H. Eichel
Edward J. Delp

TR-EE 85-13
July 1985

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

SEQUENTIAL DETECTION OF LINEAR FEATURES IN TWO-DIMENSIONAL RANDOM FIELDS

Paul H. Eichel

Edward J. Delp

Purdue University
School of Electrical Engineering
West Lafayette, IN 47907

TR-EE 85-13

July 1985

ACKNOWLEDGMENTS

This technical report describes the research work of Paul H. Eichel that was submitted as partial fulfillment of the requirements for the Ph.D. degree at the University of Michigan under the direction of Professor Edward J. Delp (currently associated with Purdue University). Dr. Eichel is currently a member of the technical staff of Sandia National Laboratories, Albuquerque, New Mexico.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

EDGE DETECTION IN IMAGES

1.1. Introduction

The detection of edges, lines, and other linear features in two-dimensional discrete images is a low level processing step of fundamental importance in the automatic processing of such data. Many subsequent tasks in computer vision, pattern recognition, and image processing depend on the successful execution of this step.

In this thesis, we will address one class of techniques for performing this task: sequential detection. Our aims are fourfold. First, we would like to discuss the use of sequential techniques as an attractive alternative to the somewhat better known methods of approaching this problem. Although several researchers have obtained significant results with sequential type algorithms, the inherent benefits of a sequential approach would appear to have gone largely unappreciated. Secondly, the sequential techniques reported to date appear somewhat lacking with respect to a theoretical foundation. Furthermore, the theory that is advanced incorporates rather severe restrictions on the types of images to which it applies, thus imposing a significant limitation to the generality of the method(s). We seek to advance a more general theory with minimal assumptions regarding the input image. A third goal is to utilize this newly developed theory to obtain *quantitative* assessments of the performance of the method. This important step, which depends on a computational theory, can answer such vital questions as: Are assumptions about the qualitative behavior of the method justified? How does signal-to-noise ratio impact its behavior? How fast is it? How accurate? The state of theoretical development of present techniques does not allow for this type of analysis. Finally, a fourth aim is to extend the earlier results to include correlated image data. Present sequential methods as well as many non-sequential methods assume that the image data is uncorrelated and cannot therefore make use of the mutual information between pixels in real-world images. We would like to extend the theory to incorporate correlated images and demonstrate the advantages incurred by the use of the existing mutual information.

The topics to be discussed are organized in the following manner. We will first provide a rather general discussion of the problem of detecting intensity edges in images. The edge detection problem will serve as the prototypical problem of linear feature extraction for much of this thesis. It will later be shown that the detection of lines, ramp edges, texture edges, etc. can be handled in similar fashion to intensity edges, the only difference being the nature of the preprocessing operator used. The class of sequential techniques will then be introduced, with a view to emphasize the particular advantages and disadvantages exhibited by the class. This Chapter will conclude with a more detailed treatment of the various sequential algorithms proposed in the literature. Chapter 2 then develops the algorithm proposed by the author, Sequential Edge Linking or SEL. It begins with some definitions, follows with a derivation of the critical path branch metric and some of its properties, and concludes with a discussion of algorithms. The third Chapter is devoted exclusively to an analysis of the dynamical behavior and performance of the method. Chapter 4 then deals with the case of correlated random fields. In that Chapter, a model is proposed for which paths searched by the SEL algorithm are shown to posses a well-known autocorrelation function. This allows the use of a simple linear filter to decorrelate the raw image data. Finally, Chapter 5 presents a number of experimental results and corroboration of the theoretical conclusions of earlier Chapters. Some concluding remarks are also included in Chapter 5.

1.2. Edge Detection

Edge detection represents one of the first processing steps in a great many computer vision and image processing tasks. Reflecting this importance, the literature devoted to the problem is enormous. One need only consult a recent bibliography such as Rosenfeld [Ros84] to gain an appreciation of this fact. For this reason, we will make no attempt to summarize the work in this area to date. Instead, we will confine ourselves to a general discussion of the edge detection problem and motivate the method to which this thesis is addressed, namely that of sequential detection.

We will consider only two-dimensional digital images. By *digital* we mean that the image is discrete in the spatial domain, e.g. the image intensity function is not continuous over the two dimensions but defined only on an array of points, and the intensity levels at these points are furthermore *quantized* into a fixed, finite number of levels.

The underlying assumption in this and many other treatments of edge detection is that *edges* of interest in real scenes such as object boundaries, etc. are represented in an image as a discontinuity in intensity. Therefore the task of *edge detection* becomes one of identifying intensity discontinuities. We note that the human visual systems and perception are such that discontinuities are not the only intensity functions that are perceived as "edges." Other possibilities include discontinuities in the first derivative of the intensity function (ramp edges), texture edges, and color changes [Cor70]. While these features are important in some contexts and are in fact sufficiently similar to the original problem of intensity discontinuities to perform their detection by sequential techniques, in this Section we will confine ourselves solely to the first problem. Bear in mind, however, that sequential algorithms attempt to exploit only the *connectivity* of edges and are largely independent of the specific edge operator used. We will return to this point later.

Given that the goal is to identify intensity discontinuities, two general classes of techniques have emerged to address this problem: gradient-type operators and parametric models. Gradient-type operators, which for the purposes of this discussion will include first and second order spatial derivative operators, are discrete spatial filters whose magnitude responses have various high pass characteristics. In particular, they attempt to perform the discrete equivalent of a two-dimensional gradient, a directional derivative operator, or a second order derivative operator. The idea is to emphasize those regions of an image where the intensity function changes rapidly with distance and suppress the areas with little change in intensity. These operators may also provide information regarding the *direction* of the gradient, or in the case of directional operators, the component of the gradient in a given direction.

Generally speaking, gradient-type operators are implemented with one of a variety of window functions [Ham83]. This is due to the fact that real edges are of finite extent and therefore the operator must have finite support, i.e. be windowed. If the window function is rectangular, the spectral response of the operator will exhibit the familiar Gibbs phenomenon of Fourier theory. The large gain at high spatial frequencies exacerbates the effects of noise on the output. As in other signal processing applications, the answer to this problem is to employ smoother window functions such as Hamming, Hanning, or Gaussian windows [Can83].

Examples of gradient-type edge operators are the Roberts [Rob65], the Sobel [Pra78], the Laplacian-based operators of Modestino and Fries [Mod77] and Marr and Hildreth [Mar80], and correlation template techniques such as Kirsch [Kir71].

Parametric models view the image intensity function as a surface and this surface is projected onto a set of basis functions. From this *modeled* surface, edge parameters such as slope position and direction are estimated. A question of importance is the completeness of the basis function set, as the parameters can be estimated only from the projection of the actual image onto the space spanned by that set. Examples of this approach are the Hueckel [Hue73] and facet model of Haralick [Har84]. For the purposes of this discussion we will include in this class the moment-type operators such as the pixel mass operator of Suciu and Reeves [Suc82]. These methods attempt to detect intensity discontinuities from the moments of the distribution of intensity values in a window.

As pointed out by Canny [Can83], practically all edge detection schemes proposed to date in the above classes involve a classification step that utilizes one or more thresholds. Having obtained estimates of the gradient magnitude of direction or edge parameters from a fitted model, some mechanism must be employed to decide whether or not those quantities indicate the presence of an intensity edge at the location in question. This classification step is performed via a decision threshold. Even the second order derivative approachs, such as Marr and Hildreth [Mar80], cannot avoid this step. Although zero crossings of the two-dimensional Laplacian nominally indicate intensity edges, even small amounts of noise contribute to a very high density of noise induced zero crossing contours. Therefore, practical implementations must apply a threshold to the slope of the second derivative perpendicular to the zero crossing.

This thresholding process may be accomplished in various ways. Earlier techniques established a global threshold on the basis of the histogram of operator outputs or the Receiver Operating Characteristic (ROC) of the operator [Abd79]. More recent methods select the threshold in an adaptive manner based on local image content or on entropy ideas.

The question of threshold selection raises two fundamental and related problems with all of these edge detection techniques. The first of these is known as streaking. This phenomenon results from the fact that real images are highly non-homogeneous and edge parameters may change substantially even along the same edge contour. Regardless of the sophistication of the threshold selection process, it is possible and in fact common that due to noise, the operator output is at times above and other times below the decision threshold along the length of a given edge contour. This results in an edge map in which edges that are in reality a single connected contour are only partially detected. The broken segments or *streaks* are a major concern since many processing tasks that follow edge detection require well connected or even closed edge contours. On the other hand, if the thresholds are set so liberally that the edges of interest are detected with good connectivity, then many *false detections* and *multiple responses* on strong edges occur. This is the classical detection theory trade-off between probability of detection and probability of false alarm. Not only is it difficult to decide on a threshold, it is fundamentally impossible to simultaneously achieve a high detection probability and low false alarm rate as the signal-to-noise ratio decreases.

A second and related problem is the performance at low signal-to-noise ratio. Since operators attempt to make a decision based only on local information, as the noise power increases, this decision becomes increasingly more difficult to make. The solution generally adopted is to increase the number of observations that contribute to the decision, i.e. make the operator larger. As Canny shows [Can83], this improves the output signal-to-noise ratio of the operator but only at the expense of spatial resolution. The way to circumvent *that* problem is to employ a set of directional operators: long skinny operators that pick up their additional observations along an edge rather than out in every direction [Nav80]. This may only be taken so far, however, because the more directional the operator, the larger the requisite set of such operators. Additional complications arise from the fact that edges in real images tend to not run straight for very long. This mandates the inclusion of curved operators which further compounds the job of choosing an operator set.

1.3. Sequential Edge Detection

Classical detection theory states that the way to improve performance at low signal-to-noise ratio is to increase the number of observations contributing to the decision. As we have just seen, simply increasing the size of the edge operator is moderately successful in this regard but at the expense of spatial resolution. In addition, the desirable "edge information" generally lies in the vicinity of the edge itself, so picking up observations far from the edge contributes little to the decision process. Using directional operators improves the output signal-to-noise ratio while maintaining good spatial resolution, but this approach soon becomes unwieldy as the number of such operators increases geometrically with their length.

One possible way out of this dilemma is to assemble observations *along* the edge contour. Observations are made in a very long and narrow *path* that is deformed to lie along the edge, including curves, corners, and straight segments. Since the set of *all possible* such paths is enormous, the paths are instead "grown" in an iterative fashion beginning at a point that is known to lie on the edge. This is the basic philosophy behind *sequential edge detection*. A *searching algorithm* attempts to hypothesize possible edge topologies or paths. These paths are extended iteratively, with the current *most probable* path extended by one observation at each iteration.

For this technique to succeed in finding edges, a means of comparing all paths hypothesized so far has to be provided. This comparison is accomplished by associating with each path a statistic called a *path metric* which reflects the given path's probability of coinciding with the edge contour. Therefore, only the most likely paths are extended by the searching algorithm. In this way an exhaustive search is avoided.

Sequential edge detection has several potential advantages over the techniques discussed earlier. First, it offers the possibility of better performance (higher detection probability, lower false alarm probability) at low image signal-to-noise ratio than the local operators, since it obtains many more observations along the edge. For the same reason, the problem of choosing a detection threshold is alleviated: it is much easier to decide "edge" or "no edge" based on many observations than on a few. Secondly, by the very nature of the searching process, the detected edge paths exhibit complete connectivity. Therefore, streaking can be eliminated. Although it is not obvious from this discussion, two subtle advantages to a sequential approach also arise. One is that it allows an analytical treatment of the probability that *segments* of detected edge contours are in error rather than merely points. The second is that it provides a framework in which the correlation between observations in an image can be exploited to aid in the detection process. The principle disadvantage of a sequential approach is one of computational speed. Actually, on a sequential processor, such algorithms tend to be more efficient than parallel algorithms. But the latter has the potential of dramatically improved speed on special purpose parallel architectures.

We will return to discuss all of the foregoing ideas in more detail later, but first we shall review preceding efforts in the literature related to the detection of edges in images by sequential methods.

1.4. Previous Sequential Detection Methods

Early work by Fishler and Elschlager [Fis73], while not precisely a sequential edge detection technique, nevertheless represents one of the earliest efforts to recognize the fact that improved performance at low signal-to-noise ratio can be accomplished by considering edge contours as a whole rather than local points. In their method, hypotheses consist of "embedded" edge contours. For each such embedded contour, an associated cost is calculated. A dynamic programming technique is used to search the (enormous) space representing all possible contours in an effort to find that with the lowest associated cost. In a similar fashion, Griffith [Gri73] has also used a dynamic programming technique. Specifically, he uses the Viterbi algorithm, a forward dynamic programming method [For73].

Both of these methods have two common problems. Dynamic Programming techniques attempt to find the *best* hypothesis in a search. Although they are more efficient than an exhaustive search, the amount of computation for even modest image sizes is still very large. Secondly, the associated cost of an embedded contour is an ad-hoc quantity requiring a considerable amount of tailoring to specific images.

In an effort to reduce the amount of computation involved with dynamic programming methods, Chien and Fu [Chi74] have proposed the use of what is known as depth-first tree searching [Nil71]. They explicitly formulate the edge detection problem as a search through a rooted tree. Their branch "costs," however, are highly specialized to the type of image under consideration and employ a great deal of a-priori information. In a similar fashion, Martelli [Mar76] formulates the problem as a graph search and uses the A^* algorithm [Nil71] to perform the search. This algorithm is quite similar to the Z-J or stack algorithm described in the next Chapter. Again, Martelli's cost function is ad-hoc and peculiar to the type of image under consideration.

Extending the work of Martelli, Cooper [Coo79] has also used the A^* search algorithm. He has attempted to take some of the arbitrariness out of the cost function by basing that function on a likelihood statistic. He models hypothesized edge contours as a Markov process. The image is modeled as consisting of two region types, "background" and "object," separated by the edge contour. The two types are assumed to be of different but constant intensity. The cost statistic is then the joint likelihood of the particular hypothesized edge contour and the image pixel values, given the assumption that all pixels inside the contour belong to the object at one gray level value and all outside belong to the background at the other gray level value. Note that for *each* hypothesized edge contour, the statistic must be calculated over the entire image.

This method represents one improvement over its forerunners but exhibits several serious drawbacks. On the plus side, the cost function is at least statistical in nature and so has a better theoretical basis than the heuristic functions discussed above. This allows for some performance analysis. Also, the Markov edge contour model captures an important characteristic of real edges as we shall discover in the next Chapter. On the other hand, the assumption of only two image pixel types, "object" and "background," independent and of constant gray level values, is highly unrealistic. In practically any real image of interest, pixel gray level values within and outside of objects may vary considerably due to lighting inhomogeneities, shadows, object inhomogeneities, etc. Pixels are almost never stochastically independent, a subject to which Chapter 4 is entirely devoted. Furthermore, such a problem statement is only useful for finding *bounding contours* of objects and is useless for finding internal edges, intersecting edges, etc., all of which may be important to subsequent processing. Since attempting to find the optimum contour in terms of this likelihood statistic is exceedingly time consuming even under the very conservative assumptions mentioned, the investigator has also employed a sub-optimum search using the A^* algorithm. All of the previous assumptions are still required, but the likelihood statistic is calculated only over the pixels in a "swath" near the hypothesized contour. This results in a search procedure that is very close to what we call sequential searching. Unfortunately, the highly restrictive assumptions still limit its applicability.

Ashkar and Modestino [Ash78], similar to Chien and Fu, start by formulating the edge detection problem as a tree search. They make the important step of applying the search to the output image of an edge operator rather than to the original image. This helps overcome some of the shortcomings of Cooper's approach. This will be discussed in Section 2.1. The Z-J or stack algorithm (Section 2.3.2) is used to perform the search. This method represents a fully mature example of sequential detection with additive branch costs or metrics and a truly sequential search in the sense of Forney's definition (Section 2.3.1). However, its metric suffers from two problems: it is again very ad-hoc in nature, with components that depend on experimentally determined parameters and look-up tables and it requires a *prototype contour*. This latter is a contour provided by some a-priori knowledge base that helps to guide the search toward some preconceived estimate of the final edge map. This represents rather high quality a-priori information and, while possibly appropriate to certain narrow classes of problems, is a severe limitation to the method's generality.

This metric formulation furthermore limits any analytical treatment of the method. In particular it is implicitly *assumed* that: 1.) the metric is indicative of the likelihood that the corresponding path coincides with the true edge contour, 2.) the metric possesses the correct conditional drift, 3.) the correct path is not ever purged from the search stack due to considerable searching of incorrect paths, and 4.) the correct path eventually arrives at the top of the stack. (The same objections might be made of the other techniques discussed in this Section.) Beginning in Chapter 2, these ideas will be carefully defined and addressed. It will be shown that it is possible, through a more rigorous derivation of a stochastic path metric, to give quantitative answers to these questions.

A rather different method proposed by Suk and Hong [Suk84], while not truly sequential in nature, also emphasizes the importance of detecting edge segments over edge points in low signal-to-noise ratio images. The cost function is composed of a variety of statistical tests, but the method does not make use of the computationally efficient searching algorithms described above. As in the other cases, no analytical treatment of the search dynamics is undertaken.

Finally, there are a number of techniques which are also loosely termed sequential edge detectors that perform their sequential search strictly along scan lines of a raster scanned image. That is, rather than attempting to find a path through a tree representing a two-dimensional edge contour in an image, they merely try to locate the position of an intensity transition along horizontal (or vertical) scan lines. Although this is useful for finding bounding contours for simple, homogeneous objects, we will restrict our use of the term *sequential search* to the two-dimensional problem rather than include this essentially one-dimensional search. Two examples of this approach are those by Basseville et al. [Bas81] and Hansen and Elliot [Han82].

With this introduction to edge detection and previous work in sequential techniques, we now turn to a new approach. This method, to be called Sequential Edge Linking (SEL), is an attempt to formalize the use of sequential searching as an alternative to the use of local thresholds as a classification criteria on the output of edge operators. Since it is a processing step *following* the application of an edge operator, it is largely independent of which specific operator is used. The intent is to allow the integration of many observations *along* an edge contour to enter into the classification process in order to improve the performance at low signal-to-noise ratio, ease the difficulty in choosing a threshold, and eliminate the incidence of streaking in the resulting edge map. It furthermore has the advantage that an analytical theory may be brought to bear on fundamental questions of its performance and operational dynamics. This theory also paves the way for the technique to make use of correlation in the image data. All of these ideas will be developed in the following Chapters.

CHAPTER 2

SEQUENTIAL EDGE LINKING

2.1. Introduction

In this Chapter, we will introduce the concept of Sequential Edge Linking, contrast its behavior with that of the other sequential techniques discussed earlier, and describe in some detail the operation of the algorithm. We will begin with a fairly general discussion of the philosophy behind this approach. The various definitions required for the discussions to follow will then be developed. A path branch metric for sequential searching in images will be derived that maximizes a joint probability criterion. This metric will be shown to possess several important characteristics for its use in a sequential tree searching algorithm. This development is followed by an examination of a number of algorithms for performing such a search. Although the treatment of these topics will be in depth, it will stop short of any performance analysis which is reserved for the next Chapter. In addition, the simplifying assumption of independent observations will be used here with the generalization to correlated observations to be developed in Chapter 4.

Let us return to the problem of finding linear features. We will view this as a post-processing task to follow the application of an emphasis operator. We consider such an operator to be any mathematical operator that, when applied to a digital image, produces an estimate of the magnitude and direction of the desired linear feature at every point in the image.

Some pre-processing (i.e. using an operator) is prudent. Although it might be possible to perform a sequential search directly on the raw image, this results in the need to introduce additional parameters. Use of a preprocessing operator decomposes the feature extraction task into two parts: emphasis and detection. The operator serves to emphasize the feature in which we are interested. The sequential searching algorithm then attempts to perform the detection process based on this emphasized information. This allows the detection algorithm to be independent of the feature. For example, to perform intensity edge detection, the preprocessing operator is designed to emphasize intensity discontinuities in the image. Likewise, the operator may be designed to emphasize lines, roof edges, texture edges, or color edges. In any case, its output is expected to be a function of the magnitude and/or the direction of that particular feature at every point in the image.

In the case of intensity edges, the preprocessing operator generally attempts to estimate the magnitude and direction of the local intensity gradient. However it usually also employs spatial low-pass filtering to avoid unacceptably high probability of false detection in situations with poor signal-to-noise ratio. By incorporating such filtering in the edge operator, which is applied over the entire image, one may take advantage of certain computational efficiencies [Can83].

We will assume for now that gradient magnitude and direction information represents the input to our system. Our goal is to determine connected lines representing intensity edges in the original image while not responding to spurious effects of noise, etc. We would like to achieve as much positional accuracy as possible, but due to the searching technique, *subpixel* accuracy cannot be obtained.

A crucial observation is that it is *not* desirable to quantize this gradient information any more coarsely than necessary. We certainly do *not* want to threshold this data (quantize to one bit) globally, locally, adaptively, with hysteresis, or whatever. Nor do we wish to non-maximum suppress it [Can83]. The experience of detection theory, coding theory, and related disciplines states that coarse quantization of noisy, correlated observations independently of one another may be expedient or acceptable where the signal-to-noise ratio is high, but leads to poor performance when the signal quality deteriorates (cf. [Sch80],[Mel78]).

Instead of making such hard decisions (thresholding) right away, experience dictates one "integrate" the soft information over many observations before concluding on a decision. This "integration" takes a variety of forms depending on the application. For example, radar problems utilize sequential detection theory [Pet54]; all decoding algorithms in coding theory base decisions on blocks or long sequences of received data [Vit79]; examples of correlational detection abound. Sequential Edge Linking is one manner in which such integration may be performed in the context of image edge detection.

The image processing literature has not entirely ignored this fact. Non-isotropic, highly directional edge operators make explicit use of this idea. Canny, in [Can83] has calculated the actual performance improvement to be derived from long operators stretched in the direction of an edge. These suffer from two related disadvantages: they are typically limited to short, straight segments or a small number of curved segments, and they are computationally prohibitive as their size increases (since they must be convolved in all orientations over all points of an image). As we shall see, even with certain restrictions imposed, the number of possible configurations grows exponentially with the number of pixels in the operator (see Section 2.2.2). Certainly, decisions based on hundreds of pixels, routine for SEL, are unthinkable by these methods.

Relaxation-based methods do make use of soft edge magnitude and direction information as well as the interaction between neighboring pixels [Zuc76]. This reportably accounts for their high performance in certain situations [Zuc78b]. The trouble with relaxation lies with its convergence properties. In particular, in many situations convergence of the algorithm is not even guaranteed or the solution is not known to be unique [Zuc78a]. No performance bounds can be determined or estimated. In short, one must resort to a "try and see" approach. Furthermore, the nature of the pixel interdependence specified in the algorithm cannot take advantage of any knowledge of the true correlation statistics.

Finally there are the sequential searching techniques. Various investigators have attempted to employ sequential tree or graph searching algorithms in this context. Ashkar and Modestino [Ash78] and Cooper [Coo79] seem to have come the closest to SEL. With the former, the metric used to guide the search is purely ad hoc and so cannot benefit from any analysis aimed at determining its behavior or probability of success. More seriously, however, their technique makes explicit use of a "training contour" to guide the search. This represents very high quality a-priori information. It is noted that in certain specific circumstances such information may indeed be available, but it greatly reduces the generality of the method. The second technique is quite different from SEL and, as noted earlier, suffers from quite restricting assumptions concerning the mean gray level of pixels and their independence. While the results are generally good on artificial images where these assumptions are explicitly met, their performance on real images is yet to be demonstrated. It can be said, however, that the path searching technique of both of these approaches is very similar to that of SEL, as is the philosophy for doing so.

Having obtained the gradient magnitude and direction information from an edge operator (plus spatial filter; see Section 2.2.4), it seems tempting to suggest a maximum likelihood search such as [Coo78]. To do this, one might hypothesize all possible edge paths of say n pixels in length, calculate their respective a-priori likelihoods, assuming some distribution on the gradient levels, and pick the largest. However, the exponential growth of the candidate configurations with n dooms this approach, regardless of the maximum likelihood algorithm used. The situation is analagous to the problem of decoding convolutional codes in coding theory. One may use ML techniques such as dynamic programming (Viterbi decoding) as long as the constraint length is small (< 9 or 10) [Lin83]. Beyond this, the exponential growth of computation and memory requirements with constraint length make the algorithm too unwieldy.

The solution to the decoding problem with large constraint lengths, sequential decoding, is the inspiration behind SEL. In both situations one does not attempt to explore all possible paths in the tree, picking the ML path, but rather one explores only a small subset of them with a high probability of including the correct one [Woz61]. This exploration is accomplished sequentially, extending only those paths which hold high promise of being the correct one.

This approach has several promising virtues. First, it can explore trees with large "constraint lengths" or memory very efficiently and at high speed. Sequential decoding of codes with constraint lengths of up to 50 and at speeds in excess of 10 Mb/s is not uncommon [Lin83]. Secondly, disregarding the problem noted in the next paragraph, the error performance is generally very good. Thirdly and perhaps most importantly the analytical machinery developed for the analysis of sequential decoding may be brought to bear, with a number of important modifications, on the edge detection problem. This allows important inferences to be drawn regarding performance bounds, probability of error events, etc.

On the negative side, such sequential algorithms suffer a variety of problems resulting from their variable computation time [Sav66]. As signal-to-noise ratio degenerates, the number of possible paths explored increases. In data transmission systems, this can lead to buffer overflow and burst errors. With image edge detection, this means only that processing time is delayed. Looked at another way, SEL can take advantage of good signal-to-noise ratio by exploring fewer paths. In any case, processing time is variable, depending on the quality of the edge information present.

One final note. By virtue of its path metric, SEL can provide subsequent processing tasks with *quantitative* information concerning the confidence in segments or whole edges. Other attributes such as length, curvature, direction, and an ordering of the constituent pixels are explicitly determined and immediately available to higher level tasks.

2.2. Definitions

2.2.1. Images, Random Fields

For our purposes, the terms image or digital image will refer to a sample function of a two-dimensional discrete parameter random field. See [Ros76] for a general discussion of digital images or [Won71] for a more rigorous treatment of random fields.

Sample functions of this random field consist of a rectangular array of numbers. These numbers can represent various quantities depending on the nature of the field. If the array is an image of a natural scene, the numbers may represent the gray level or intensity of light at that point of the scene. Alternatively, they may represent an estimate of the magnitude of the gradient of such gray levels or the direction of the gradient. Whatever their interpretation, these numbers, henceforth to be called pixel values, pixels, or observations are defined at points of a rectangular lattice. The points of the lattice are called nodes, their spacing is uniform and equal in both directions, and they are indexed by $I \times I$ where I is the set of integers.

For a given random field, we assume the existence of a probability space (Ω, A, P) and a measurable function, $f_{\vec{r}}(\omega)$; $\omega \in \Omega$, $\vec{r} \in I^2$ defined on this space:

$$f_{\vec{r}}(\omega) \ : \ \Omega \times I^2 \rightarrow F \subset R \qquad (2.1)$$

Here, $\vec{r} = (r^1, r^2)$ is the pair of coordinates of a node in the array, and $f_{\vec{r}}(\omega)$ is the pixel value at that node, and F is some countable subset of the set of real numbers.

Due to the rectangular nature of the lattice, each node has a unique set of eight *neighbors* [Ros70]. The node at coordinates (r^1, r^2) has the neighbor set:

(r^1+1,r^2), (r^1,r^2+1), (r^1-1,r^2), (r^1,r^2-1), (r^1-1,r^2-1), (r^1-1,r^2+1), (r^1+1,r^2-1), (r^1+1,r^2+1). Any ordered set of nodes for which all adjacent pairs are neighbors will be called *connected.* An ordered pair of neighbors also uniquely defines a *direction.* Eight such directions are therefore possible.

2.2.2. Paths

A *path* will be defined as a connected set of nodes with the following property: for any subset of three nodes in the ordered set, the directions defined by the first two nodes and by the second two nodes differ by $\pi/4$ or less. Thus, if one were to draw line segments between nodes of a path, the resulting curve would contain no changes of direction greater than or equal to $\pi/2$ (see Figure 2.1).

Paths may be denoted in one of two ways. The first is simply the ordered set of nodes comprising the path:

$$\mathbf{m} = [(r_1^1,r_1^2),(r_2^1,r_2^2),\ldots(r_n^1,r_n^2)] \tag{2.2}$$

The second method is by specifying a *start node* and *direction,* and an ordered set of *letters:*

$$\mathbf{m} = (r_0^1,r_0^2) \times \vec{d}_0 \times [a_1,a_2,a_3,\ldots a_n] \tag{2.3}$$

where (r_0^1,r_0^2) is the start node, $\vec{d}_0$ is the start direction, and the a_i are taken from an alphabet A = [L,S,R]. The letters of the alphabet stand for left, straight, and right respectively. The coordinates of the nodes of the path may be obtained in a recursive fashion from the letters a_i. The first node is obtained from the start node by moving in the array in the direction $\vec{d}_0$. The direction to the second node is then obtained from

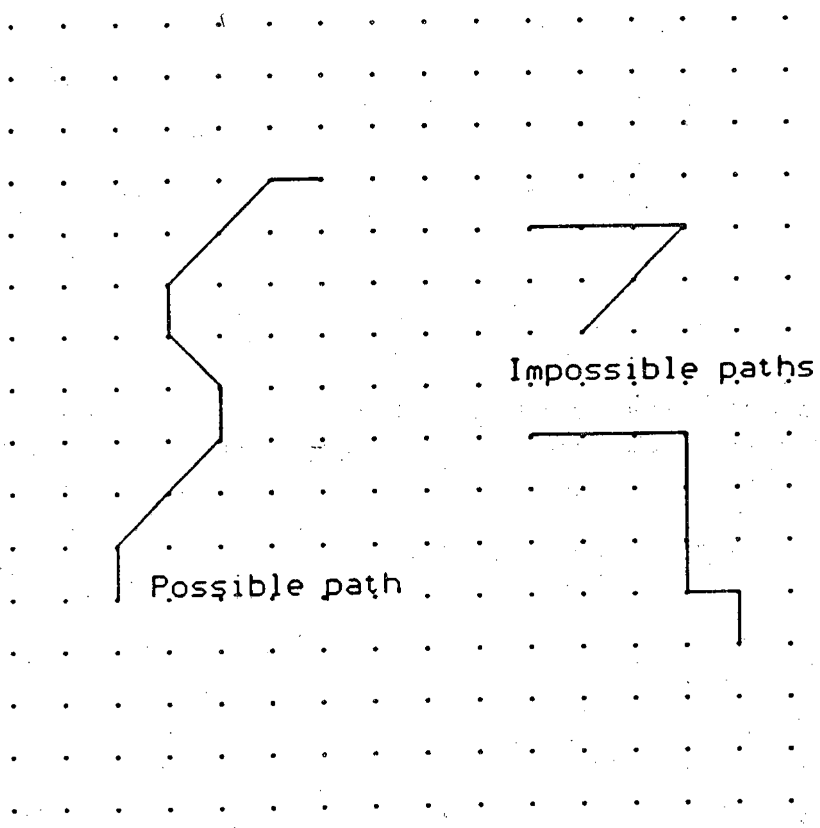

Figure 2.1: The path topology on the left is permissible since all changes of direction are of $\pi/4$ or less. The path topologies on the right contain changes of direction greater than $\pi/4$.

$\vec{d}_0$ and a_1 (and so on) by the obvious rule:

$$\vec{d}_1 = \left\{ \begin{array}{ll} \vec{d}_0 + \frac{\pi}{4} & \text{if } a_1 = L \\ \vec{d}_0 & \text{if } a_1 = S \\ \vec{d}_0 - \frac{\pi}{4} & \text{if } a_1 = R \end{array} \right\} \tag{2.4}$$

This process can likewise be reversed to obtain the list - of - letters description of a path from the list of node coordinates. Thus, either description uniquely specifies a given path and they are equivalent.

The second path description has been introduced in order to clarify the following model for paths. We will assume that paths may be modeled as a k^{th} order Markov Chain [Ros83]. In the following discussion, we will focus on the letters a_i of paths and take up the question of the start node and direction later.

Let us consider a discrete time stochastic process S_i, i = 0,1,2,3, . . . :

$$S_i\ (\omega) :\ \Omega \rightarrow \Sigma \tag{2.5}$$

The underlying space Ω is just the space of all possible state sequences:

$$\Omega = (s_0, s_1, s_2, \ldots) \tag{2.6}$$

and Σ is the state space. When the process enters the state at $i+1$ from that at i, it outputs a letter $a_{i+1} \in A = [\text{ L,S,R }]$. The state of the system is defined to be the ordered sequence of the last k letters put out by the system:

$$s_i = (a_i, a_{i-1}, \ldots, a_{i-(k-1)}) \tag{2.7}$$

so that $\Sigma = A \times A \times \ldots \times A = A^k$. We will assume s_0 is fixed and known. The Markov assumption is then:

$$Pr\ (S_{i+1} = s_{i+1} \mid s_i, s_{i-1}, \ldots, s_0) = Pr\ (S_{i+1} = s_{i+1} \mid s_i) \tag{2.8}$$

or:

$$Pr\ (a_{i+1} \mid s_i, \ldots, s_0) = Pr\ (a_{i+1} \mid s_i) \tag{2.9}$$

from which it follows by the chain rule:

$$\begin{aligned} Pr(\mathbf{m}) &= Pr(a_1, a_2, \ldots, a_n) \\ &= Pr(s_n \mid s_{n-1})\, Pr(s_{n-1} \mid s_{n-2}) \ldots Pr(s_1 \mid s_0) \end{aligned} \tag{2.10}$$

This is our stochastic model for the *path process*. It bears a strong resemblance to that used in [Coo79]. By manipulation of the transition probabilities, $Pr(s_{i+1} \mid s_i)$, certain types of paths achieve higher probability of occurrence than do others. For example, the model may favor paths exhibiting long segments of straight transitions over those that meander incessantly with highly probable L to R and R to L transitions. Such a nonuniform distribution of transition probabilities may be experimentally verified for images. The Appendix describes a program that traces edge lines in images of representative objects and calculates estimates of state transition probabilities based on their frequency of occurrence. The estimates given in the examples there clearly show that all the transitions are not of equal probability.

2.2.3. Path Metric

The crucial concept in any sequential tree searching algorithm is that of a path metric. The algorithm's efficiency and chance of success lie in its ability at any point of execution to rank all the paths explored so far by a measure of their "quality". This measure is the path metric. The path branch metric of Ashkar and Modestino [Ash78] is an example, but it suffers from a lack of generality and is largely ad hoc. In this Section, we will derive a path branch metric that optimizes a joint probability criterion.

The concept of a branch metric for sequential tree searching was first introduced in the problem of sequential decoding of convolutional codes by R. M. Fano [Fan63]. That metric is probabilistic, reflecting the statistical nature of the underlying coding theory. The metric we introduce is also probabilistic. This is in part due to the models we are assuming for images and paths, but also because it allows a statistical treatment of the operation and performance of the algorithm.

Path metrics should exhibit certain desirable qualities. First, they should not be biased by path length. That is, the metric should not favor a long path over a short one just because of its length or vice versa [McE77]. All paths need to be compared on the basis of their probability of being an edge path regardless of length. Second, they need to be efficient to compute. This is best served by a metric that is recursively computed: the metric of a path to a certain node is obtained from that to the preceding node by adding a "correction value" that depends only on the new node. This eliminates the necessity for calculating the metric anew over the nodes of a long path every time it is extended by a node. This is very important from the standpoint of efficiency and becomes an even more crucial requirement when the observations are correlated (see Chapter 4). Finally, the metric should exhibit the all - important "drift" characteristics. That is, it should tend to increase along true edge paths and decrease elsewhere. This is critical to ensure that a sequential searching algorithm proceeds correctly.

Our path metric will be defined for paths in a random field as defined in the preceding Sections. We will assume the following two measures on the random field. The first, $p_1(f_{\vec{r}} = y)$, is the probability the field value or observation at node $\vec{r}$ is y conditioned on the hypothesis, H_1, that that node is on an intensity edge in the original image:

$$\begin{aligned} p_1(f_{\vec{r}} = y) &= Pr\,(f_{\vec{r}} = y \mid H_1) \\ &= Pr\,(f_{\vec{r}} = y \mid \vec{r} \textit{ is on an edge }) \end{aligned} \tag{2.11}$$

The second is the probability of the observation conditioned on the null hypothesis, H_0, that the node is randomly positioned in the field with respect to intensity edges:

$$\begin{aligned} p_0(f_{\vec{r}} = y) &= Pr\,(f_{\vec{r}} = y \mid H_0) \\ &= Pr\,(f_{\vec{r}} = y \mid \vec{r} \textit{ is a random node }) \end{aligned} \tag{2.12}$$

Note that, at this point, we are not even specifying what the random field, f, represents but merely assuming the existence of these two conditional measures on that field. It will later turn out that the field is a gradient magnitude or gradient direction field or a function of both.

For a given path, $\mathbf{m}$,

$$\begin{aligned} \mathbf{m} &= [\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_n] \\ &= \vec{r}_0 \times \vec{d}_0 \times [a_1, a_2, \ldots, a_n] \end{aligned} \tag{2.13}$$

we now define the ***likelihood ratio***, $l(\mathbf{f})$, of the path in the conventional manner as [Van68]:

$$l(\mathbf{f}) = \frac{p_1(f_{\vec{r}_1}, \ldots, f_{\vec{r}_n})}{p_0(f_{\vec{r}_1}, \ldots, f_{\vec{r}_n})} \tag{2.14}$$

This is simply the joint probability of the observations along the path conditioned on the H_1 hypothesis divided by the joint probability of those values conditioned on the H_0 hypothesis.

From Equation (2.10), we have the probability of the path $\mathbf{m}$ is:

$$Pr\ (\mathbf{m}) = \prod_{i=1}^{n} Pr\ (s_i \mid s_{i-1}) \tag{2.15}$$

We are now in a position to define our path metric. For the given path $\mathbf{m}$, the path metric, $\Gamma(\mathbf{m},\mathbf{f})$, will be defined as the logarithm of the product of the path probability and its likelihood ratio:

$$\Gamma(\mathbf{m},\mathbf{f}) = \ln\ [Pr(\mathbf{m}) \cdot l(\mathbf{f})] \tag{2.16}$$

If we make the very important assumption that the random variables of the field are stochastically independent, then the likelihood ratio becomes:

$$l(\mathbf{f}) = \frac{p_1(f_{\vec{r}_1})\, p_1(f_{\vec{r}_2}) \cdots p_1(f_{\vec{r}_n})}{p_0(f_{\vec{r}_1})\, p_0(f_{\vec{r}_2}) \cdots p_0(f_{\vec{r}_n})} \tag{2.17}$$

and Equation (2.16) simplifies to:

$$\Gamma(\mathbf{m},\mathbf{f}) = \sum_{i=1}^{n} \left[\ln \frac{p_1(f_{\vec{r}_i})}{p_0(f_{\vec{r}_i})} + \ln Pr(s_i \mid s_{i-1}) \right] \tag{2.18}$$

The above assumption, known as the statistical independence of the observations, is an important one. It allows the factorization of the joint probabilities of the likelihood ratio which in turn gives the path metric its recursive computational efficiency. Note that path $\mathbf{m}$ may be extended by a node and the new metric obtained from the old by merely adding a term:

$$\Gamma_{n+1} = \Gamma_n + \left[\ln \frac{p_1(f_{\vec{r}_n+1})}{p_0(f_{\vec{r}_n+1})} + \ln Pr(s_{n+1} | s_n) \right] \qquad (2.19)$$

In Chapter 4 we will treat the situation of correlated observations where this assumption is no longer true.

The path metric of Equation (2.18) is a sum of branch terms each of which has two components. These components play different roles. The first is a likelihood ratio of the probability of observing the random variable $f_{\vec{r}_i}$ under the hypothesis H_1 to the probability under the null hypothesis. This component therefore is a function of the *data* in the image. The second component is the branch transition probability and is a measure of the *a-priori* probability that the edge path proceeds in that direction, given the last k branch directions.

The stochastic nature and recursiveness of our path metric are clear. The fact that it possesses the correct "drift" characteristics will now be demonstrated. For this purpose, we will find the conditional mean of the statistic, $\frac{\Gamma(\mathbf{m},\mathbf{f})}{n}$, i.e. the path metric normalized by its length:

$$E\left\{\frac{\Gamma(\mathbf{m},\mathbf{f})}{n} \mid H_1\right\} = E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln\frac{p_1(f_j)}{p_0(f_j)} + \ln Pr(s_j \mid s_{j-1})\right] \mid H_1\right\} \quad (2.20)$$

Now since the path process is stochastically independent of the random field under the H_1 hypothesis, the right hand side becomes:

$$E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln\frac{p_1(f_j)}{p_0(f_j)}\right] \mid H_1\right\} + E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln Pr(s_j \mid s_{j-1})\right]\right\} \quad (2.21)$$

Considering the first term above, since the observations, f_j, are equidistributed, the expectation of the sum is equal to the sum of n identical expectations, so that:

$$\begin{aligned} E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln\frac{p_1(f_j)}{p_0(f_j)}\right] \mid H_1\right\} &= E\left\{\ln\frac{p_1(f)}{p_0(f)} \mid H_1\right\} \\ &= \sum_{f} p_1(f)\ln\frac{p_1(f)}{p_0(f)} \end{aligned} \quad (2.22)$$

by the definition of the p_1 measure. This last quantity is seen to be the Kullback's Information [Kul59] between the p_1 and p_0 measures, denoted $I(p_1 \mid p_0)$.

The second term of Equation (2.21) can be rewritten using the chain rule as:

$$\begin{aligned} E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln Pr(s_j \mid s_{j-1})\right]\right\} &= -E\left\{\frac{1}{n}\ln\frac{1}{Pr(s_n, s_{n-1}, \ldots, s_1)}\right\} \\ &= -E\left\{\frac{1}{n}\ln\frac{1}{Pr(a_n, a_{n-1}, \ldots, a_1)}\right\} \end{aligned} \quad (2.23)$$

which is defined to be the entropy per letter of the Markov source [Gal68]. In the limit as $n \to \infty$, this quantity can be shown [Gal68] to be equal to:

$$-H_{\infty}(S) = -\sum_{i=1}^{3^t} q(i) \sum_{l=1}^{3} Pr(a_l \mid s = i) \ln \frac{1}{Pr(a_l \mid s = i)} \quad (2.24)$$

where $q(i)$ is the time averaged state probability of state i:

$$q(i) = \lim_{n \to \infty} \frac{1}{n} \sum_{j=1}^{n} Pr\,(s_j = i) \quad (2.25)$$

Since all the states in the Markov Chain we have been considering are recurrent and in the same equivalence class, i.e. each state can be reached from any other in one or more transitions and there are no states outside this set, $q(i)$ is independent of the initial state, s_0 [Ros83]. Combining Equations (2.21-2.23), we therefore have the final result that:

$$\lim_{n \to \infty} E\left\{\frac{\Gamma(\mathbf{m},\mathbf{f})}{n} \mid H_1\right\} = I(p_1 \mid p_0) - H_{\infty}(S) \quad (2.26)$$

Likewise, under the H_0 hypothesis, the observations term becomes:

$$\begin{aligned} E\left\{\frac{1}{n}\sum_{j=1}^{n}\left[\ln\frac{p_1(f_j)}{p_0(f_j)}\right] \mid H_0\right\} &= E\left\{\ln\frac{p_1(f)}{p_0(f)} \mid H_0\right\} \\ &= \sum_f p_0(f) \ln\frac{p_1(f)}{p_0(f)} \quad (2.27) \\ &= -I(p_0 \mid p_1) \end{aligned}$$

so that:

$$\lim_{n \to \infty} E\left\{\frac{\Gamma(\mathbf{m},\mathbf{f})}{n} \mid H_0\right\} = -I(p_1 \mid p_0) - H_{\infty}(S) \quad (2.28)$$

We see then that since both $I(\cdot \mid \cdot)$ and $H_{\infty}(S)$ are strictly non-negative, the expected value of the normalized metric conditioned on the hypothesis that the path is not an edge path is negative. When conditioned on the H_1 hypothesis, the normalized metric will be positive if:

$$I(p_1 \mid p_0) > H_{\infty}(S) \tag{2.29}$$

Therefore, as long as Equation (2.29) is satisfied, the metric tends to grow with length if the path is truly an edge path and tends to decrease with length otherwise. This is the desired "drift" tendency that allows the searching algorithm to find the edges in the image. It is interesting to note that the difference between the statistics conditioned on the two hypotheses grows with n as:

$$\begin{aligned} E\left\{\frac{\Gamma(\mathbf{m},\mathbf{f})}{n} \mid H_1\right\} - E\left\{\frac{\Gamma(\mathbf{m},\mathbf{f})}{n} \mid H_0\right\} &= I(p_1 \mid p_0) - I(p_0 \mid p_1) \\ &= J(p_0, p_1) \end{aligned} \tag{2.30}$$

where $J(\cdot,\cdot)$ is Kullback's Divergence. This says that, on average, the difference between the rate at which the metric grows along a true edge path and the rate at which it decreases along a random path is equal to the Kullback's Divergence between the two measures p_1 and p_0.

The fact that this path metric is not biased by path length is more difficult to justify. We will do so by rederiving Equation (2.18) in such a way as to bring out this characteristic. The approach is similar to that used by Massey in the convolutional coding situation [Mas72]. This furthermore gives additional insight into and justification of the proposed metric.

Let us suppose we are given a set, M, of possible paths of varying lengths,

$$\begin{aligned} M &= [\mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^k] \\ \mathbf{m}^i &= \vec{r}_0^{\,i} \times \vec{d}_0^{\,i} \times [a_1^i, \ldots, a_{n^i}^i] \end{aligned} \tag{2.31}$$

where the length of the i^{th} path, $\mathbf{m}^i$, is n^i. As before, the i^{th} path has probability:

$$Pr(\mathbf{m}^i) = \prod_{j=1}^{n^i} Pr(s_j^i \,|\, s_{j-1}^i) \tag{2.32}$$

Of course, each node of every path in M has an associated pixel value. To simplify notation, we will denote the observation corresponding to the j^{th} node of the i^{th} path as:

$$f_{\vec{r}_j}^{\,i} = f_j^i \tag{2.33}$$

The set of f -values associated with the i^{th} path will be denoted:

$$\mathbf{f}^i = [f_1^i, f_2^i, \ldots, f_{n^i}^i] \tag{2.34}$$

Now, just as in [Mas72], we will append a string of a-letters and associated f -values to the end of the shorter paths to make them all the same length. The tail of a-letters, $\mathbf{m}_{tail}^i = [a_{n^i+1}^i, \ldots, a_N^i]$ where N is the length of the longest path in M, are chosen *at random*. Since this path extension is random through the field, the associated f -values are chosen *independently* and at *random* according to a measure, $p_0(f)$, known as the random-path f measure. (It is the same as the p_0 measure defined previously). We assume the values of the random tails are chosen independently of everything else, so that:

$$\begin{aligned} Pr(f^i_{n^i+1}, \ldots, f^i_N \mid \mathbf{m}^i, \mathbf{m}^i_{tail}, \mathbf{f}^i) &= Pr(f^i_{n^i+1}, \ldots, f^i_N) \\ &= Pr(\mathbf{f}^i_{tail}) \\ &= \prod_{j=n^i+1}^{N} p_0(f^i_j) \end{aligned} \qquad (2.35)$$

If we now assume the probabilities of the f -values of any path that coincides with an intensity edge are given by the measure, p_1,

$$Pr(\mathbf{f}^i \mid H_1) = p_1(\mathbf{f}^i) \qquad (2.36)$$

then due to the conditional independence of the observations, we have:

$$Pr(\mathbf{f}^i, \mathbf{f}^i_{tail} \mid H_1) = \prod_{j=1}^{n^i} p_1(f^i_j) \prod_{j=1}^{N-n^i} p_0(f^i_{n^i+j}) \qquad (2.37)$$

Thus the joint probability of tracing path i, observing the values $\mathbf{f}^i$ along it, and adding a random tail, $\mathbf{m}^i_{tail}$, under the hypothesis that path i is indeed an edge path may be written:

$$\begin{aligned} Pr(\mathbf{m}^i, \mathbf{f}^i, \mathbf{m}^i_{tail}, \mathbf{f}^i_{tail}) &= \prod_{j=1}^{n^i} Pr(s^i_j \mid s^i_{j-1}) \prod_{j=1}^{n^i} p_1(f^i_j) \\ &\cdot \prod_{j=1}^{N-n^i} Pr(s^i_{n^i+j} \mid s^i_{n^i+j-1}) \prod_{j=1}^{N-n^i} p_0(f^i_{n^i+j}) \end{aligned} \qquad (2.38)$$

Summing over all possible choices of a random tail, we have:

$$Pr(\mathbf{m}^i,\mathbf{f}^i,\mathbf{f}^i_{tail}) = \sum_{\mathbf{m}^i_{tail}} Pr(\mathbf{m}^i,\mathbf{f}^i,\mathbf{m}^i_{tail},\mathbf{f}^i_{tail})$$

$$= \prod_{j=1}^{n^i} Pr(s^i_j | s^i_{j-1}) \prod_{j=1}^{n^i} p_1(f^i_j) \prod_{j=1}^{N-n^i} p_0(f^i_{n^i+j}) \qquad \textbf{(2.39)}$$

Now an optimum rule for deciding among the paths in M is to choose that path i such that i maximizes $Pr(\mathbf{m}^i,\mathbf{f}^i,\mathbf{f}^i_{tail})$ or equivalently, such that i maximizes:

$$\ln \left\{ \frac{Pr(\mathbf{m}^i,\mathbf{f}^i,\mathbf{f}^i_{tail})}{\prod_{j=1}^{N} p_0(f_j)} \right\} \qquad (2.40)$$

since the denominator is independent of i and the logarithm is a monotonic function. We thus have as the statistic to be maximized:

$$\Gamma(\mathbf{m}^i,\mathbf{f}^i) = \sum_{j=1}^{n^i} \left[\ln \frac{p_1(f^i_j)}{p_0(^i_j)} + \ln Pr(s^i_j | s^i_{j-1}) \right] \qquad (2.41)$$

This is obviously the same metric that was presented in a constructive manner earlier. This derivation lends support to the contention that the metric orders the prospective paths according to a rule that maximizes the joint probability of the path letters and the observed random field values along the path. It also demonstrates that the metric has no built in length bias: that path which maximizes the joint probability exhibits the highest metric. Just as in [Mas72], the use of the random tail is a convenient device for equalizing the length of the observed paths for the purposes of comparing their probabilities. In a sense, the a-priori path probability, which decreases with length, is balanced by the likelihood ratio, which increases with length.

2.2.4. Gradient Operators

As discussed in Section 2.1, the SEL algorithm is usually preceded by the application of a spatial gradient operator. The algorithm operates on the output of this operator rather than on the raw image itself.

The literature devoted to spatial gradient operators for images and other multidimensional signals is quite extensive. The interested reader is referred to several survey articles for more complete bibliographies on the subject [Kun82],[Ros84].

The function of the gradient operator is twofold. First, as its name implies, it forms an estimate of the (discrete) spatial intensity gradient at every node of the field. This includes direction as well as magnitude estimates. The rationale for this is that intensity edges in images generally coincide with large gradient values. Gradient magnitude is a local measure of "edginess" at a node that is independent of the average or baseline intensity in the local region. This high pass (in spatial frequency response) nature is characteristic of all gradient-type signal processors.

Due to the predominance of high (spatial) frequency noise in many images, this high pass response of a gradient operator does little to reduce the average noise power in the operator output. Partly to increase the output signal-to-noise ratio and partly to provide a measure of control over the overall spatial frequency response of the operator, gradient operators often include some low pass spatial filtering as their second function. Again, the choice of filter functions is large, with many "optimum" filters having been derived for specific situations and criteria. Most of the results given later were obtained using a 2-D Gaussian filter. This choice was made for two reasons. The kernel of a 2-D Gaussian filter is separable, i.e. the 2-D convolution can be decomposed into two concatenated 1-D convolutions. This has a dramatic effect on increasing computational efficiency, especially for filters with large support. Secondly, the Gaussian has the unique property among the myriad possible filter functions of having a minimum product of bandwidth and spatial resolution [Mar80]. Thus it offers an excellent compromise between noise rejection (narrow low-pass bandwidth) and edge position blurring (spatial resolution). In fact, Canny [Can83] has shown through a variational Calculus argument that the Gaussian is very nearly equal to the optimum filter shape for intensity step edges in images where the criteria of optimality are edge positional accuracy and output signal-to-noise ratio.

Edge direction information is also useful in detection algorithms. Gradient operators are vector operators and provide estimates of the local gradient direction as well as the magnitude. These two quantities can be combined to provide a *directional gradient operator*. If $\vec{n}$ is a unit vector in a certain direction, then:

$$\nabla I \cdot \vec{n} \tag{2.42}$$

is the gradient of the field I in the direction of $\vec{n}$. It has been argued by various investigators [Har84],[Can83],[Mar76a], that directional operators provide better performance in the presence of noise than do isotropic operators.

When Gaussian filtering is included, a directional gradient operator takes the form:

$$(\nabla G * I) \cdot \vec{n} \tag{2.43}$$

where I is the original field (image), G is a 2-D Gaussian kernel, * denotes 2-D convolution, and $\vec{n}$ is a unit vector in the desired direction. In other words, this operator only considers the component of the gradient in the direction $\vec{n}$; the component of the gradient parallel to $\vec{n}$ is disregarded.

The quantity in Equation (2.43) can be determined at the nodes of hypothesized paths in the following manner. The gradient operator is first applied to the entire image, with the magnitude and direction information stored separately in two output fields. The magnitude field is given by:

$$M = | \nabla G * I | \tag{2.44}$$

and the direction field by:

$$\vec{D} = \frac{\nabla G * I}{| \nabla G * I |} \tag{2.45}$$

When a given node, $\vec{r}$, is visited by the algorithm, the directional gradient at $\vec{r}$ can be found by:

$$f_{\vec{r}} = (\vec{D}_{\vec{r}} \cdot \vec{d}_{\vec{r}}^{\perp}) M_{\vec{r}} \tag{2.46}$$

where $\vec{d}_{\vec{r}}^{\perp}$ is orthogonal to the path direction at $\vec{r}$ and $\cdot$ is a vector dot product. Since the path to $\vec{r}$ defines a certain direction, $\vec{d}_{\vec{r}}$, the normal to this direction, $\vec{d}_{\vec{r}}^{\perp}$, is dotted into the gradient direction at $\vec{r}$ and this is multiplied by the gradient magnitude. Thus, $f_{\vec{r}}$ is precisely the directional gradient of Equation (2.43) at the node $\vec{r}$. Note that $f_{\vec{r}}$ is a function of the *path* direction at $\vec{r}$ as well as the gradient magnitude and direction.

It should be pointed out that (2.43) gives only one possible f-field input to SEL. Many others could be used. For example, the input field could be just the gradient magnitude,

$$f_{\vec{r}} = M_{\vec{r}} \tag{2.47}$$

or it could be the output of the Sobel operator [Dud73], the $\nabla^2 G$ operator of Marr and Hildreth [Mar80], compass gradient masks [Kir71], or any of a large variety of mathematical operators. Equation (2.43) merely represents one possible input field that has been used. The path metric used by the SEL algorithm requires scalar values, $f_{\vec{r}}$, however. Where vector information is present, the path direction is used to reduce it in an appropriate fashion to scalar quantities.

2.3. Sequential Search Algorithms

2.3.1. Sequential Searching

We come now to the important question of sequential searching. So far, we have defined what we mean by paths and can associate with each path a quantitative measure of quality or metric. This metric is a function of the joint probability of the path transition letters and the f -values of the random field along the path.

What is now needed is a systematic way of hypothesizing paths so that they may be compared via their corresponding metrics. As was discussed in Section 2.1, every path in the field cannot be hypothesized because the number of possibilities is enormous. Thus, maximum likelihood detection is impossible. What we do instead is make hypotheses in such a way that only a small subset of possible paths is actually explored, but this subset contains an actual edge path with high probability.

The procedure used is called sequential tree searching and is borrowed from coding theory [Lin83]. It presumes that a *start* or *root node* is given. This start node must be on an intensity edge but is otherwise arbitrary. The selection of start nodes will be examined in Section 2.4. The root node defines a tree of semi-infinite paths in the random field. Because of the special structure of paths, there are precisely 3^n paths of length n (or to *depth* n in the tree) beginning at the root node. The sequential searching algorithm begins at the root node and sequentially examines paths in the tree. The exact procedure by which this is done depends on the algorithm used; these will be discussed in detail shortly. The sequential behavior, however, is common to them all. At each iteration of the algorithm, one of the paths explored so far is extended by one node and the metric for this new path is calculated. Decisions are made based on these metrics as to which path should be extended on the next iteration. This is in accord with the definition of sequential tree searching as defined by Forney [For74b]: a search is sequential only if each new path hypothesized is an extension of a previously examined one and the decision as to which path to extend is based only on the set of paths visited thus far.

Because long paths are built up node by node, it is entirely possible that the algorithm can mistakenly follow a non-edge path for some depth into the tree. In Section 2.2.3, however, we saw that the average tendency of the metric under these circumstances is to decrease with increasing length. Our hope, therefore, is that these forays along incorrect paths will eventually be halted by the resulting decreases in metric. The algorithm at some point settles back onto the correct (edge) path for which the average tendency is for the metric to increase with length. Of course, this average behavior of the metric does not *guarantee* that all incorrect paths will eventually be abandoned. Such questions can be formulated and answered in a quantitative manner, but this analysis shall be addressed in Chapter 3. For now, it is sufficient to consider only this general, expected behavior in order to obtain some insight into the workings of the algorithms.

We now turn to the searching algorithms themselves. In this Section we will only describe their operation. Analysis will be deferred to Chapter 3.

2.3.2. The Stack Algorithm

Originally proposed independently by Zigangirov [Zig66] and Jelinek [Jel69], the Z-J or stack algorithm is a very efficient method of performing sequential searching in a tree, especially when the implementation is in software. It is also the easiest of the sequential algorithms to understand and so provides the most accessible route to gaining insight into how sequential searching works. This is the reason it is treated first in this Section despite the fact that it was historically predated by the Fano Algorithm.

At the heart of the Z-J algorithm is a stack or priority queue [Aho74]. Each entry in the stack consists of a path and its corresponding metric. These entries are ordered by their metric, with the "top" of the stack occupied by the path with the largest metric.

When the algorithm is first invoked, the stack is initialized by placing the root node at the top of the stack and assigning it a metric of 0. At this point a slight deviation from the classical Z-J algorithm must be accommodated due to the random field model on which paths are based. This modification is to specify a root direction as well. This direction information may be obtained directly from the gradient direction map or from a local computation on the f -field values.

Having specified a root node and direction, exactly three paths to depth 1 in the tree are defined. These are the three descendents of the root node. The first iteration of the stack algorithm proceeds as follows. The root node is deleted from the stack, the three descendents to depth 1 are placed onto the stack and ordered by their corresponding metrics. Subsequent iterations are performed by deleting the top - most path from the stack, finding its three descendent paths, calculating their metrics, and inserting them onto the stack according to their metrics. Thus at each iteration, the current best path is extended by one node. The current best path can be any path examined thus far since the largest metric always percolates to the top of the stack regardless of when it was last extended.

This algorithm brings out several important characteristics of all sequential searching algorithms. Perhaps most importantly from an operational standpoint, the amount of computation is not fixed or a simple function of path length. When the noise becomes more and more severe, the algorithm investigates more false paths and to larger depths in the tree before abandoning them. When the signal-to-noise ratio is high, the true edge path consistently has the best metric and so very little computational effort is spent on incorrect paths. This variable computational effort is typical of sequential searching algorithms and has received much attention in the coding literature [Sav66],[Gei73]. This is because data buffer overflows resulting from highly variable computational load under conditions of poor SNR are a major cause of output errors for sequential decoding. It is interesting to note that its application to image edge linking does not suffer from this particular problem. In effect, the data in its entirety is always available to the algorithm as the input field or fields (e.g. gradient magnitude and direction fields) are assumed to be given. The algorithm cannot therefore "fall behind" relative to constantly arriving data as in the coding/decoding situation. Of course, it is still true that additional delay must be accepted in these situations.

This algorithm clearly illustrates the characteristic that very long paths can be explored at the expense of not performing maximum likelihood decisions. Paths to depths, n, of hundreds of nodes in the tree can be readily examined by the algorithm but obviously not all to the 3^n possible paths are examined for such large n. The vast majority are never even taken up by the algorithm because they exhibit such poor metrics in the first few nodes compared to the relatively few paths that are taken to great depths. It is conceivable that the correct path is among those that are discarded early, but it will be shown (Chapter 3) that following an incorrect path to large n has a probability that decreases exponentially in n. Thus, sequential searching cannot guarantee selection of the most probable path among all possible paths (ML). It does, however, allow the most promising paths to be explored without expending computational effort on large numbers of unpromising paths.

One characteristic of the stack algorithm that is not shared by all sequential algorithms is its large memory requirement. Maintenance of the stack when exploring long paths requires the use of a significant amount of memory. The algorithm is thus suited to applications where the implementation is on software.

2.3.3. Variations on the Stack Algorithm

Three variations on the original Z-J algorithm have arisen in the coding literature. The first is the stack-bucket algorithm of Jelinek [Jel69]. This is an attempt to partially alleviate a time consuming aspect of the stack algorithm. As the number of entries in the stack becomes large, it takes increasingly longer to insert successor paths into their appropriate place on the stack. With the stack-bucket algorithm, no ordering of the stack is performed at all. Instead, the stack is divided into a number of segments called buckets, with each bucket corresponding to an interval of possible metric values. With each iteration of the algorithm, paths are placed in the bucket appropriate to their metrics. No ordering in the buckets takes place. The path to be extended is simply taken from the top of the highest non-empty bucket. Note that this is not necessarily the best current path, but only a very good one, unless the highest non-empty bucket contains only this one path. On the other hand, the placement of paths onto the stack involves a computation that depends only on the number of buckets there are and does not grow as the stack increases in length. Thus, this algorithm trades away some performance for a substantial improvement in speed.

A second variation, introduced by Haccoun and Ferguson [Hac75], is called the generalized stack algorithm. In this case, paths are organized and extended just as in the Z-J algorithm but more than one path can be extended at the same time. This algorithm also can detect remerging paths. When two paths remerge, the path with lower metric is discarded from the stack, thus eliminating unwanted baggage. This modification seems to influence the buffer overflow problem of decoding. In view of its added complexity, its value in image edge linking is questionable.

The third variation is the multiple stack algorithm (Chevillat and Costello [Che77]). This algorithm completely eliminates the buffer overflow problem by forcing the decoder ahead along "reasonably good" paths in times of much noise, rather than continuing the search for the best path. The manner in which this is done involves the judicious use of additional smaller stacks to which the algorithm turns when the main stack fills up. The smaller stacks limit the search space of the algorithm and thus speeds up its penetration of the tree. This technique may very well have application to the image problem.

2.3.4. The A * Algorithm

The A * algorithm is a heuristic tree-searching algorithm originating in the artificial intelligence literature [Nil71]. In operation it differs little from the stack algorithm. The search is once again sequential with the best path extended at each iteration by a node. In this case, best is taken to mean that path with the lowest associated cost (an inverted metric). This cost is computed recursively by summing branch costs associated with each path transition from the start node to the current node. The algorithm includes a provision for eliminating inferior paths when remergings occur. The primary difference between A * and the Z-J algorithm is that the former also provides for the inclusion of a cost associated with *completing* the path from the current node to some specified *goal* node. This completion cost as well as the specification of the goal node must be provided by some a-priori or heuristic information source. It has been shown [Har68] that if this completion cost is a lower bound on the minimal cost path from the current node to the goal, then the algorithm will find an optimal, minimum cost path to the goal. If no such heuristic information is available, A * reduces essentially to the Z-J algorithm.

2.3.5. The Fano Algorithm

The Fano Algorithm, named after its originator, R.M. Fano, was actually the first algorithm to be developed for sequential searching [Fan63]. This algorithm is particularly well suited for hardware implementations of sequential decoders as it requires very little memory. It is, however, more difficult to understand than the algorithms discussed above. The interested reader is referred to [Lin83] or [McE77] for more complete expositions of this algorithm.

The Fano algorithm never makes large jumps about the tree as do the various stack algorithms. Instead, its movements are confined to three types: forward moves, from a given node to one of its successor nodes; backward moves from a successor to a predecessor node; and sideways from a successor node to another successor node of the same predecessor. Decisions as to which move to make at any particular time are based on comparisons between the path metric associated with the node under consideration and a running threshold, T. This running threshold is practically the only quantity that must be stored and always changes by multiples of some constant, Δ.

When the algorithm is constantly moving forward, deeper into the tree, the threshold is increased by multiples of Δ such that $\Gamma - \Delta < T \leq \Gamma$. Such forward moves are only allowed so long as $\Gamma \geq T$ for the new node. If none of the successor nodes satisfies this requirement, the algorithm executes a series of backward and sideways moves in an effort to find a new node such that $\Gamma \geq T$. If none can be found, the threshold is decremented by Δ and the search continues. The decision structure of the algorithm is so constructed that it is impossible for the algorithm to lodge itself in an infinite loop [McE77]. Furthermore, the Fano algorithm practically always chooses the same path as does the stack algorithm [Gei73].

2.4. Root Node Selection

The operation of the various algorithms described in the Section above is critically dependent on the identification of a root node on each intensity edge of interest. The only requirement imposed on these root nodes is that they actually lie on the edge. This Section addresses the selection of these nodes.

2.4.1. ROC Curves for Gradient Operators

It seems reasonable to consider the output of the spatial gradient operator (Section 2.2.4) which precedes the SEL searching algorithm. The magnitude of the gradient is a measure of "edginess" at that point in the image. Therefore, nodes that exhibit a large gradient magnitude ought to be good candidates for root nodes. Questions to be asked are: 1) With what confidence do such nodes actually lie on an edge? 2) What is the probability that no root nodes for a given edge are generated in this way? and 3) How is a classification threshold on gradient magnitude values to be chosen?

These questions may be addressed using the Receiver Operating Characteristic (ROC) curves for gradient operators. ROC curves, common in the detection theory literature, are parametric plots of the probability of detection (PD) versus the probability of false alarm (PF) [Van68]. Their use with linear edge operators for images has been discussed by Abdou and Pratt [Abd79]. An example of such a curve is illustrated in Figure 2.2. This curve is for a simple 3 x 3 Sobel gradient-type edge operator and a signal-to-noise ratio of 10 dB. The precise definitions for these terms are given in [Abd 79]. The curve is parameterized by the decision threshold; as the threshold is raised, the locus of the operating point moves along the curve from the upper right hand corner to the lower left. The fact that the curve does not reach the upper left hand corner (PD = 1, PF = 0) is a manifestation of the imperfect performance of such operators: it is impossible to choose a threshold such that all the edge points are found and no non-edge points are misclassified as belonging to an edge.

The closeness with which an operator's ROC curve comes to the (ideal) upper left corner is an indication of that operator's performance. Large support gradient operators may be expected to do considerably better than the Sobel operator of Figure 2.2. Conversely, poorer SNR ratios tend to push the curve for any operator away from the upper left corner. It is a fundamental characteristic of all such curves, however, that they are convex $\cap$ and lie above the diagonal, PD = PF, known as the chance diagonal [Van68].

Returning to the problem of finding root nodes, we make the following important observation.

Observation: In contrast to the classical detection problem for which operation at a high probability of detection is desired, the problem of selecting root nodes demands operation very near the *lower left* corner of the ROC curve.

This observation is crucial to the starting node problem. If a high threshold on the gradient magnitude is employed (operation near the lower left corner of the ROC curve), then both the probability of detection and the probability of false alarm are small.

At any point along the ROC curve, the slope of the curve is equal to the likelihood ratio:

$$\mathit{slope} = l(f) = \frac{Pr\ (f \mid f \ \mathit{is\ on\ an\ edge}\ )}{Pr\ (f \mid f \ \mathit{is\ not\ on\ an\ edge}\ )} \qquad (2.48)$$

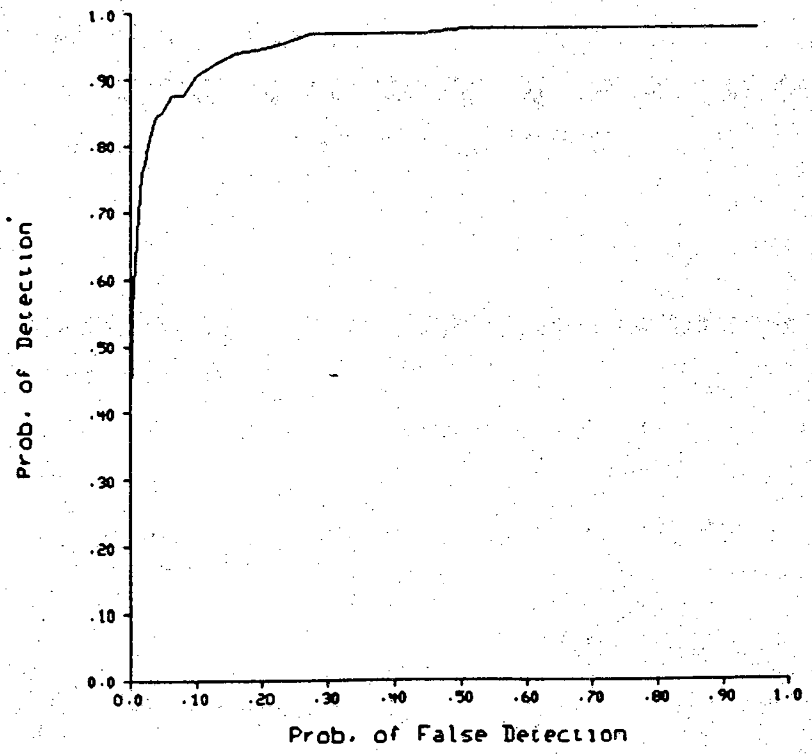

Figure 2.2: Receiver Operating Characteristic (ROC) curve for 3x3 Sobel edge operator on Pratt test image at SNR = 10 dB.

An important property of such curves is that this slope generally is quite large for large f, i.e. near the lower left corner. For example, if the conditional densities of Equation (2.48) are Gaussian, the likelihood ratio asymptotically approaches infinity. Thus, under quite general conditions it is reasonable to assume that the slope of the ROC curve is very high in the immediate vicinity of the origin. Imposing a high threshold on the gradient magnitude therefore implies that any magnitude value exceeding this threshold has a much higher probability of lying on an edge than not. The price paid for this high likelihood ratio is that the PD is small in the vicinity of the origin. That is, only a small fraction of the actual edge points will be identified by a high threshold. However, sequential searching techniques require only *one* starting node on an edge. A low PD is therefore not a drawback for SEL.

An example may serve to fix these ideas. Consider a gradient operator whose output in the presence of high noise power is characterized by the ROC curve of Figure 2.3. This ROC curve results from the conditional probabilities:

$$\begin{aligned} Pr\,(f \mid H_1) &= N\,(0.5,1) \\ Pr\,(f \mid H_0) &= N\,(-0.5,1) \end{aligned} \tag{2.49}$$

i.e. the conditional probabilities at the *output* of the operator are normal with a signal-to-noise ratio of 1.0. This corresponds to exceedingly poor quality in the original image. At a threshold value of:

$$f_t = 3.2 \tag{2.50}$$

the likelihood ratio is:

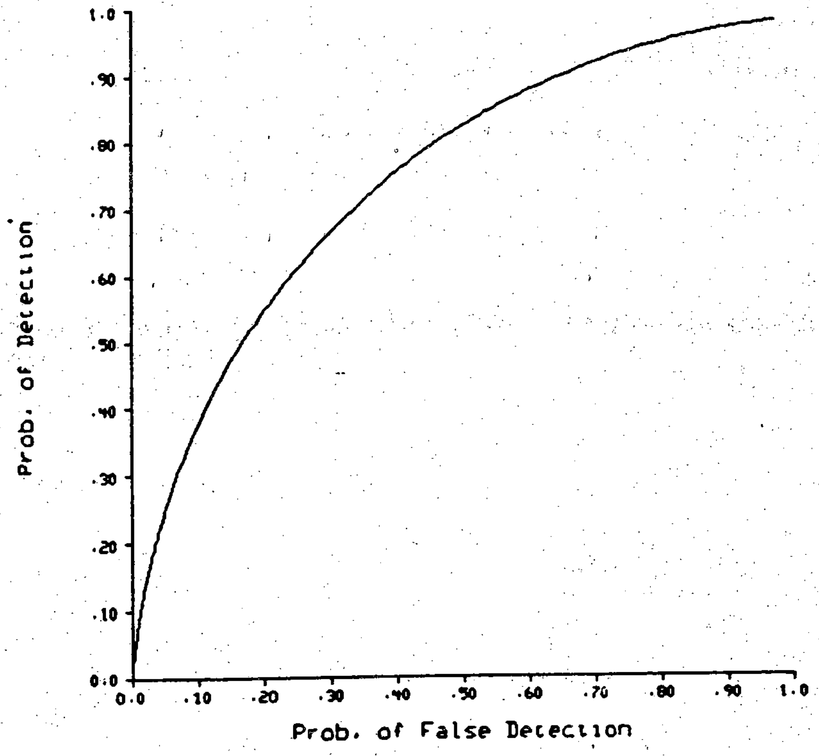

Figure 2.3: Receiver Operating Characteristic (ROC) curve where conditional density functions are both Normal.

$$\frac{Pr\ (f \mid H_1)}{Pr\ (f \mid H_0)} > 25 \tag{2.51}$$

and the PD is:

$$PD = erf\ (3.2 - 0.5) = 0.0035 \tag{2.52}$$

Thus, on average, only 1 in every 300 true edge points are identified by such a high threshold, but any point so identified is 25 times as likely to lie on an edge as not. The probability of false alarm is:

$$PF = erf\ (3.2 + 0.5) < 0.0001 \tag{2.53}$$

This example may be rather extreme; the output of large support Gaussian filtered gradient operators exhibit much higher SNR than this on many images of interest. However it serves to illustrate that, even under severe conditions, highly reliable start nodes may be generated by thresholding gradient magnitude values.

2.4.2. Using A-Priori Knowledge

The previous Section described a technique for finding candidate start nodes in very general settings. For certain applications, where a-priori knowledge concerning the image under consideration is available, this technique can be augmented or even replaced by scene-specific methods that make use of that information. One such method is described in [Mod76] for the case of angiocardiograms. Since the image is known to contain the boundary of a heart for which a prototype outline has been supplied, that technique searches a restricted portion of the image for candidate start nodes that best fit the prototype model.

Depending on how much a-priori information is available, such scene-specific methods may employ any number of pattern recognition or template correlation techniques to search for areas in the image that have a high probability of lying on an edge of interest. These techniques may in fact provide even better rejection of false edge points than that of 2.5.1. However, they are application specific. Their effectiveness is only as good as the scene model which they exploit. Where the image does not closely match the assumed model, their performance may degrade seriously. These techniques will not be treated in any depth here.

2.5. Search Termination

Strategies for terminating a sequential searching algorithm tend to be somewhat heuristic. Four such conditions suggest themselves for practical implementations. Others may perform well, especially where a-priori information is available.

The first three are obvious enough. Practical digitized images are of finite extent. Thus, intensity edges contained therein are of finite length. Several situations are possible. One, an edge may close on itself (e.g. a circle or the bounding contour of an object wholly contained in the image). In this case, the search may be terminated when the current best path closes on itself. Second, an edge may intersect another. Here again, search is terminated when the best path intersects another, previously found, edge path. Third, an edge may be continued off the support of the image. For this case, termination occurs when the search reaches the image boundary.

Another case is possible. Here an edge either stops abruptly without intersecting another or gradually fades in contrast to zero. These are the only really difficult situations for a sequential algorithm to handle. The approach suggested here is based on the running path metric. Since the algorithm will continue to search beyond the end of an abrupt edge, all such paths will exhibit a sharp fall off in metric beyond the end of the edge. Likewise, a strong intensity edge that begins to fade in contrast will be tracked for awhile by the algorithm, but the resulting path metric will fall from its previous high value before the fade. It is therefore reasonable to suggest a termination condition based on running metric. When the metric of the best path falls below some specified fraction of the highest metric along that path, the search is terminated. Of course, one runs the risk of terminating search just before the metric picks up again unless the fraction is fairly small. Abrupt edge ends are likely to be handled well by this technique, but slowly fading edges must always present a compromise to a sequential algorithm (and indeed to any other algorithm for that matter).

CHAPTER 3

BOUNDS ON SEQUENTIAL SEARCHING IN 2-D FIELDS

3.1. Introduction

A very important attribute of Sequential Edge Linking as a result of its log-likelihood metric is the fact that some attributes of its operation and performance may be quantified analytically. That is, inferences can be made and bounds imposed on operational parameters such as search time or dynamic stack behavior as a function of image quality. Also, performance questions such as probability of error can be formulated and addressed by analytical means.

This is in sharp contrast with the sequential techniques of Ashkar et. al. [Ash78] and Cooper [Coo79] or the various relaxation approaches such as [Zuc76]. With none of these techniques is it possible to predict the behavior of the algorithm or the probability of error on the detected edges a-priori from knowledge of parameters of the random field. Indeed, in the case of relaxation, convergence in the output is not even guaranteed except under certain sets of conditions [Zuc78a]. With these techniques, investigators have been forced to benchmark their performance with standardized performance measures such as the Pratt figure of merit [Pra78] or the Kitchen and Rosenfeld figure of merit [Kit81] as a means of comparison. On the other hand, the analysis developed in this Chapter describes a relationship between the workings of the SEL algorithm and the image random field model.

This Chapter is divided into two principle topics, 3.2: Distribution of Computation, and 3.3: Bound on the Probability of Error. A third Section, 3.4: Comparisons with the Coding Problem, is added to emphasize and clarify the similarities and differences between the edge linking and coding problems. The Distribution of Computation and Probability of Error questions are very much interrelated from an analytical standpoint, but are directed toward different phenomena. The first deals with the dynamic search behavior of the SEL algorithm whereas the second deals with the resulting edge paths.

3.2. Distribution of Computation

3.2.1. Introduction

In Section 2.2.3, we showed that the average behavior of the path metric is to increase along true edge paths (assuming Equation (2.29) is satisfied) and to decrease otherwise. This alone does not guarantee that the algorithm always follows a true edge since noise in the image can combine to fool the algorithm into making some wrong choices. Various questions therefore suggest themselves: How much time does the algorithm spend investigating wrong paths? Does it always return to the correct path? What is the relationship between image noise and the searching behavior of the algorithm? These and other questions will be formulated and answered in this Section.

Let us use the stack algorithm to illustrate the problem before us and the nature of the solution we seek. Suppose the algorithm has successfully followed an edge path up to node n. Beyond node n, however, the metric values along the true edge path decrease for several nodes before resuming its upward trend. Because of this dip in metric, the algorithm is forced to explore some of the incorrect paths emanating from node n, since their metrics place them higher in the stack than the correct path in the midst of the dip. When followed far enough, however, all of these incorrect paths eventually exhibit metrics below that of the correct path at the dip. Since the correct path is then at the top of the stack, the algorithm resumes its tracing of that path.

From this simple scenario, we see that because of the noise-induced dip in the correct path metric, the algorithm is forced to waste computation time on the exploration of incorrect paths which are eventually discarded. The amount of computation required to extend the correct path by a node is therefore not fixed but variable. We presume that as the noise in the image increases, this computational burden becomes more severe. What we seek is an analytical relationship between the image "noise" and the amount of computation required to properly follow the correct path. We would like to be able to determine the distribution of computation of the algorithm; i.e. the distribution of the random variable C_n, where C_n is the number of algorithm iterations (or computations) required to successfully extend the correct path from node n to its successor. Such would be a complete description of the dynamical behavior of the algorithm from a statistical point of view.

In fact, we are not able to analytically determine the distribution of the variable C_n, but we are able to bound this distribution. This bound, while not particularly tight (especially for small values of C_n), is nevertheless extremely useful.

3.2.2. Definitions

In this Section we introduce some definitions and notation to be used in the following Sections. In Section 2.2, we have already defined what we mean by paths, path metrics, nodes, successor and predecessor nodes, and random field observations or f-values. Suppose we are given a random field with an edge path in it as defined by Equations (2.3) and (2.11), and further that we are given an arbitrary root node and direction on that path. We will denote by $\mathbf{f}_n$ a sequence of f-values along the correct edge path of length n. We will denote by $\mathbf{f}'_m$ a sequence of f-values associated with any incorrect path of length m.

Now consider the t^{th} node along the correct path. Emanating from this node is of course the rest of the correct path as well as an entire sub-tree of incorrect paths. If we consider the random field to be of infinite extent in the plane, then this sub-tree contains an infinite number of incorrect paths. This sub-tree of incorrect paths emanating from node t of the correct path will be denoted S_t. S_t contains paths that remerge with the correct path beyond node t. Because remerging paths are a nuisance in various bounding arguments, let us consider a hypothetical incorrect path subtree, denoted $\overline{S}_t$, in which remergings do not occur. That is, every node in $\overline{S}_t$ has three *unique* successor nodes. Every unique node in the physical subtree S_t has a corresponding node in $\overline{S}_t$, but in addition, $\overline{S}_t$ contains *extra* nodes where paths remerge in S_t. Note that such a hypothetical tree cannot be defined on a rectangular array of nodes. The situation of paths that remerge with the correct path will be treated in Section 3.3.

We will define a computation as the effort required by the algorithm to extend any path by a node. C_t denotes the number of paths (and therefore the number of computations) ever extended by the algorithm in the incorrect subtree S_t. Likewise, $\overline{C}_t$ denotes the number of paths extended in $\overline{S}_t$. Since every unique node in S_t has a corresponding node in $\overline{S}_t$ but not vice versa, we have the inequality: $\overline{C}_t \geq C_t$.

Finally, we define Δ_t to be the value of the largest dip in path metric along the correct path after node t:

$$\Delta_t \;=\; \Gamma(\mathbf{f}_t) - \min_{s \geq t} \Gamma(\mathbf{f}_s) \qquad \geq 0 \tag{3.1}$$

This quantity Δ_t is fundamental to all sequential tree searching problems since it largely determines how much searching of incorrect paths is necessary at node t. In Equation (3.1), we have shortened the notation for the path metric $\Gamma\,(\mathbf{m},\mathbf{f})$ to $\Gamma\,(\mathbf{f})$ for the sake of clarity in what follows. Bear in mind that Γ is a function of the path letters, $\mathbf{m}$, as well as the random field values, $\mathbf{f}$.

3.2.3. Bounding C_t

We require some preliminary results before establishing our bound on C_t. Following the example of sequential tree searching in coding theory, we begin with a lemma from Forney [For74b]:

Lemma 3.1: Let $\mathbf{f}^a$ and $\mathbf{f}^b$ be any two paths emanating from a common node, and let the minimum metric on the path $\mathbf{f}^a$ after the common node exceed $\Gamma\,(\mathbf{f}^b)$. Then the Z-J algorithm cannot extend $\mathbf{f}^b$ before extending $\mathbf{f}^a$.

The proof is in [For74b] allowing for the fact that the Z-J algorithm does not delete the inferior path when two paths merge. This lemma is used to immediately prove Theorem

3.1, also from [For74b].

Theorem 3.1: The number of computations, C_t, is overbounded by the number of paths $\mathbf{f}'$ in the incorrect subtree, S_t, with metrics not less than the minimum metric on the correct path at node t or later:

$$C_t \leq \sum_{\mathbf{f}' \in S_t} \phi(\mathbf{f}') \tag{3.2}$$

where

$$\phi(\mathbf{f}') = \left\{ \begin{array}{ll} 1 & ,if \quad \Gamma(\mathbf{f}') \geq \Gamma(\mathbf{f}_t) - \Delta_t \\ 0 & ,otherwise \end{array} \right\} \tag{3.3}$$

Proof: If $\mathbf{f}_s$, $s \geq t$ is not on the stack, then $C_t = 0$ since node t has not been reached. Otherwise, by Lemma 1, if $\Gamma(\mathbf{f}') < \Gamma(\mathbf{f}_t) - \Delta_t$, then $\mathbf{f}'$ cannot be extended.

Since it is easier to not have to worry about remerging paths and since $C_t \leq \overline{C}_t$, we will actually work with $\overline{C}_t$ and $\overline{S}_t$. Also, it will later be shown that taking $t = 0$ results in no loss of generality while facilitating certain limit arguments.

We are now in a position to state and prove the main theorem of this Section.

Theorem 3.2: For any $\alpha \geq 0$ and any $0 \leq \rho \leq 1$, if the distributions $p_1(\cdot)$ and $p_0(\cdot)$ are such that:

$$\begin{aligned} \sum_f p_0(f)^{1-\alpha\rho} p_1(f)^{\alpha\rho} &< 3^{-\rho} \\ \sum_f p_0(f)^{\alpha\rho} p_1(f)^{1-\alpha\rho} &< 3^{-\alpha\rho} \end{aligned} \tag{3.4}$$

then:

$$Pr\ (\overline{C}_0 \geq N) \quad \leq \quad K\ N^{-\rho} \qquad ;K \textit{ a constant.} \tag{3.5}$$

Proof:

As in [For74], we consider the following quantity:

$$T\ (\alpha,\rho) \quad = \quad \left[\sum_{\mathbf{f}' \ \in \bar{S}_0} e^{\alpha \Gamma(\mathbf{f}')} \right]^{\rho} \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.6}$$

The reason for defining $T\ (\alpha,\rho)$ as above is rooted in the Chernoff bound, discussed by Gallager [Gal68]. Briefly, we wish to obtain a bound involving the quantity $\Gamma\,(\mathbf{f}'\ )$. The metric was seen in Equation (2.18) to be the sum of independent random variables. A common and useful bound on the probability that a sum of independent random variables exceeds a given number may be obtained from various forms of generalized Chebyshev Inequalities. Since the sum in $\Gamma\,(\mathbf{f}'\ )$ is of natural logs of random variables, the generalized Chebyshev Inequality that is most useful, usually called the Chernoff bound, involves the substitution $w\ = e^{\Gamma(\mathbf{f}'\ )}$.

Returning to Equation (3.6), using Theorem 3.1 we have:

$$\begin{aligned} T\ (\alpha,\rho) \quad &= \quad \left[\sum_{\mathbf{f}' \ \in \bar{S}_0} e^{\alpha \Gamma(\mathbf{f}')} \right]^{\rho} \\ &= \quad e^{-\alpha\rho\Delta_0} \left[\sum_{\mathbf{f}' \ \in \bar{S}_0} e^{\alpha \Gamma(\mathbf{f}'\ ) + \alpha \Delta_0} \right]^{\rho} \\ &\geq \quad e^{-\alpha\rho\Delta_0} \left[\overline{C}_0 \right]^{\rho} \end{aligned} \tag{3.7}$$

where the last step results from Theorem 3.1. This is because:

$$\phi(\mathbf{f}') \leq e^{\alpha\Gamma(\mathbf{f}') + \alpha\Delta_0} \tag{3.8}$$

which can easily be seen from the fact that:

$$\begin{aligned}
\phi(\mathbf{f}') &= 1 \leq e^{\alpha\Gamma(\mathbf{f}') + \alpha\Delta_0} \qquad ,\textit{if} \quad \Gamma(\mathbf{f}') \geq \Gamma(\mathbf{f}_0) - \Delta_0 = -\Delta_0 \\
\phi(\mathbf{f}') &= 0 \leq e^{\alpha\Gamma(\mathbf{f}') + \alpha\Delta_0} \qquad ,\textit{otherwise.}
\end{aligned}$$

Therefore:

$$T(\alpha,\rho)\, e^{\alpha\rho\Delta_0} \geq \left[\overline{C}_0\right]^\rho \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.9}$$

Now, from the Chebyshev Inequality we obtain:

$$\begin{aligned}
Pr\,(\overline{C}_0 \geq N) &= Pr\,(\overline{C}_0^\rho \geq N^\rho) \\
&\leq Pr\,(T(\alpha,\rho)\, e^{\alpha\rho\Delta_0} \geq N^\rho) \\
&\leq N^{-\rho}\, \overline{T(\alpha,\rho)\, e^{\alpha\rho\Delta_0}}
\end{aligned} \tag{3.10}$$

where the overbar in the last expression denotes expected value. This expectation is over four distributions: all choices of state sequences of the correct path, $\mathbf{m}$, state sequences of incorrect paths, $\mathbf{m}'$, observations along the correct path, $\mathbf{f}$, and observations along the incorrect paths, $\mathbf{f}'$. This is in counterdistinction to the decoding problem where the equivalent expectation is over an ensemble of codes, the correct path, and the channel transitions, and so the two problems diverge from this point. This fundamental difference is explored more fully in Section 3.4. We may loosely bound the term:

$$\begin{aligned}
e^{\alpha\rho\Delta_0} &= e^{-\alpha\rho\left[\min_{s \geq 0} \Gamma(\mathbf{f}_s)\right]} \\
&\leq \sum_{s=0}^{\infty} e^{-\alpha\rho\Gamma(\mathbf{f}_s)}
\end{aligned} \tag{3.11}$$

Combining Equations (3.10) and (3.11), we obtain our bound:

$$Pr\ (\overline{C}_0 \geq N) \ \leq \ N^{-\rho} \sum_{s=0}^{\infty} \overline{e^{-\alpha\rho\Gamma(\mathbf{f}_s)} \left[\sum_{\mathbf{f}' \in \overline{S}_0} e^{\alpha\Gamma(\mathbf{f}')} \right]^{\rho}} \tag{3.12}$$

In the above Equation, the inner sum is over the set of incorrect paths $\mathbf{f}'_m$ in the (infinite) incorrect subtree $\overline{S}_0$. This set can be partitioned in the following manner:

$$\overline{S}_0 \ = \ \bigcup_{r=0}^{\infty} F'_r \tag{3.13}$$

where:

$$F'_r \ = \ \left\{ \mathbf{f}'_m \in \overline{S}_0 \ | \ m = r \right\} \tag{3.14}$$

Thus, Equation (3.12) becomes:

$$Pr\ (\overline{C}_0 \ \geq \ N) \ \leq \ N^{-\rho} \sum_{s=0}^{\infty} \overline{e^{-\alpha\rho\Gamma(\mathbf{f}_s)} \left[\sum_{r=0}^{\infty} \ \sum_{\mathbf{f}_r' \in F_r'} e^{\alpha\Gamma(\mathbf{f}_r')} \right]^{\rho}} \tag{3.15}$$

Jensen's Inequality holds that for $a_i \geq 0$,

$$E \left\{ \left[\sum_i a_i \right]^{\rho} \right\} \ \leq \ E \left\{ \sum_i a_i{}^{\rho} \right\} \qquad\qquad 0 \leq \rho \leq 1 \tag{3.16}$$

so that:

$$Pr\ (\overline{C}_0 \ \geq \ N) \ \leq \ N^{-\rho} \sum_{s=0}^{\infty} \sum_{r=0}^{\infty} \overline{\left[\sum_{\mathbf{f}_r' \in F_r'} e^{\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)} \right]^{\rho}} \tag{3.17}$$

We will denote the summand as $P_{r,s}(\alpha,\rho)$, so:

$$Pr\,(\overline{C}_0 \geq N) \;\leq\; N^{-\rho} \sum_{r=0}^{\infty} \sum_{s=0}^{\infty} P_{r,s}(\alpha,\rho) \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.18}$$

Up to this point, the only major difference between our problem and the approach in the coding literature is the nature of distributions over which the expectation of Equation (3.10) has been taken. The next step in the analysis of convolutional decoding is to bound (the equivalent to) $P_{r,s}(\alpha,\rho)$ using a random coding argument of Gallager [Gal65]. This method is not applicable in the problem under consideration, but we may proceed in a more direct fashion. Consider the quantity,

$$P_{r,s}(\alpha,\rho) \;=\; \overline{\left[\sum_{\mathbf{f}_r' \,\in F_r'} e^{\,\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)} \right]^{\rho}} \tag{3.19}$$

We reiterate that the indicated expectation is with respect to all possible $\mathbf{m}_s$, $\mathbf{m}'_r$, $\mathbf{f}_s$, and $\mathbf{f}_r'$. Explicitly performing the expectation with respect to the last two random vectors,

$$P_{r,s}(\alpha,\rho) \;=\; \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r') \;\overline{\left[\sum_{\mathbf{f}_r' \,\in F_r'} e^{\,\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)} \right]^{\rho}} \tag{3.20}$$

where the $\bar{\cdot}$ indicates an expectation with respect to $\mathbf{m}$ and $\mathbf{m}'$. Again from Jensen's Inequality, for $a_i \geq 0$ we have:

$$\sum p_i\; a_i^{\,\rho} \;\leq\; \left[\sum p_i\; a_i \right]^{\rho} \qquad 0 \leq \rho \leq 1 \tag{3.21}$$

or

$$E\left\{ a_i^{\,\rho} \right\} \;\leq\; \left[E\left\{ a_i \right\} \right]^{\rho}, \tag{3.22}$$

so:

$$
\begin{aligned}
P_{r,s}(\alpha,\rho) &\le \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r') \left[\overline{\sum_{\mathbf{f}_r' \in F_r'} e^{\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)}}\right]^{\rho} \\
&= \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r') \left[\sum_{\mathbf{f}_r' \in F_r'} \overline{e^{\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)}}\right]^{\rho} \qquad (3.23) \\
&= |F_r'|^{\rho} \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r') \left[\overline{e^{\alpha\Gamma(\mathbf{f}_r') - \alpha\Gamma(\mathbf{f}_s)}}\right]^{\rho}
\end{aligned}
$$

where $|F_r'|$ is the modulus of the set F_r'. Considering the bracketed term in (3.23), from Equation (2.18), a path metric consists of a sum of terms:

$$
\begin{aligned}
\Gamma(\mathbf{f}_s) &= \sum_{i=0}^{s} \left[\ln \frac{p_1(f_i)}{p_0(f_i)} + \ln P(s_i \mid s_{i-1})\right] \\
&= \sum_{i=0}^{s} \left[a_i + b_i\right] \qquad (3.24)
\end{aligned}
$$

$$
\begin{aligned}
a_i &= \ln \frac{p_1(f_i)}{p_0(f_i)} \qquad (3.25) \\
b_i &= \ln P(s_i \mid s_{i-1})
\end{aligned}
$$

Using this notation, Equation (3.23) becomes:

$$
\begin{aligned}
P_{r,s}(\alpha,\rho) &\le |F_r'|^{\rho} \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r') \left[\overline{e^{\alpha\sum_{i=0}^{r}(a_i' + b_i') - \alpha\sum_{j=0}^{s}(a_j + b_j)}}\right]^{\rho} \\
&\qquad (3.26) \\
&= \left[\overline{e^{\alpha\sum_{i=0}^{r} b_i' - \alpha\sum_{j=0}^{s} b_j}}\right]^{\rho} |F_r'|^{\rho} \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_r'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r')\, e^{\alpha\rho\sum_{i=0}^{r} a_i' - \alpha\rho\sum_{j=0}^{s} a_j}
\end{aligned}
$$

We first concentrate on the exponential term of the double sum. At this point we use the crucial assumption that the f-values, conditioned on either the H_1 or H_0

hypothesis, are stochastically independent and that f -values from different paths are independent. It is well known that the average of a product of independent, identically distributed random variables is equal to the product of the average of any one of them. We thus have the identity:

$$\begin{aligned}
&\sum_{\mathbf{f}_s,\, \mathbf{f}_r'}\sum p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r')\, e^{\alpha\rho\sum_{i=0}^{r} a_i' - \alpha\rho\sum_{j=0}^{s} a_j} \\
&= \sum_{\mathbf{f}_r'} p_0(\mathbf{f}_r')\, e^{\alpha\rho\sum_{i=0}^{r} a_i'} \sum_{\mathbf{f}_s} p_1(\mathbf{f}_s)\, e^{-\alpha\rho\sum_{j=0}^{s} a_j} \\
&= \prod_{i=0}^{r} p_0(f_i')\, e^{\alpha\rho a_i'} \prod_{j=0}^{s} p_1(f_j)\, e^{\alpha\rho a_j} \\
&= \left[\sum_{f'} p_0(f')\, e^{\alpha\rho a'}\right]^r \left[\sum_{f} p_1(f)\, e^{-\alpha\rho a}\right]^s
\end{aligned} \tag{3.27}$$

where the first equality is due to the independence of the paths, the second is due to the conditional independence of f -values along each path, and the last is due to the fact that the conditional f -values have identical distributions. Substituting $a = \ln \dfrac{p_1(f)}{p_0(f)}$,

$$\begin{aligned}
&\sum_{\mathbf{f}_s,\, \mathbf{f}_r'}\sum p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r')\, e^{\alpha\rho\sum_{i=0}^{r} a_i' - \alpha\rho\sum_{j=0}^{s} a_j} \\
&= \left[\sum_{f'} p_0(f') \left[\frac{p_1(f')}{p_0(f')}\right]^{\alpha\rho}\right]^r \left[\sum_{f} p_1(f) \left[\frac{p_1(f)}{p_0(f)}\right]^{-\alpha\rho}\right]^s \\
&= \left[\sum_{f'} \left[p_0(f')\right]^{1-\alpha\rho} \left[p_1(f')\right]^{\alpha\rho}\right]^r \left[\sum_{f} \left[p_1(f)\right]^{1-\alpha\rho} \left[p_0(f)\right]^{\alpha\rho}\right]^s
\end{aligned} \tag{3.28}$$

Because of the importance of this quantity, we shall define:

$$D(\delta) \; = \; \sum_{f} p_1(f)^{1-\delta} \, p_0(f)^{\delta} \tag{3.29}$$

Therefore:

$$\sum_{\mathbf{f}_s,\,\mathbf{f}_r'}\sum p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_r')\, e^{\alpha\rho\sum_{i=0}^{r} a_i' \,-\, \alpha\rho\sum_{j=0}^{s} a_j} \; = \; [D(1-\alpha\rho)]^r \, [D(\alpha\rho)]^s \tag{3.30}$$

Returning to Equation (3.26), the bracketed term on the right hand side is:

$$\begin{aligned}\left[\,\overline{e^{\alpha\sum_{i=0}^{r} b_i' \,-\, \alpha\sum_{j=0}^{s} b_j}}\,\right]^{\rho} &= \\ \left[\sum_{\mathbf{m}_r'} p(\mathbf{m}_r')\left[p(\mathbf{m}_r')\right]^{\alpha}\right]^{\rho} & \left[\sum_{\mathbf{m}_s} p(\mathbf{m}_s)\left[p(\mathbf{m}_s)\right]^{-\alpha}\right]^{\rho}\end{aligned} \tag{3.31}$$

The first term of (3.31) satisfies the inequality (for $\alpha \geq 0$) :

$$\begin{aligned}\left[\sum_{\mathbf{m}_r'} p(\mathbf{m}_r')\left[p(\mathbf{m}_r')\right]^{\alpha}\right]^{\rho} &\leq \left[\sum_{\mathbf{m}_r'} p(\mathbf{m}_r')\,[\,1\,]^{\alpha}\right]^{\rho} \\ &= \left[\sum_{\mathbf{m}_r'} p(\mathbf{m}_r')\right]^{\rho} \\ &= 1^{\rho} \; = \; 1\end{aligned} \tag{3.32}$$

For the second term, since the function $f(x) = x^{\alpha}$; $0 \leq \alpha \leq 1$; $x \geq 0$ is convex $\cap$, we may use Jensen's Inequality:

$$\left[\sum_{\mathbf{m}_s} p(\mathbf{m}_s) \left[\frac{1}{p(\mathbf{m}_s)}\right]^{\alpha}\right]^{\rho} \le \left[\sum_{\mathbf{m}_s} p(\mathbf{m}_s) \frac{1}{p(\mathbf{m}_s)}\right]^{\alpha\rho} = \left[\sum_{\mathbf{m}_s} 1\right]^{\alpha\rho} = \left[3^s\right]^{\alpha\rho} \qquad (3.33)$$

where the last line results from the fact that there are 3^s paths of length s in $\overline{S}$ (Section 2.2.2). This same path structure overbounds $| F_r' |$:

$$| F_r' | \le 3^r \qquad (3.34)$$

Putting together Equations (3.30), (3.31-3.33), and (3.34), Equation (3.26) has the final form:

$$\begin{aligned} P_{r,s}(\alpha,\rho) &\le 1 \cdot 3^{s\alpha\rho} \cdot 3^{r\rho} \cdot [D(1-\alpha\rho)]^r \, [D(\alpha\rho)]^s \\ &= [3^{\rho} \cdot D(1-\alpha\rho)]^r \, [3^{\alpha\rho} \cdot D(\alpha\rho)]^s \qquad \begin{matrix} \alpha \ge 0 \\ 0 \le \rho \le 1 \end{matrix} \end{aligned} \qquad (3.35)$$

Finally, substituting this into Equation (3.18), our bound is:

$$Pr\,(\overline{C}_0 \ge N) \le N^{-\rho} \sum_{r=0}^{\infty} \sum_{s=0}^{\infty} \beta_1^r \beta_2^s \qquad \begin{matrix} \alpha \ge 0 \\ 0 \le \rho \le 1 \end{matrix} \qquad (3.36)$$

where:

$$\begin{aligned} \beta_1 &= 3^{\rho} \cdot D(1-\alpha\rho) \\ \beta_0 &= 3^{\alpha\rho} \cdot D(\alpha\rho) \end{aligned} \qquad (3.37)$$

We see from this that if the two conditions, $\beta_1 < 1$ and $\beta_0 < 1$, are satisfied (which are the hypotheses of the theorem), then:

$$Pr\ (\overline{C}_0 \geq N) \ \leq \ K\ N^{-\rho} \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.38}$$

$$K \ = \ \frac{1}{1-\beta_1} \cdot \frac{1}{1-\beta_2}$$

and Theorem 2 is proved.

Using the fact that $C_0 \leq \overline{C}_0$, we have immediately:

Corollary 3.1: For any $\alpha \geq 0$ and any $0 \leq \rho \leq 1$, if $D(1-\alpha\rho) < 3^{-\rho}$ and $D(\alpha\rho) < 3^{-\alpha\rho}$, then:

$$Pr\ (C_0 \geq N) \ \leq \ K\ N^{-\rho} \qquad ;K \textit{ as in } (3.38). \tag{3.39}$$

Sequential Edge Linking, in common with all other sequential tree searching algorithms, uses a path metric that is additive (Equation (2.19)). All metrics of paths emanating from a node, t, along the correct path can thus be decomposed as:

$$\Gamma(\mathbf{f}_n) \ = \ \Gamma(\mathbf{f}_t) \ + \ \Gamma(\mathbf{f}_{n-t}) \tag{3.40}$$

Since the component $\Gamma(\mathbf{f}_t)$ is common to all paths emanating from node t, the searching dynamics depend only on those metric components beyond t, $\Gamma(\mathbf{f}_{n-t})$. Thus, the distribution of computation is identical at any node t along the correct path to that at $t = 0$, which leads us directly to the following Corollary.

Corollary 3.2: For any $\alpha \geq 0$ and any $0 \leq \rho \leq 1$, if $D(1-\alpha\rho) < 3^{-\rho}$ and $D(\alpha\rho) < 3^{-\alpha\rho}$, then:

$$Pr\ (C_t \geq N)\ \leq\ K\ N^{-\rho} \qquad ;K \text{ as in } (3.38). \tag{3.41}$$

We have made a great deal of progress so far. While we cannot find an analytical expression for the precise distribution of computation, we have obtained a bound on the distribution in Equation (3.41). This Equation provides an important qualitative link between the random field model of the image and the amount of incorrect path searching performed by the algorithm. As the signal-to-noise ratio degrades or improves, it is reflected in the conditional densities p_1 and p_0 which overlap more or less, respectively. This has a direct bearing on the constant K in Equation (3.41) via Equations (3.37). The higher the signal-to-noise ratio, the smaller the constant K and the lower the bound on the amount of searching per node. In the next Section, we extend these results by examining a bound on the first moment of the distribution.

3.2.4. The Expectation of C_t and the Pareto Exponent

In the previous Section, we showed that under suitable conditions, the number of computations performed at any node along the correct path is bounded by Equation (3.41) for $0 \leq \rho \leq 1$. The right hand side is known as a Pareto distribution [Lin83], and ρ the Pareto exponent. Let us introduce a continuous random variable Y with this distribution. Then the cumulative distribution function of C_t is lower bounded by that of Y:

$$\begin{aligned} F_{C_t}(n) &= Pr(C_t \leq n) \geq 1 - Pr(C_t \geq n) \\ &= 1 - Kn^{-\rho} \\ &= F_Y(n) \end{aligned} \tag{3.42}$$

From this it can be concluded that

$$E\left\{C_t\right\} \le E\left\{Y\right\} \tag{3.43}$$

The mean of the variable Y is easily computed. The probability density function of Y is:

$$p_Y(y) = \frac{d}{dy} F_Y(y) = \rho K\, y^{-\rho-1} \tag{3.44}$$

Therefore:

$$\begin{aligned} E\left\{Y\right\} &= \int_1^\infty y\, p_Y(y)dy \\ &= \int_1^\infty \rho K\, y^{-\rho} dy \\ &= \lim_{y\to\infty} \frac{\rho K}{1-\rho}\,(y^{1-\rho}-1) \end{aligned} \tag{3.45}$$

From this we have the following theorem.

Theorem 3.3: If the distribution of computation is bounded by a Pareto distribution with exponent $\rho > 1$,

$$Pr\,(C_t \ge N) \le K\,N^{-\rho} \qquad \rho > 1 \tag{3.46}$$

then the mean number of computations per node along the correct path is finite.

Proof: The proof follows directly from the discussion above. If $\rho > 1$, then

$$E\left\{C_t\right\} \le E\left\{Y\right\} = \lim_{y\to\infty} \frac{\rho K}{1-\rho}\left(y^{1-\rho}-1\right) = \frac{\rho K}{\rho-1} < \infty \tag{3.47}$$

In fact, as is shown by [Lin83], if $\rho > i$, then the i^{th} moment of Y is bounded. Although straightforward, this theorem is very important. It guarantees that the expected value of decoding effort is finite along the correct path if the bound on C_t has an exponent $\rho > 1$. What is missing here is a condition that will ensure $\rho > 1$. The next theorem will supply that condition.

The following theorem bounds the distribution of C_t for a Pareto exponent $1 \le \rho \le 2$. It is clear from the proof how this bound may be extended for $\rho > 2$. The proof itself is somewhat abbreviated since it closely follows that of Theorem 3.2.

Theorem 3.4: For any $\alpha \ge 0$ and any $1 \le \rho \le 2$, if $D(\alpha\rho) < 3^{-\alpha\rho}$

$$D(1-\alpha\rho) < 3^{-\frac{\rho}{2}} \quad \text{and} \quad D(1-\frac{\alpha\rho}{2}) < 3^{-\frac{\rho}{2}}, \text{then:}$$

$$Pr\ (\overline{C}_0 \ge N) \ \le\ K\ N^{-\rho} \qquad ;K\ a\ constant. \tag{3.48}$$

Proof:

We consider *pairs* of incorrect paths in $\overline{S}_0$, $(\mathbf{f}'^{\,1}, \mathbf{f}'^{\,2})$, and define $T\ (\alpha,\rho)$ as:

$$T(\alpha,\rho) = \left[\sum_{\mathbf{f}'^1 \in \bar{S}_0} \sum_{\mathbf{f}'^2 \in \bar{S}_0} e^{\alpha \Gamma(\mathbf{f}'^1)} e^{\alpha \Gamma(\mathbf{f}'^2)} \right]^{\frac{\rho}{2}} \qquad \begin{array}{c} \alpha \geq 0 \\ 1 \leq \rho \leq 2 \end{array}$$

$$= e^{-\alpha\rho\Delta_0} \left[\sum_{\mathbf{f}'^1 \in \bar{S}_0} \sum_{\mathbf{f}'^2 \in \bar{S}_0} e^{\alpha\Gamma(\mathbf{f}'^1) + \alpha\Delta_0} e^{\alpha\Gamma(\mathbf{f}'^2) + \alpha\Delta_0} \right]^{\frac{\rho}{2}} \quad \textbf{(3.49)}$$

$$\geq e^{-\alpha\rho\Delta_0} \left[\bar{C}_0^2 \right]^{\frac{\rho}{2}}$$

where the last step is again a result of:

$$\sum_{\mathbf{f}'^1 \in \bar{S}_0} \sum_{\mathbf{f}'^2 \in \bar{S}_0} e^{\alpha\Gamma(\mathbf{f}'^1) + \alpha\Delta_0} e^{\alpha\Gamma(\mathbf{f}'^2) + \alpha\Delta_0}$$
$$\geq \sum_{\mathbf{f}'^1 \in \bar{S}_0} \sum_{\mathbf{f}'^2 \in \bar{S}_0} \phi(\mathbf{f}'^1)\,\phi(\mathbf{f}'^2) \quad (3.50)$$
$$\geq \bar{C}_0^2$$

Therefore:

$$T(\alpha,\rho)\, e^{\alpha\rho\Delta_0} \geq \left[\bar{C}_0^2\right]^{\frac{\rho}{2}} \qquad \begin{array}{c} \alpha \geq 0 \\ 1 \leq \rho \leq 2 \end{array} \quad (3.51)$$

So, as before, we employ the Chebyshev Inequality to obtain our bound:

$$Pr\; \bar{C}_0 \geq N) \leq Pr\,(T(\alpha,\rho)\, e^{\alpha\rho\Delta_0} \geq N^{\rho})$$
$$\leq N^{-\rho}\, \overline{T(\alpha,\rho)\, e^{\alpha\rho\Delta_0}} \quad (3.52)$$
$$\leq N^{-\rho} \overline{\sum_{s=0}^{\infty} e^{-\alpha\rho\Gamma(\mathbf{f}_s)} \left[\sum_{\mathbf{f}'^1 \in \bar{S}_0} \sum_{\mathbf{f}'^2 \in \bar{S}_0} e^{\alpha\Gamma(\mathbf{f}'^1)} e^{\alpha\Gamma(\mathbf{f}'^2)} \right]^{\frac{\rho}{2}}}$$

where, in the last equation we have used (3.31). This time, it is somewhat more difficult to partition the set of all pairs of paths $(\mathbf{f}'^1, \mathbf{f}'^2) \in \bar{S}_0$. This is because it is possible for $\mathbf{f}'^{\,1}$ and $\mathbf{f}'^{\,2}$ to be merged for awhile before separating. Fortunately, most of the large number of path topologies or configurations possible in tree structures in the general case are not possible in $\overline{S}_0$ where re-merging does not take place [For76a]. Thus path topologies such as Figure 3.1 or 3.2 do not exist. In fact, Figure 3.3 represents the only topology for pairs of paths in $\overline{S}_0$. Therefore, if we denote:

$$(F_p' \times F_q')_r \;=\; \left\{ (\mathbf{f}_p' , \mathbf{f}_q') \mid \mathbf{f}_p' \in \overline{S}_0, \mathbf{f}_q' \in \overline{S}_0, p \geq r, q \geq r, f_{p_i} = f_{q_i} \; 0 \leq i \leq r \right\} \quad (3.53)$$

i.e. this is the set of all pairs of paths in $\overline{S}_0$ such that the two paths are merged up to node r and are distinct from there to nodes p and q. With this notation, all pairs of paths in $\overline{S}_0$ may be partitioned as:

$$\left\{ (\mathbf{f}'^{\,1}, \mathbf{f}'^{\,2}) \mid \mathbf{f}'^{\,1} \in \overline{S}_0, \mathbf{f}'^{\,2} \in \overline{S}_0 \right\} = \bigcup_{r=0}^{\infty} \left[\bigcup_{p=r}^{\infty} \bigcup_{q=r}^{\infty} (F_p' \times F_q')_r \right] \quad (3.54)$$

and Equation (3.52) becomes:

$$Pr\,(\overline{C}_0 \geq N) \leq$$

$$N^{-\rho} \sum_{s=0}^{\infty} e^{-\alpha\Gamma(\mathbf{f}_s)} \left[\sum_{r=0}^{\infty} \sum_{p=r}^{\infty} \sum_{q=r}^{\infty} \sum_{\mathbf{f}_p', \mathbf{f}_q' \in (F_p' \times F_q')_r} e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')} \right]^{\frac{\rho}{2}} \quad (3.55)$$

Jensen's Inequality is then employed to put this in the form:

$$Pr\,(\overline{C}_0 \geq N) \leq N^{-\rho} \sum_{s=0}^{\infty} \sum_{r=0}^{\infty} \sum_{p=r}^{\infty} \sum_{q=r}^{\infty} P_{p,q,r,s}(\alpha,\rho) \qquad \begin{array}{c} \alpha \geq 0 \\ 1 \leq \rho \leq 2 \end{array} \quad (3.56)$$

where:

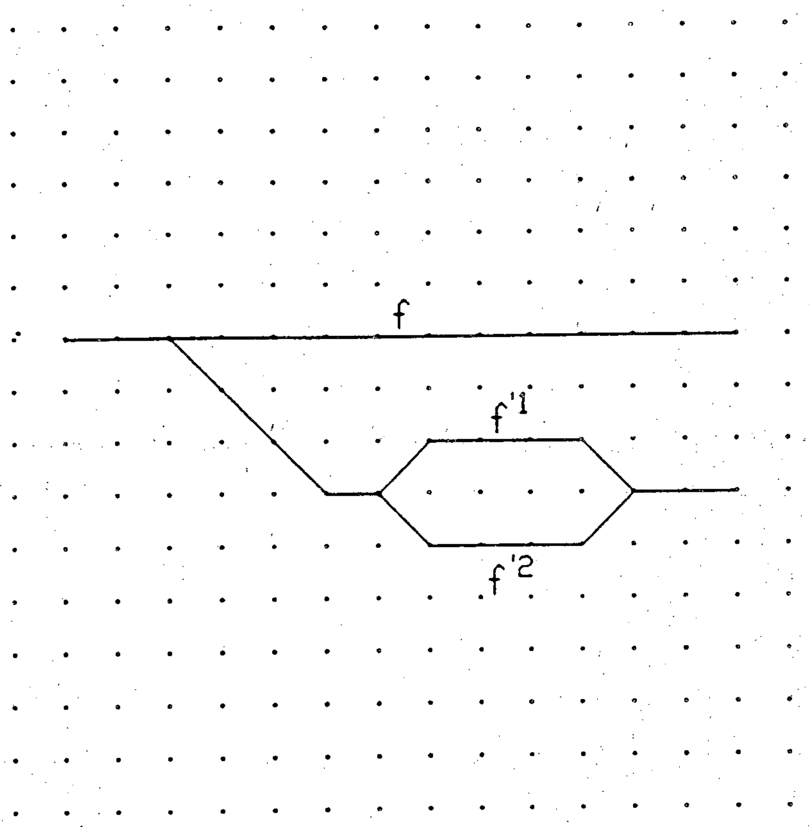

Figure 3.1: A topology of pairs of incorrect paths that cannot exist in the incorrect subtree $\overline{S}_0$.

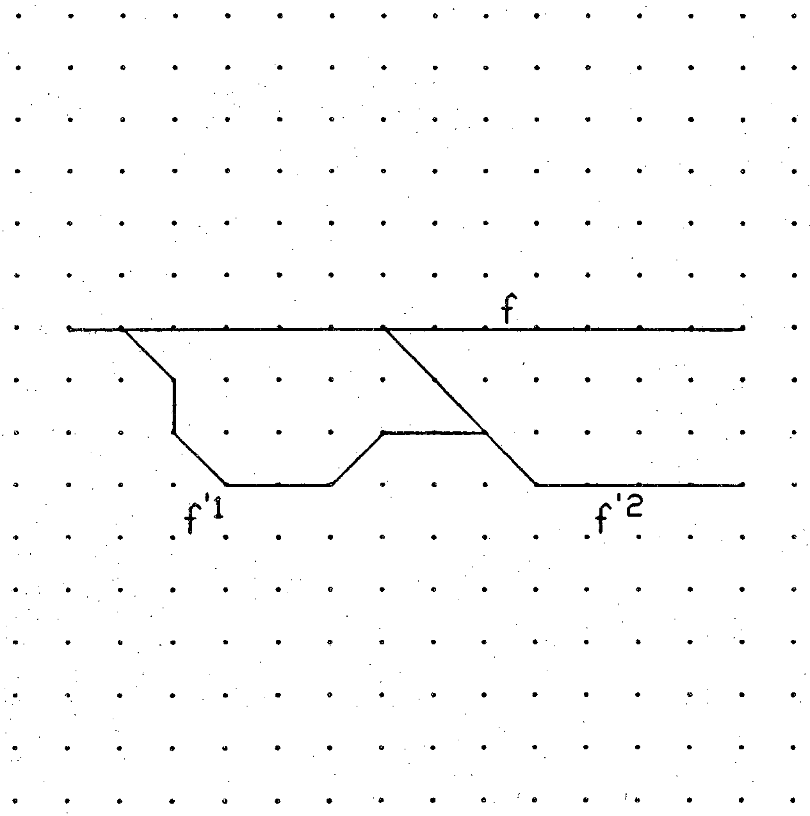

Figure 3.2: Another topology of pairs of incorrect paths that cannot exist in the incorrect subtree S_0.

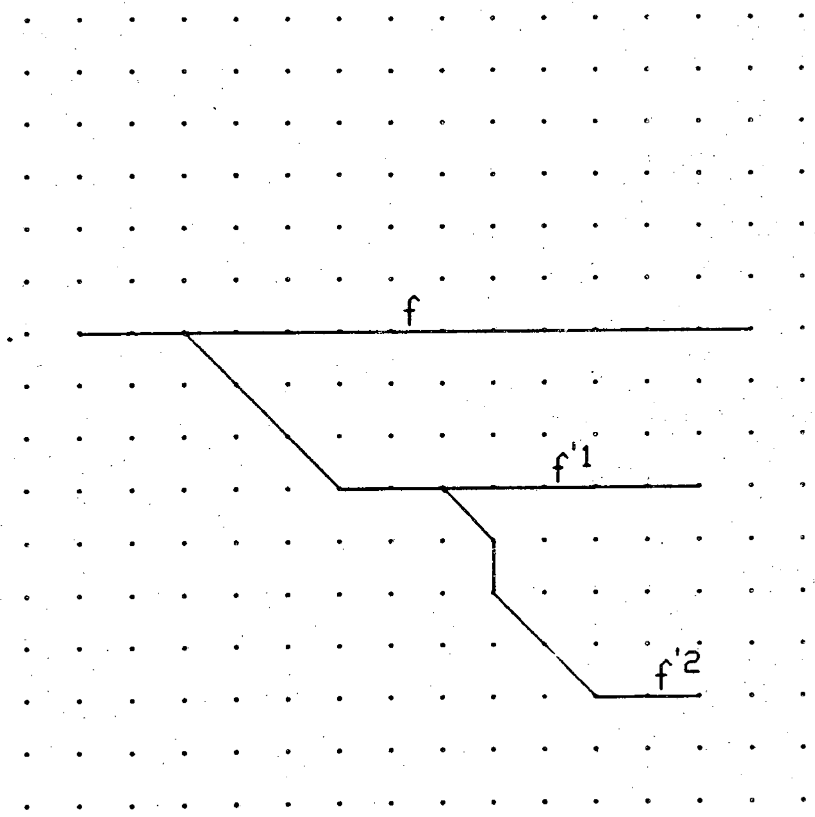

Figure 3.3: The only_topology of pairs of incorrect paths that can exist in the incorrect subtree $\bar{S}_0$.

$$P_{p,q,r,s}(\alpha,\rho) \;=\; \overline{\left[\sum_{\mathbf{f}_p', \mathbf{f}_q' \in (F_p' \times F_q')_r} e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')}\, e^{-2\alpha\Gamma(\mathbf{f}_s)} \right]^{\frac{\rho}{2}}} \tag{3.57}$$

Here, the expectation is with respect to all choices of $\mathbf{m}_s$, $\mathbf{m}_p'$, $\mathbf{m}_q'$, $\mathbf{f}_s$, $\mathbf{f}_p'$, and $\mathbf{f}_q'$. To save space, $\mathbf{f}_p'$,$\mathbf{f}_q'$ will be understood to mean $(\mathbf{f}_p' ,\mathbf{f}_q') \in (F_p' \times F_q')_r$ in the Equations to follow. Now,

$$\begin{aligned}
&P_{p,q,r,s}(\alpha,\rho) \\
&= \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_p'}\sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q') \;\overline{\left[\sum_{\mathbf{f}_p', \mathbf{f}_q'} e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')}\, e^{-2\alpha\Gamma(\mathbf{f}_s)} \right]^{\frac{\rho}{2}}} \\
&\le \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_p'}\sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q') \left[\overline{\sum_{\mathbf{f}_p', \mathbf{f}_q'} e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')}\, e^{-2\alpha\Gamma(\mathbf{f}_s)}} \right]^{\frac{\rho}{2}} \qquad (3.58) \\
&= \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_p'}\sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q') \left[\sum_{\mathbf{f}_p', \mathbf{f}_q'} \overline{e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')}\, e^{-2\alpha\Gamma(\mathbf{f}_s)}} \right]^{\frac{\rho}{2}} \\
&= |(F_p' \times F_q')_r|^{\frac{\rho}{2}} \sum_{\mathbf{f}_s}\sum_{\mathbf{f}_p'}\sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q') \\
&\qquad \cdot \left[\overline{e^{\alpha\Gamma(\mathbf{f}_p')}\, e^{\alpha\Gamma(\mathbf{f}_q')}\, e^{-2\alpha\Gamma(\mathbf{f}_s)}} \right]^{\frac{\rho}{2}}
\end{aligned}$$

where the inequality in the second step results from Equation (3.21), and the third step is due to the fact that the averages of a sum of terms is equal to the sum of the averages. Using the definitions of a_i and b_i given earlier, we have:

$$P_{p,q,r,s}(\alpha,\rho) \;\le\; \left[\overline{e^{\alpha \sum_{i=0}^{p} b_i' + \alpha \sum_{j=0}^{q} b_j' - 2\alpha \sum_{k=0}^{s} b_k}} \right]^{\frac{\rho}{2}} \left| (F_p' \times F_q')_r \right|^{\frac{\rho}{2}}$$

$$\cdot \sum_{\mathbf{f}_s} \sum_{\mathbf{f}_p'} \sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q')\, e^{\alpha\frac{\rho}{2} \sum_{i=0}^{p} a_i' + \alpha\frac{\rho}{2} \sum_{j=0}^{q} a_j' - \alpha\rho \sum_{k=0}^{s} a_k} \tag{3.59}$$

We now use the independence of f-values from different paths and the conditional independence of f-values along the same path. What is different here from Equation (3.27) is that $\mathbf{f}_p'$ and $\mathbf{f}_q'$ are *merged* (same f values) up to node r. We have the correct path to s, *one* incorrect path to r, and *two* incorrect paths from r to p and r to q.

$$\sum_{\mathbf{f}_s} \sum_{\mathbf{f}_p'} \sum_{\mathbf{f}_q'} p_1(\mathbf{f}_s)\, p_0(\mathbf{f}_p')\, p_0(\mathbf{f}_q')\, e^{\alpha\frac{\rho}{2} \sum_{i=0}^{p} a_i' + \alpha\frac{\rho}{2} \sum_{j=0}^{q} a_j' - \alpha\rho \sum_{k=0}^{s} a_k}$$

$$= \sum_{\mathbf{f}_s} p_1(\mathbf{f}_s) e^{-\alpha\rho \sum_{k=0}^{s} a_k} \sum_{\mathbf{f}_r'} p_0(\mathbf{f}_r') e^{\alpha\rho \sum_{l=0}^{r} a_l'} \sum_{\mathbf{f}_{p-r}'} p_0(\mathbf{f}_{p-r}') e^{\alpha\frac{\rho}{2} \sum_{i=r}^{p} a_i'} \sum_{\mathbf{f}_{q-r}'} p_0(\mathbf{f}_{q-r}') e^{\alpha\frac{\rho}{2} \sum_{j=r}^{q} a_j'}$$

$$= \prod_{k=0}^{s} p_1(f^k) e^{-\alpha\rho a_k} \prod_{l=0}^{r} p_0(f_l') e^{\alpha\rho a_l'} \prod_{i=r}^{p} p_0(f_i') e^{\alpha\frac{\rho}{2} a_i'} \prod_{j=r}^{r} p_0(f_j') e^{\alpha\frac{\rho}{2} a_j'}$$

$$= \left[\sum_{f} p_1(f)\, e^{-\alpha\rho a} \right]^{s} \left[\sum_{f'} p_0(f')\, e^{\alpha\rho a'} \right]^{r} \left[\sum_{f'} p_0(f')\, e^{\alpha\frac{\rho}{2} a'} \right]^{(p-r)+(q-r)}$$

$$= \left[\sum_{f} \left[p_1(f) \right]^{1-\alpha\rho} \left[p_0(f) \right]^{\alpha\rho} \right]^{s} \left[\sum_{f'} \left[p_1(f') \right]^{\alpha\rho} \left[p_0(f') \right]^{1-\alpha\rho} \right]^{r} \tag{3.60}$$

$$\cdot \left[\sum_{f'} \left[p_1(f') \right]^{\alpha\frac{\rho}{2}} \left[p_0(f') \right]^{1-\alpha\frac{\rho}{2}} \right]^{(p-r)+(q-r)}$$

$$= \left[D(\alpha\rho) \right]^{s} \left[D(1-\alpha\rho) \right]^{r} \left[D\left(1-\alpha\frac{\rho}{2}\right) \right]^{p+q-2r}$$

Likewise, the bracketed term of (3.59) is bounded in the same way as in Theorem 3.2 allowing for the fact that the path state sequences of the incorrect paths are identical up to node r .

$$\left[\overline{e^{\alpha \sum_{i=0}^{p} b_i' + \alpha \sum_{j=0}^{q} b_j' - 2\alpha \sum_{k=0}^{s} b_k}} \right]^{\frac{\rho}{2}}$$

$$= \left[\overline{e^{2\alpha \sum_{l=0}^{r} b_l' + \alpha \sum_{i=r}^{p} b_i' + \alpha \sum_{j=r}^{q} b_j' - 2\alpha \sum_{k=0}^{s} b_k}} \right]^{\frac{\rho}{2}}$$

$$= \left\{ \sum_{\mathbf{m}_r'} p(\mathbf{m}_r') \left[p(\mathbf{m}_r') \right]^{2\alpha} \cdot \sum_{\mathbf{m}_{p-r}'} p(\mathbf{m}_{p-r}') \left[p(\mathbf{m}_{p-r}') \right]^{\alpha} \right.$$
$$\left. \cdot \sum_{\mathbf{m}_{q-r}'} p(\mathbf{m}_{q-r}') \left[p(\mathbf{m}_{q-r}') \right]^{\alpha} \cdot \sum_{\mathbf{m}_s} p(\mathbf{m}_s) \left[p(\mathbf{m}_s) \right]^{-2\alpha} \right\}^{\frac{\rho}{2}} \qquad (3.61)$$

$$\leq \left\{ 1 \cdot 1 \cdot 1 \cdot \left[\sum_{\mathbf{m}_s} p(\mathbf{m}_s) \frac{1}{p(\mathbf{m}_s)} \right\}^{2\alpha} \right\}^{\frac{\rho}{2}}$$

$$\leq \left[3^s \right]^{2\alpha \frac{\rho}{2}} = 3^{s\alpha\rho}$$

Finally, we have:

$$| (F_p' \times F_q')_r |^{\frac{\rho}{2}} \leq \left[3^r \cdot 3^{p-r} \cdot 3^{q-r} \right]^{\frac{\rho}{2}}$$
$$= 3^{(p+q-r)\frac{\rho}{2}} \qquad (3.62)$$

so Equation (3.59) becomes:

$$
\begin{aligned}
& P_{p,q,r,s}(\alpha,\rho) \\
& \leq 3^{s\alpha\rho} \cdot 3^{(p+q-r)\frac{\rho}{2}} \cdot [D(\alpha\rho)]^s \, [D(1-\alpha\rho)]^r \left[D(1-\alpha\frac{\rho}{2})\right]^{p+q-2r} \\
& = [3^{\alpha\rho} \cdot D(\alpha\rho)]^s \left[3^{\frac{\rho}{2}} \cdot D(1-\alpha\rho)\right]^r \left[3^{\frac{\rho}{2}} \cdot D(1-\alpha\frac{\rho}{2})\right]^{p+q-2r} \qquad \textbf{(3.63)} \\
& = \beta_1^s \, \beta_2^r \, \beta_3^{p+q-2r} \qquad \begin{matrix} \alpha \geq 0 \\ 1 \leq \rho \leq 2 \end{matrix}
\end{aligned}
$$

Thus, if $\beta_1 < 1$, $\beta_2 < 1$, and $\beta_3 < 1$ (the hypothesis of the theorem), Equation (3.56) is bounded by:

$$
\begin{aligned}
Pr\,(\overline{C}_0 \geq N) & \leq N^{-\rho} \sum_{s=0}^{\infty} \sum_{r=0}^{\infty} \sum_{p=r}^{\infty} \sum_{q=r}^{\infty} \beta_1^s \, \beta_2^r \, \beta_3^{p-r} \, \beta_3^{q-r} \\
& = K \, N^{-\rho}
\end{aligned} \qquad (3.64)
$$

where:

$$
K = \frac{1}{(1-\beta_1)} \cdot \frac{1}{(1-\beta_2)} \cdot \frac{1}{(1-\beta_3)^2} \qquad \mathit{QED} \qquad (3.65)
$$

We offer the following Corollary without proof. It may be justified in similar fashion as Corollaries 3.1 and 3.2.

Corollary 3.3: For any $\alpha \geq 0$ and any $1 \leq \rho \leq 2$, if $D(\alpha\rho) < 3^{-\alpha\rho}$, $D(1-\alpha\rho) < 3^{-\frac{\rho}{2}}$, and $D(1-\frac{\alpha\rho}{2}) < 3^{-\frac{\rho}{2}}$, then:

$$
Pr\,(C_t \geq N) \leq K \, N^{-\rho} \qquad ;K \text{ as in } (3.65). \qquad (3.66)
$$

It seems reasonable to assume that the arguments developed in Theorems 3.2 and 3.4 can be extended to Pareto exponents greater than two. The hypothesis would continue to be more restricting on the f-value distributions p_0 and p_1 for increasing ρ. Furthermore, the number of possible incorrect path topologies involved in the partitioning of $\overline{S}_0$ increases rapidly with ρ. The notational burden would therefore become extreme.

Nevertheless, the above extension to $1 \leq \rho \leq 2$ is important in light of Theorem 3.3. The condition necessary to insure that the mean number of computations per node be finite is simply the hypothesis of Theorem 3.4. This result is more important from a qualitative standpoint than it is from a quantitative one. The problem with Chernoff bounds, as with most generalized Chebychev bounds is that they are not particularly tight. It has been found by simulation techniques that the analogous bound to Equation (3.41) in the context of coding theory is approximately two orders of magnitude higher than the true numbers of computations required in practical situations [Woz65]. Therefore, these bounds are not directly useful in a numerical sense. Theorems 3.3 and 3.4 have important *qualitative* implications, however. They indicate that sequential tree searching in two-dimensional random fields has a *threshold behavior* similar to such searching in code trees. That is, if the signal-to-noise ratio is high, the searching algorithm may be expected to explore a small number of incorrect paths at each correct path node, on the average. As the noise becomes more severe, a point is reached where the average number of computations per correct node becomes very large. (In an image of finite support, it can never really reach infinity). Thus there exists a threshold above which the algorithm is expected to be well behaved and below which the number of computations is expected to increase dramatically. In the case of code tree searching, this threshold can be shown to take the form of a code rate, known as the computational cutoff rate. In the image searching problem under consideration here, we see from these two theorems that this threshold is expressed in terms of the conditional densities p_1 and p_0 from the random field model.

3.3. Bound on the Probability of Error

3.3.1. Introduction: Edge Error Events

The question of quantifying edge errors has not received much attention in the literature. This is perhaps a result of the fact that many edge detection techniques deal only with *points* of an image and do not treat edge *segments* as entities. In this Section we shall develop a bound on the probability of certain edge segment error events.

We must first define the concept of an error event. The traditional edge detector performance measures consider only point-wise events. That is, they examine all the pixels labeled by an edge detector as *edge points* and flag as an error event any point that does not coincide with an edge. In the case of the Pratt figure of merit [Pra78], these error points are assigned a distortion measure: the mean square distance from the point to the edge. However, individual pixels are considered without regard to the surrounding edge pixels. Indeed, the Pratt figure of merit is useful even where the edge map is ambiguous. For example, in Figure 3.4, an edge map contains a smeared edge of multiple pixel width. It is not clear just where the detected edge is. The Pratt figure of merit can nevertheless calculate the distortion for every pixel in the map.

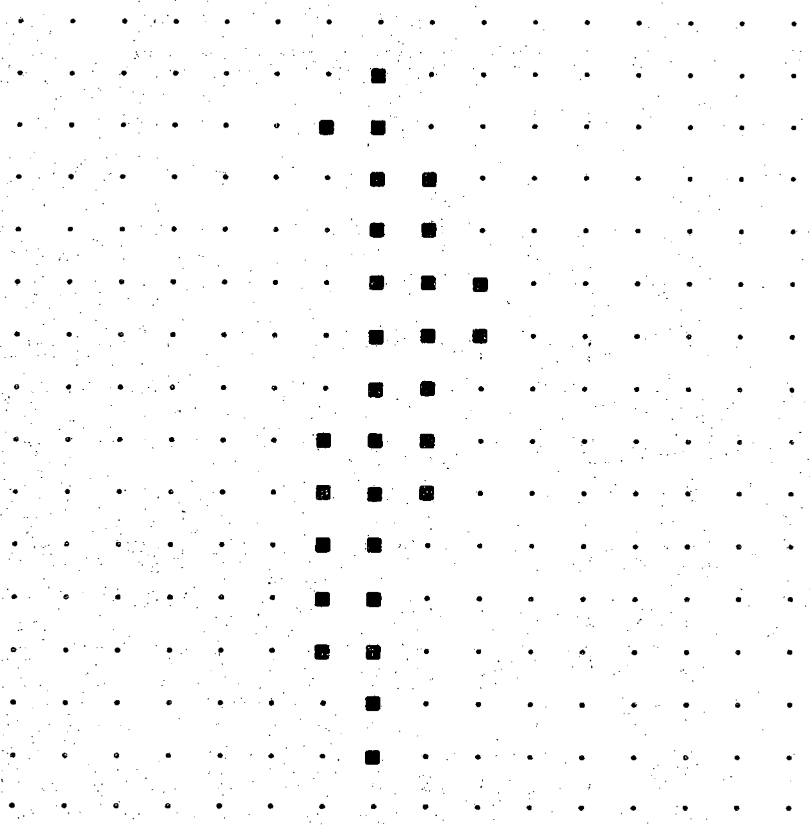

Figure 3.4: A smeared edge of multiple pixel width. The exact location of the edge is ambiguous.

Using the concept of paths as developed in Section 2.2.2, we can introduce the idea of an edge *segment* error event to supercede the edge *point* error event. The definition is straightforward. An error segment of length n, denoted $\mathbf{e}(n)$, is a path of length n nodes such that none of the nodes of $\mathbf{e}(n)$ coincides with any node of a true edge path. This situation is depicted in Figure 3.5 A true edge path, $\mathbf{m}$, is shown along with two error segments, $\mathbf{e}^1(n)$ and $\mathbf{e}^2(k)$. Note that an error segment can, but need not, terminate at a true edge. We would argue that since the goal of edge detection is usually to find *connected* points that represent an intensity edge in an image, the concept of error segments is a more natural choice for an error event than is that of point errors. (Actually, error segments are a generalization of point errors since a point error is an error segment of length one.) Furthermore, error segments are particularly useful when investigating sequential algorithms as they are a complete description of all possible error events that can be generated by such algorithms.

3.3.2. Bound on Long Error Segment Probability

In this Section we examine long error segments. We will show that the probability of occurrence of these events depends on the image "quality" (via the true-edge and random-path probability measures p_1 and p_0), and is bounded by an exponentially decreasing function of length.

Error segments can, but need not, be connected at one end or both to a true edge path. We shall begin our investigation with segments connected at both ends (Figure 3.5). These events arise in the following manner. At some node, t, along a true edge path, the searching algorithm begins exploring paths in the incorrect subtree, S_t.

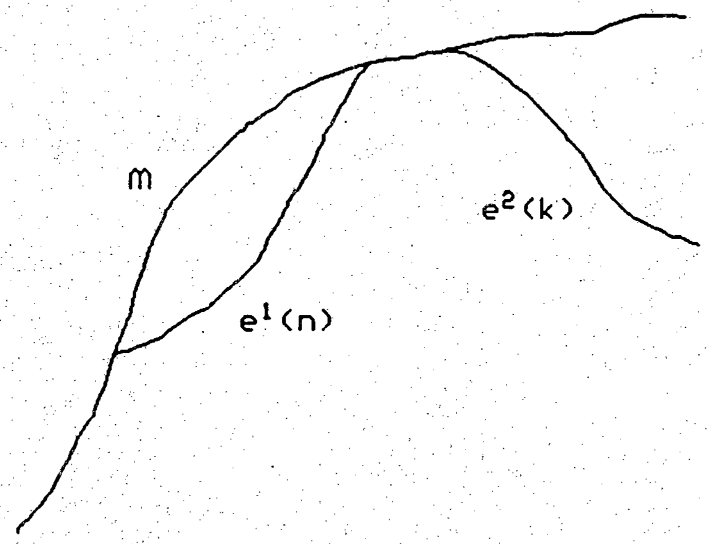

Figure 3.5: A correct edge path $\mathbf{m}$ and two error segments $\mathbf{e}^1(n)$ and $\mathbf{e}^2(k)$.

Because in a two-dimensional lattice (and therefore in S_t) remergings *do* occur, it may turn out that, due to noise in the observations, the algorithm eventually chooses a path $\mathbf{f}'$ instead of the correct path, $\mathbf{f}$, where $\mathbf{f}'$ coincides with $\mathbf{f}$ up to node t, branches off at t, and remerges with $\mathbf{f}$ some n nodes later. Thus the chosen path $\mathbf{f}'$ contains an error segment of length n, $\mathbf{e}(n)$.

Let us denote the event the chosen path contains an error segment of length n *or longer* by $E(n)$. We will denote by $S_{t,t+n}$ the set of all paths in the incorrect subtree S_t that do not remerge with the correct path over the interval $[t, t+n]$. That is, all paths in $S_{t,t+n}$ contain an error segment at least n nodes long. Now from Lemma 3.1,

Lemma 3.2: A path $\mathbf{f}'_m \in S_t$ that remerges with the correct path $\mathbf{f}$ at node m, $m > t$, will be chosen over $\mathbf{f}$ *only if:*

$$\Gamma(\mathbf{f}'_m) \geq \Gamma(\mathbf{f}_t) - \Delta_t \qquad (3.67)$$

This lemma follows directly from Lemma 3.1 since, unless Equation (3.67) is true, $\mathbf{f}'_m$ will never be extended by the searching algorithm and hence cannot be chosen. Note that this establishes only necessity not sufficiency.

As in Section 3.2.3, we suffer no loss of generality by assuming $t = 0$. Lemma 3.2 then implies (similarly to Theorem 3.1):

Theorem 3.5: The probability of the event $E(n)$ is upper bounded by:

$$Pr\left\{E(n)\right\} \leq Pr\left\{\sum_{\mathbf{f}' \in S_{0,n}} \phi(\mathbf{f}') \geq 1\right\} \qquad (3.68)$$

where:

$$\phi(\mathbf{f}') = \begin{Bmatrix} 1 & , \text{if } \Gamma(\mathbf{f}') \geq -\Delta_t \\ 0 & , \textit{otherwise} \end{Bmatrix} \tag{3.69}$$

This theorem follows directly from Lemma 3.2. For the event $E(n)$ to occur, some path in $S_{0,n}$ must be chosen over the correct path. This can occur only if Equation (3.67) is satisfied ($t = 0; \Gamma(\mathbf{f}_t) = 0$), from which Equation (3.68) follows.

We now state and prove the following important theorem.

Theorem 3.6: For any $\alpha \geq 0$ and $0 \leq \rho \leq 1$, if: $D(1-\alpha\rho) < 3^{-\rho}$ and $D(\alpha\rho) < 3^{-\alpha\rho}$, then:

$$Pr\,(E(n)) \leq K\,\gamma^n \qquad ; \gamma < 1 \quad ; K \textit{ a constant.} \tag{3.70}$$

Proof:

We now define the quantity $T(\alpha,\rho)$ as:

$$T(\alpha,\rho) = \left[\sum_{\mathbf{f}' \in S_{0,n}} e^{\alpha\Gamma(\mathbf{f}')} \right]^{\rho} \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.71}$$

Since:

$$\phi(\mathbf{f}') \leq e^{\alpha\Gamma(\mathbf{f}') + \alpha\Delta_0} \tag{3.72}$$

(which is Equation (3.8)), Equations (3.71) and (3.72) give us:

$$T(\alpha,\rho) = e^{-\alpha\rho\Delta_0}\left[\sum_{\mathbf{f}' \in S_{0,n}} e^{\alpha\Gamma(\mathbf{f}') + \alpha\Delta_0}\right]^{\rho} \geq e^{-\alpha\rho\Delta_0}\left[\sum_{\mathbf{f}' \in S_{0,n}} \phi(\mathbf{f}')\right]^{\rho} \qquad (3.73)$$

and so, by the Chebyshev Inequality:

$$Pr\left(\sum_{\mathbf{f}' \in S_{0,n}} \phi(\mathbf{f}') \geq 1\right) \leq \overline{T(\alpha,\rho)\, e^{\alpha\rho\Delta_0}} \qquad (3.74)$$

Upper bounding $e^{\alpha\rho\Delta_0}$ with Equation (3.11) and using Theorem 3.5, we have:

$$Pr(E(n)) \leq \sum_{s=0}^{\infty} \overline{e^{-\alpha\rho\Gamma(\mathbf{f}_s)}\left[\sum_{\mathbf{f}' \in S_{0,s}} e^{\alpha\Gamma(\mathbf{f}')}\right]^{\rho}} \qquad \begin{matrix}\alpha \geq 0\\ 0 \leq \rho \leq 1\end{matrix} \quad (3.75)$$

where the expectation is over all choices for $\mathbf{m}$, $\mathbf{m}'$, $\mathbf{f}$, and $\mathbf{f}'$. Equation (3.75) is very similar to Equation (3.12) except that the term $N^{-\rho}$ is not present and the inner sum is over the set $S_{0,n}$ instead of $\overline{S}_0$. We partition this set as:

$$S_{0,n} = \bigcup_{r=0}^{\infty} G'_r \qquad (3.76)$$

where:

$$G'_r = \left\{ \mathbf{f}'_m \in S_{0,n} \mid m = r \right\} \qquad (3.77)$$

So:

$$Pr(E(n)) \leq \sum_{s=0}^{\infty} \overline{e^{-\alpha\rho\Gamma(\mathbf{f}_s)}\left[\sum_{r=n}^{\infty} \sum_{\mathbf{f}_r' \in G_r'} e^{\alpha\Gamma(\mathbf{f}_r')}\right]^{\rho}} \qquad (3.78)$$

The remaining steps of the proof follow exactly those of Theorem 3.2, Equations (3.15) -

(3.36), the difference being: no $N^{-\rho}$ term, sets G'_r instead of F'_r, and the index r running from n to infinity instead of 0 to infinity. Since remergings occur in the sets G'_r but not in the sets F'_r, we have, using Equation (3.34):

$$| G'_r | \leq | F'_r | \leq 3^r \tag{3.79}$$

We therefore obtain, corresponding to Equation (3.36) of Theorem 3.2:

$$Pr\,(E(n)) \leq \sum_{r=n}^{\infty} \sum_{s=0}^{\infty} \beta_1^r \beta_2^s \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.80}$$

where:

$$\begin{aligned} \beta_1 &= 3^{\rho} \cdot D(1-\alpha\rho) < 1 \\ \beta_2 &= 3^{\alpha\rho} \cdot D(\alpha\rho) < 1 \end{aligned} \tag{3.81}$$

and the inequalities are satisfied by hypothesis. We have finally:

$$Pr\,(E(n)) \leq K\,\gamma^n \qquad \begin{matrix} \alpha \geq 0 \\ 0 \leq \rho \leq 1 \end{matrix} \tag{3.82}$$

$$K = \frac{1}{1-\beta_1} \cdot \frac{1}{1-\beta_2}$$

$$\gamma = \beta_1 < 1 \qquad\qquad Q.E.D.$$

Theorem 3.6 says that the probability of long error segments in the edges found by SEL is bounded by an exponentially decreasing function of length. The important parameter $\gamma = 3^{\rho} \cdot D(1-\alpha\rho)$ depends on the measures p_0 and p_1. As we will see in Section 3.4, a high image signal to noise ratio results in a low value for $D(1-\alpha\rho)$ and γ. This provides a link between the theory and the intuitive belief that higher SNR results in fewer (and shorter) error events.

More importantly, however, this theorem provides us with an upper bound on the probability that the real edge path does not fall in the stack of paths searched by the algorithm. As pointed out in Chapter 2, this searching technique does not guarantee that the chosen path is the *maximum likelihood* path. The path with the highest metric is the path *among those searched* that maximizes the joint probability of Equation (2.39). It is not necessarily the path that maximizes that statistic over the entire set of paths in the field. The root node is assumed to lie on the correct edge. Therefore, at least the first node of the correct path will always be on the search stack. But there is some nonzero probability that the correct path is only partially searched and then discarded. This implies that the path on the top of the stack is an incorrect path. Therefore, the probability that the correct edge path is not on the search stack is also upper bounded by Theorem 3.6. As the best path on the stack grows longer and longer, this bounding probability becomes exponentially smaller.

The theorem above appears to address only the error event situation of Figure 3.5 where the incorrect path diverges from a correct path and later remerges again. In fact, it also applies to the situation of Figure 3.5 where no remerging takes place. This nonphysical case results in an infinite length error segment which, according to the theorem, has a probability of occurrence that approaches zero in the limit.

The situations of Figure 3.6 (an isolated error segment) or of Figure 3.7 (an error segment that begins at a random node and merges with a true edge path) are not governed by the theorem because there is no correct path on the stack of the searching algorithm with metric dip, Δ, to compete with the incorrect paths. The sets S_t or $S_{t,n}$ are not even defined where a correct path does not go through node t. These two situations are more properly treated as a root node problem (Section 2.4) since they arise on account of improper root node selection.

3.3.3. Short Error Segment Events

The approach taken in Section 3.3.2 for long segments can tell us nothing about short segments. This is a consequence of Chernoff bounding techniques in general, since such bounds become tight only when sums are taken over many random variables [Gal68].

Short error segment events are fundamentally more difficult to handle in detection of edges in two-dimensional random fields than in coding applications. This is because there is nothing in images corresponding to the concept of the *minimum free distance* of a code [McE77]. In sequential decoding of convolutional codes, when the searching algorithm branches off from the correct path, the structure of the code tree is such that remergings within some minimum number of nodes, say ν, *is not possible*. Thus, short error events cannot occur. The philosophy regarding error probability is to design codes with distance ν large enough that the probability of a long error event, $E(\nu)$, is small enough to meet design requirements [For74b].

That this is not the case for images is quite easy to see. Figure 3.8 depicts a hypothetical edge, $\mathbf{m}$, through nodes t_1 and t_2, and an incorrect path, $\mathbf{m}'$, with an error segment of length 2 through nodes t'_1 and t'_2. $\mathbf{m}'$ is clearly a valid path as defined earlier. Thus, with some non-zero probability, the noise in the f-values at nodes t_1, t_2, t'_1, and t'_2 may be such that the path metric along $\mathbf{m}'$ is higher than that along $\mathbf{m}$.

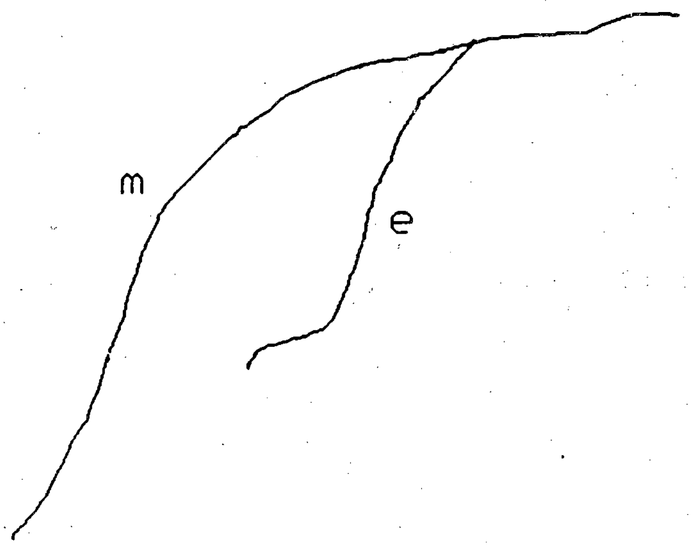

Figure 3.6: An error segment **e** that merges with the correct path **m**.

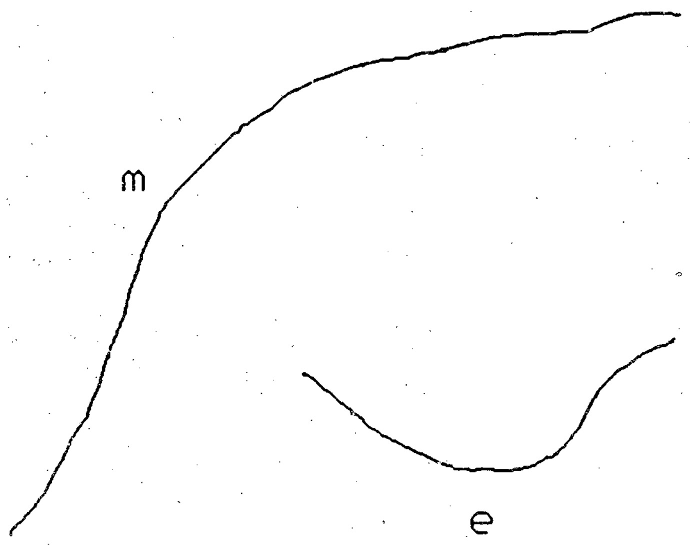

Figure 3.7: A correct edge path **m** and an isolated error segment **e**.

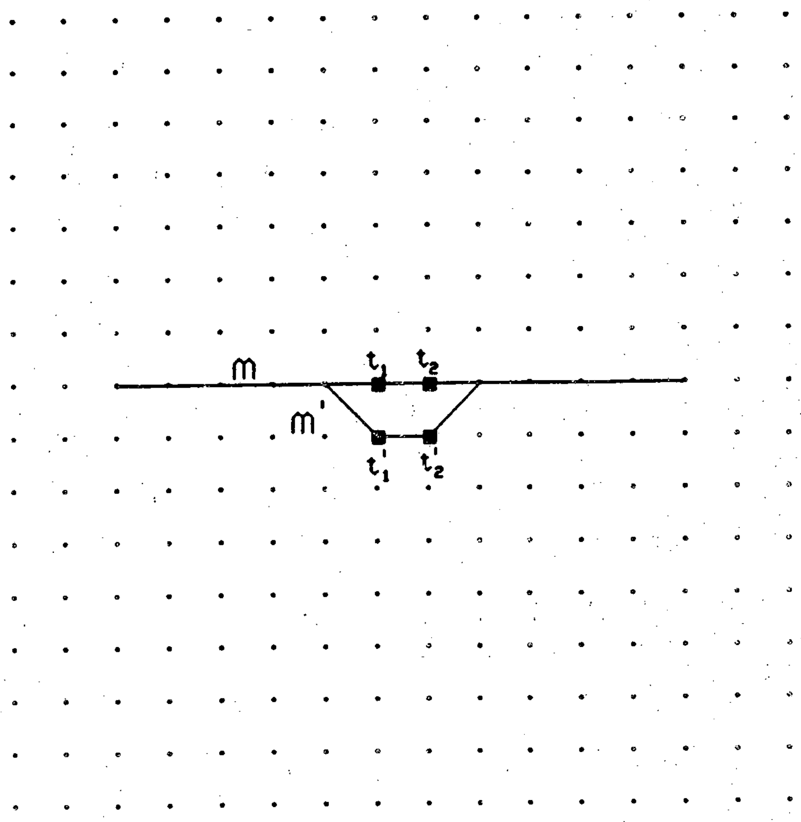

Figure 3.8: A correct path **m** through nodes t_1 and t_2 and an incorrect path **m**$'$ with an error segment through nodes t'_1 and t'_2.

Short error segments are difficult to prevent. In the example above, using the *only data available* (the corrupted image data), the path $\mathbf{m}'$ is more likely than $\mathbf{m}$ to be a true edge path. Thus, no matter what edge detection strategy is used, in the absence of more a-priori information, $\mathbf{m}'$ *ought* to be chosen over $\mathbf{m}$.

An important source of a-priori information *is* available in even very general situations, however. This is that certain edge paths (such as those containing straight segments) are more probable than others (Appendix). The path branch metric of SEL takes advantage of this information by modeling real edge paths by the Markov process of Section 2.2.2. Returning again to the example above, path $\mathbf{m}'$ has a lower *a-priori* probability than path $\mathbf{m}$, so $\mathbf{m}'$ will be chosen only if the f-values at t_1, t_2, t'_1, and t'_2 are so noisy that the likelihood ratio of the f-values dominates the probability of the path transitions. In other words, the incorporation of a-priori path probabilities into the branch metric helps to offset the tendency to commit short segment errors.

In general, further a-priori information on the true edge paths in an image is not available. In specific situations more may be known. In *temporal sequences* of images, for example, one may use the edges found in one frame of the sequence as a guide to those in the subsequent frame. Or, in medical angiocardiograms, where the desired edge is the contour of the heart, one may employ a training set of prototype contours to aid in the process. These types of very specific a-priori information *can* be incorporated into a sequential searching algorithm in a heuristic fashion. The reader is referred to Modestino [Mod76] for a discussion of this technique.

3.4. Comparisons with the Coding Problem

As noted in previous Sections, SEL is similar in many respects to the method of decoding convolutional codes by sequential tree searching. In this Section, we will clarify some of the major differences and similarities between these two problems.

3.4.1. The Path Branch Metric

The branch metric for sequential decoding of convolutional codes was first proposed by Fano [Fan63] using a heuristic argument and was later derived by Massey [Mas72]. That metric is:

$$\Gamma(\mathbf{y}) = \sum_{i=1}^{n} \left\{ \ln \left[\frac{P(y_i \mid x_i)}{P_0(y_i)} \right] - R \right\} \tag{3.83}$$

where $\mathbf{y} = (y_1, \ldots, y_n)$ are the received symbols, $\mathbf{x} = (x_1, \ldots, x_n)$ are the transmitted symbols corresponding to a particular path through the tree, R is the code rate, $P(\cdot \mid \cdot)$ are the channel transition probabilities, and $P_0(y_i) = \sum_i P(y_i \mid x_i) Pr(x_i)$ is the a-priori probability of receiving y_i. The derivation of Equation (3.83) assumes that the x_i's are i.i.d. and the channel is memoryless, i.e. y_i depends only on x_i.

Although the metrics of Equations (2.18) and (3.83) appear to be quite different, they are closer than it might seem at first glance. The bracketed term of (3.83) is a likelihood ratio of the probability of observing y_i under the hypothesis that $\mathbf{x}$ is the correct path, to the probability of y_i under the null hypothesis that it is a random path. The primary difference between this and Equation (2.17) is in the observations. In SEL, when the searching algorithm hypothesizes two different paths, $\mathbf{m}^1$ and $\mathbf{m}^2$, it obtains *two* sets of observations, $\mathbf{f}^1$ and $\mathbf{f}^2$, to incorporate into their corresponding metrics. In decoding, however, the decoder may hypothesize two paths, $\mathbf{x}^1$ and $\mathbf{x}^2$, but the observations, $\mathbf{y}$, are simply the received symbols and do not change. Thus, two metrics corresponding to two paths in sequential decoding are a function of *three* vectors, $\mathbf{y}$, $\mathbf{x}^1$, and $\mathbf{x}^2$, whereas in image searching they are a function of *four* vectors, $\mathbf{f}^1$, $\mathbf{f}^2$, $\mathbf{m}^1$, and $\mathbf{m}^2$. This important difference surfaces again in performance analysis where the expectation of Equation (3.10) must be taken over $\mathbf{m}$, $\mathbf{m}'$, $\mathbf{f}$, and $\mathbf{f}'$. The corresponding expectation in the coding problem is taken over $\mathbf{x}$, $\mathbf{x}'$, and $\mathbf{y}$.

The rate term, R in Equation (3.83) is closely related to the Markov transition probability term in Equation (2.18). To see this, we note that the a-priori probability of any transmitted vector $\mathbf{x}$ of length n is:

$$Pr\ (\mathbf{x}_n) = e^{-R\,n} \tag{3.84}$$

assuming the source is independent and identically distributed. Substituting this into (3.83) for R we have:

$$\begin{aligned}\Gamma(\mathbf{y}) &= \sum_{i=1}^{n} \left\{ \ln \left[\frac{P(y_i \mid x_i)}{P_0(y_i)} \right] + \frac{1}{n} \ln \left[Pr\ (\mathbf{x}) \right] \right\} \\ &= \sum_{i=1}^{n} \left\{ \ln \left[\frac{P(y_i \mid x_i)}{P_0(y_i)} \right] \right\} + \ln \left[Pr\ (\mathbf{x}) \right]\end{aligned} \tag{3.85}$$

But Equation (2.18) for SEL is:

$$\begin{aligned}\Gamma(\mathbf{m},\mathbf{f}) &= \sum_{i=1}^{n} \left\{ \ln\left[\frac{p_1(f_i)}{p_0(f_i)}\right] + \ln\left[Pr\ (s_i \mid s_{i-1})\right] \right\} \\ &= \sum_{i=1}^{n} \left\{ \ln\left[\frac{p_1(f_i)}{p_0(f_i)}\right] \right\} + \ln\left[Pr\ (\mathbf{m})\right]\end{aligned} \tag{3.86}$$

so the second term of both metrics is present to take into account the a-priori probability of the hypothesized path. The difference between them is that the path letters in sequential decoding are assumed to be independent and identically distributed (iid). In SEL they are correlated and modeled as a Markov chain. If they were assumed to be iid, then the SEL metric would have a constant term in its metric. Conversely, this analysis indicates that if the information source of a sequential encoder was Markov instead of iid, then the correct path metric would be of the form:

$$\Gamma(\mathbf{y}) = \sum_{i=1}^{n} \left\{ \ln\left[\frac{P(y_i \mid x_i)}{P_0(y_i)}\right] + \ln\left[Pr\ (s_i \mid s_{i-1})\right] \right\} \tag{3.87}$$

3.4.2. D(δ) and the Random Coding Exponent

As we have seen in Sections 3.2 and 3.3, the quantity D(δ) plays a decisive role in the amount of incorrect path searching the algorithm suffers and the probability of error. It is in some sense a measure of the "quality" of edges in an image; the lower the value of D(δ), the smaller the probability of error and number of incorrect paths searched.

From the Holder Inequality we see that:

$$\begin{aligned} D(\delta) &= \sum_f \left[p_0(f)\right]^\delta \left[p_1(f)\right]^{1-\delta} \qquad 0 \le \delta \le 1 \\ &\le \left[\sum_f p_0(f)\right]^\delta \left[\sum_f p_1(f)\right]^{1-\delta} \\ &= 1 \end{aligned} \tag{3.88}$$

with equality if and only if $p_0(f) = p_1(f)$ for all f. Therefore, the "quality" measure D(δ) is worst when $p_0 = p_1$; that is, when edge paths and random paths are *indistinguishable*. It becomes better (lower) as p_0 and p_1 have less overlap. Figures 3.9a and 3.9b illustrate this point.

In the coding problem, the quantity $E_0(\rho)$,

$$E_0(\rho) = -\ln \sum_j \left[\sum_k p(x_k) p(y_j \mid x_k)^{\frac{1}{1+\rho}} \right]^{1+\rho} \tag{3.89}$$

plays approximately the same role as D(δ) does for SEL. $E_0(\rho)$, called the random coding exponent, is a function of the channel transition probabilities. When the cross-over probability of the channel is small, i.e. $Pr\ (y_i = x_i)$ is very nearly equal to 1, the random coding exponent is large and sequential decoding proceeds with few incorrect path hypotheses.

As in the case of the branch metrics, D(δ) and $E_0(\rho)$ are also similar in form. To see this, E_0 is rewritten as:

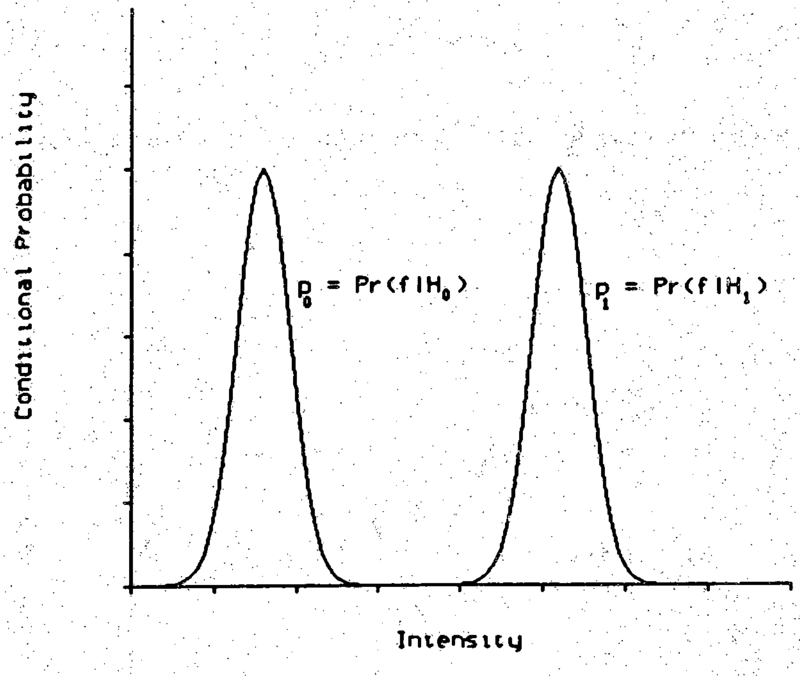

Figure 3.9a: High signal-to-noise ratio; little overlap of the conditional measures p_0 and p_1. $D(\delta) \ll 1$.

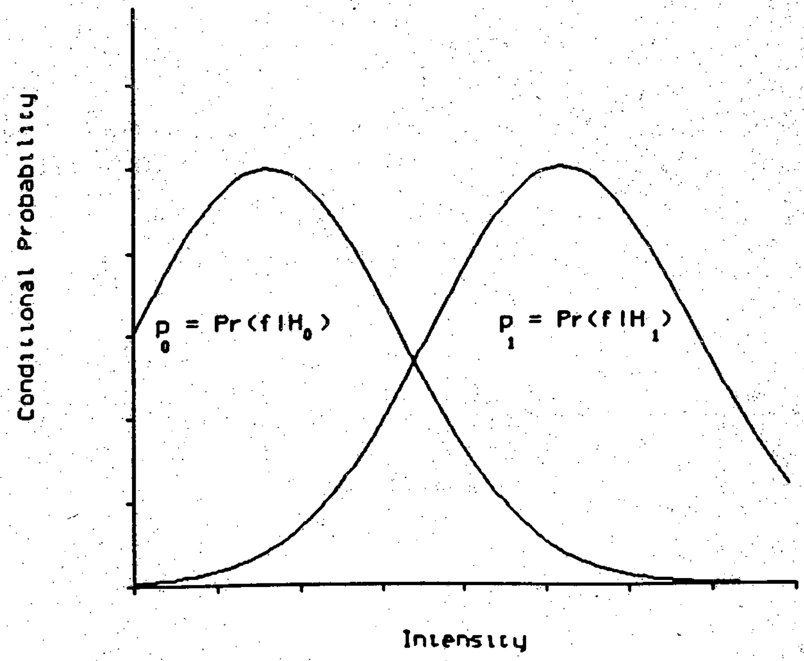

Figure 3.9b: Low signal-to-noise ratio; much overlap of the conditional measures p_0 and p_1. $D(\delta) \sim 1$.

$$e^{-E_0(\rho)} = \sum_j \left[\sum_k p(x_k) p(y_j \mid x_k)^{\frac{1}{1+\rho}} \right]^{1+\rho}$$

$$= \sum_j \sum_k p(x_k) p(y_j \mid x_k)^{\frac{1}{1+\rho}} \left[\sum_k p(x_k) p(y_j \mid x_k)^{\frac{1}{1+\rho}} \right]^{\rho} \quad (3.90)$$

$$= \sum_j \sum_k p(x_k) p(y_j \mid x_k)^{1-\alpha\rho} \left[\sum_k p(x_k) p(y_j \mid x_k)^{\alpha} \right]^{\rho}$$

where $\alpha = \dfrac{1}{1+\rho}$. Contrast this with D($\delta$):

$$D(\alpha,\rho) = \sum_f p_1(f)^{1-\alpha\rho} p_0(f)^{\alpha\rho} \quad (3.91)$$

Furthermore, it can be shown [For74b] that sequential decoding possesses finite average decoding time when the code rate R satisfies:

$$R < R_{comp} = E_0(1) \quad (3.92)$$

The rate is given by: $R = \dfrac{\ln(M)}{L}$, where M is the alphabet size of code symbols and L is the number of code symbols per source symbol. Equation (3.92) is therefore:

$$\ln(M) < -n \ln \sum_j \left[\sum_k p(x_k) p(y_j \mid x_k)^{\frac{1}{2}} \right]^2 \quad (3.93)$$

Corollary 3.3 specifies similar conditions for SEL to have a finite mean number of computations, one of which is:

$$\ln(3) \quad < \quad -\ln\left[\sum_{f} p_1(f)^{\frac{1}{2}}\; p_0(f)^{\frac{1}{2}}\right]^2 \qquad \textbf{(3.94)}$$

where $\alpha\,\rho$ is set to $\frac{1}{2}$. Equations (3.93) and (3.94) point out the very similar roles that D(δ) and $E_0(\rho)$ play in the analysis of SEL and sequential decoding respectively.

CHAPTER 4

THE DISCRETE-STEP ISOTROPIC MARKOV FIELD

4.1. Introduction

In Chapter 2, we derived our branch metric under the simplifying assumption that the observations or random field values $\{f_{\vec{r}_i}\}$ along paths are conditionally independent. This allowed the joint probability to be written as the product shown in Equation (2.17). A direct result of this assumption is the recursive computability of the path metric. Various aspects of the analysis of the distribution of computation and probability of error presented in Chapter 3 also required this assumption of independence. In this Chapter, we will introduce a new model for *correlated* random fields for which it is possible to generate a sequence of path *innovations* that are independent. With the path metric defined in terms of this innovations process, the previous results of Chapter 2 and 3 are preserved.

It is important to note that none of the previous investigators of sequential searching techniques for images have generalized their methods to include correlated data. The early work of Martelli [Mar76b] and Chien and Fu [Chi74] are not explicit on this point, but their running path metrics implicitly assume that the data along the paths are independent. The random field model of Cooper [Coo79] explicitly assumes the field is composed of a deterministic two-valued component ("background" and "object") and an additive *i.i.d.* Gaussian random noise field. Two investigators did consider Markov random field models [Han82], but in the context of a raster scan search rather than a sequential search. Finally, the ad hoc metric of Ashkar and Modestino [Ash78] is the sum of independent branch terms that do not depend on any correlation with neighboring branches.

This is in spite of the fact that it has been demonstrated by many investigators that the pixel values in many real world images are highly correlated. (See, for example, [Wil71], or [Nis65].) In fact, the rather large discipline of compression coding of images is based precisely on this fact [Dav72]. The observation that compression ratios of 8:1 or higher have been achieved with little subjective distortion of real images attests to the large measure of correlatedness present in those images.

The organization for this Chapter is as follows. In Section 4.2. we review the more common two-dimensional random field models. Section 4.3. details the advantages obtained by the imposition of path structures. This is followed by a development of the Discrete-Step Isotropic Markov Random Field (D-SIM) model together with its impact on SEL in Section 4.4.

4.2. Two-Dimensional Random Field Models

4.2.1. Notation and Assumptions

An image will be considered as a sample function of a two-dimensional discrete parameter random field. The real-valued function $f_{\vec{r}_i}(\omega)$, defined on the product space of Ω and the nodes of the lattice I $\times$ I is the value of the sample function at the node whose coordinates are $\vec{r}_i = (r_i^1, r_i^2) \in I^2$.

We will denote the autocorrelation function of f as $R_f\ (i,j,m,n) = E\left\{f_{ij}\ f_{mn}\right\}$. Throughout this Chapter, we will only be concerned with homogeneous random fields so that:

$$R_f\ (i,j,m,n) = R_f\ (i-m,j-n) = R_f\ (k,l) \tag{4.1}$$

The spectrum of a field f is defined in the usual manner as the two-dimensional Fourier Transform of the autocorrelation function (assuming R_f is absolutely summable):

$$S_f\,(u,v) = \begin{cases} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} R_f\ (k,l)\ e^{2\pi j(ku+lv)} & -0.5 \le u,v \le 0.5 \\ 0 & |u| > 0.5 \ \ or\ \ |v| > 0.5 \end{cases} \tag{4.2}$$

4.2.2. The Markov Mesh Model

In their 1965 paper [Abe65], Abend, et. al. generalized the idea of a Markov Chain to two dimensions in an effort to remove the restricting assumption of statistical independence among the random variables of a field. A fundamental question is how to extend the notions of "past", "present", and "future" from one-demensional Markov theory to two dimensions. To summarize their results, we shall need the following definitions:

(1) The field is of finite extent, restricted to an array of size M rows and N columns.

(2) $X_{m,n} = \left\{f_{r^1,r^2} \mid r^1 \le m \ \text{ and } \ r^2 \le n\ \right\}$; an m x n array of random variables comprising the upper left hand quadrant with respect to the node (m,n).

Note that $X_{M,N}$ is the entire field.

(3) $X_{m,n}^{a,b} = X_{m,n} - \left\{ f_{a,b} \right\}$; i.e. $X_{m,n}$ with the variable $f_{a,b}$ deleted.

(4) $Z_{a,b} = \left\{ f_{r^1,r^2} \mid r^1 < a \quad or \quad r^2 < b \right\}$; a nonrectangular array of variables lying to the left of, or above the node (a,b).

The assumption employed as a starting point in the work of Abend, et. al. is that for any (a,b), there exists a subset $U_{a,b} \subset X_{a,b}^{a,b}$ such that the conditional probability of a random variable $f_{a,b}$ given the variables $Z_{a,b}$ is equal to the conditional probability of $f_{a,b}$ given $U_{a,b}$, i.e.:

$$Pr\,(f_{a,b} \mid Z_{a,b}) = Pr\,(f_{a,b} \mid U_{a,b}) \quad , U_{a,b} \subset X_{a,b}^{a,b} \tag{4.3}$$

For example, what they call a third-order Markov Mesh has the property:

$$Pr\,(f_{a,b} \mid Z_{a,b}) = Pr\,(f_{a,b} \mid f_{a-1,b} \, , f_{a-1,b-1} \, , f_{a,b-1}) \tag{4.4}$$

Such a model has also been called a strict-sense Markov random field [Ros76]. With this formulation, the "past" in the two-dimensional sense is composed of the variables $Z_{a,b}$, the "present" is $f_{a,b}$ and the "future" is everything else.

The paper also shows that a key property for one-dimensional Markov Chains may be extended to the two-dimensional case defined above. This property is that the one-sided Markov dependence implies a two-sided dependence. For first-order chains, this means:

$$Pr\ (x_k \mid x_{k-1}, x_{k-2}, \ldots) = Pr\ (x_k \mid x_{k-1}) \tag{4.5}$$

implies:

$$Pr\ (x_k \mid \ \ldots, x_{k-1}, x_{k+1}, \ldots) = Pr\ (x_k \mid x_{k-1}, x_{k+1}) \tag{4.6}$$

and similarly for higher-order chains. The converse is not necessarily true. In direct analogy to this, Abend shows that, for example, the third-order Markov mesh assumption implies:

$$Pr\ (f_{a,b} \mid X_{M,N}^{a,b}) = Pr \left(f_{a,b} \middle| \begin{array}{ccc} f_{a-1,b-1} & f_{a-1,b} & f_{a-1,b+1} \\ f_{a,b-1} & & f_{a,b+1} \\ f_{a+1,b-1} & f_{a+1,b} & f_{a+1,b+1} \end{array} \right) \tag{4.7}$$

This result demonstrates that the Markov mesh assumption leads to the reasonable conclusion that the conditional probability of $f_{a,b}$ given the entire array is the same as that given only its eight nearest neighbors.

As discussed in Kanal [Kan80], this model has not proven particularly useful for parameter estimation or classification in real images but is more suited to image generation or texture synthesis.

4.2.3. Wide-Sense Markov Random Fields

Wide-sense Markov random fields are a direct generalization of wide-sense Markov Chains. They are also known as Autoregressive Models (ARM) [New77]. In these models, the field is defined in terms of a linear minimum mean squared-error (MMSE) estimator. Using the definition for $X_{a,b}$ given in the Section above, we may form a linear estimate of $f_{a,b}$ as:

$$\hat{f}_{a,b} = \sum_{f_{i,j} \in X_{a,b}} c_{ij}\, f_{i,j} \qquad (4.8)$$

If we denote the mean squared-error to be:

$$e_{a,b} = E\left\{[f_{a,b} - \hat{f}_{a,b}]^2\right\} \qquad (4.9)$$

then it can be shown [Hab72] that the following set of constraint equations must be satisfied by the coefficients, c_{ij}:

$$E\left\{[f_{a,b} - \hat{f}_{a,b}]\, f_{i,j}\right\} = 0 \qquad \text{all } f_{i,j} \in X_{a,b} \qquad (4.10)$$

This is the same as viewing the estimate as the orthogonal projection of $f_{a,b}$ onto the Hilbert Space of the "past" observations, $X_{a,b}$.

Denoting the subset,

$$P_{a,b} = \left[f_{a-1,b}\ ,\ f_{a-1,b-1}\ ,\ f_{a,b-1}\right] \subset X_{a,b} \qquad (4.11)$$

a random field is defined to be wide-sense Markov if:

$$\hat{f}_{a,b} = \sum_{f_{i,j} \in P_{a,b}} c_{ij}\, f_{i,j} \qquad (4.12)$$

i.e. the MMSE estimator of $f_{a,b}$ given all of $X_{a,b}$ is the same as that given only the three immediate neighbors of $f_{a,b}$ to the left and above. A generalization of this to higher orders is given by Newman [New77]:

$$\hat{f}_{a,b} = \sum_{i=0}^{Q} \sum_{j=0}^{R} c_{ij} f_{i,j} \qquad \textbf{(4.13)}$$

It can be shown that Equation (4.12) is equivalent to a difference equation in $f_{i,j}$:

Theorem [Ros76]. A discrete random field f is wide-sense Markov iff it satisfies the difference equation:

$$f_{m,n} - \sum_{(i,j)\in P_{m,n}} c_{ij} f_{i,j} = \xi_{m,n} \qquad (4.14)$$

where the $\xi_{m,n}$ are uncorrelated,

$$E\left\{\xi_{m,n}\xi_{p,q}\right\} = 0 \quad ; m \neq p \;\; or \;\; n \neq q$$

It may be noted that fields with a separable autocorrelation function of the form $R(k,l) = \sigma^2 e^{-\alpha_1|k| - \alpha_2|l|}$ are wide-sense Markov [Hab72].

This model therefore allows for the generation of a sample function recursively. Furthermore, the model can be fitted to a given image by solving Equation (4.10) for the c_{ij}'s. This involves the inversion of a block Toeplitz autocorrelation matrix as in the one- dimensional autoregressive (AR) model [New77].

The spectrum of a general wide-sense Markov random field given by Equation (4.13) is:

$$S_f(u,v) = \left| \sum_{k=0}^{Q} \sum_{l=0}^{R} \gamma_{k,l}\, e^{-2\pi j(ku+lv)} \right|^{-2} \quad -0.5 \leq u,v \leq 0.5 \qquad (4.15)$$

4.2.4. Discrete Gauss-Markov Fields

Using an altogether different interpretation of "past", "present", and "future" in two-dimensions, Woods [Woo72] has proposed the following model. We think of a field as being divided into two regions G^+ (the future) and G^- (the past) by a band of minimum width p called ∂G (the present). Figure 4.1 is one example of such a division. The field is said to be Gauss-Markov-p if:

$$Pr\left(f_{G^+} \mid f_{\partial G}, f_{G^-}\right) = Pr\left(f_{G^+} \mid f_{\partial G}\right) \tag{4.16}$$

If one now defines the set of indices:

$$D_p = \left\{ (k,l) \mid k^2+l^2 \leq p^2 \; ; \; (k,l) \neq (0,0) \right\} \tag{4.17}$$

Woods shows that the definition given above is equivalent (for the case of Gaussian, homogeneous, zero mean random fields) to the field generated by the following interpolative difference equation:

$$f_{i,j} = \sum_{D_p} h_{kl} \, f_{i-k,j-l} + u_{ij} \tag{4.18}$$

where:

(1) $E\left\{ f_{i,j} \, u_{k,l} \right\} = c \, \delta_{ik} \, \delta_{jl} \quad , \; c > 0.$

(2) u_{ij} is a homogeneous, zero mean, Gaussian random field with

$$R_u(k,l) = \left\{ \begin{array}{ll} -ch_{k,l} & , (k,l) \in D_p \\ c & , (k,l) = (0,0) \\ 0 & , \textit{elsewhere} \end{array} \right\}$$

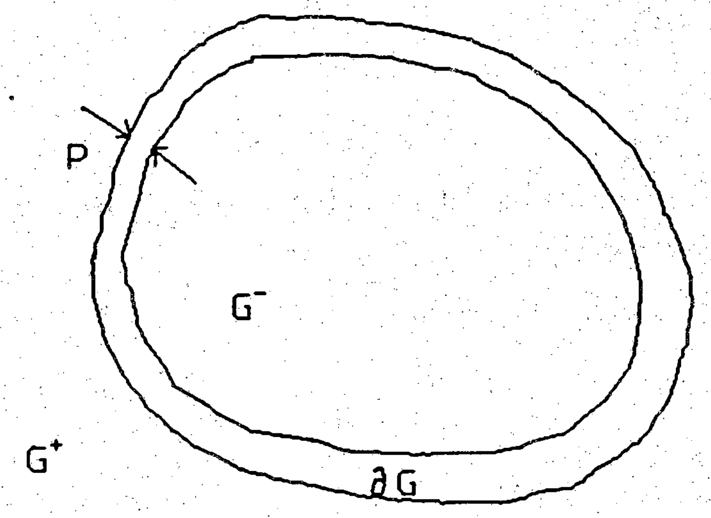

Figure 4.1: An example of a Gauss-Markov-p random field.

(3) h_{kl}'s are the coefficients of the linear MMSE estimator:

$$E\left\{f_{a,b} \mid X_{M,N}^{a,b}\right\} = \sum_{D_p} h_{kl}\, f_{a-k,b-l} \tag{4.19}$$

It is easy to see from Equation (4.18) and property (1) above that the autocorrelation of the field f satisfies:

$$R_f(k,l) = \sum_{D_p} h_{ij}\, R_f(k-i,l-j) + c\,\delta_{kl} \tag{4.20}$$

so that the spectrum is:

$$S_f(u,v) = \frac{c}{1 - \sum_{D_p} h_{kl}\, e^{2\pi j(ku+lv)}} \qquad -0.5 \leq u,v \leq 0.5 \tag{4.21}$$

4.2.5. Maximum Entropy Fields

A fourth method of modeling a two-dimensional random field is by a maximum entropy spectrum. This model has its roots in the problem of two-dimensional spectral estimation [New77].

The method of estimating power spectra in one dimension by maximizing entropy was first introduced by Burg [Bar69] and made rigorous by Edwards and Fitelson [Edw73]. This has been expanded to the two-dimensional case by Barnard and Burg [Bar69], Newmann [New77], and Woods [Woo76]. The idea is to determine the spectrum $S_f(u,v)$ of a field f from a small set of known (or estimated) autocorrelation values $R_f(k,l)$, $|k| < Q$ $|l| < R$. The method is to maximize the entropy of the field,

$$H = \frac{1}{2} \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} \ln \left[S_f(u,v) \right] du \; dv \quad + \quad \ln \left[2\pi e \right]^{\frac{1}{2}} \qquad (4.22)$$

(for Gaussian fields) consistent with the known autocorrelation values:

$$R_f(k,l) = \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} S_f(u,v) \, e^{-2\pi j(ku+lv)} \, du \; dv \qquad ; \begin{matrix} |k| < Q \\ |l| < R \end{matrix} \qquad (4.23)$$

Upon solving the resulting variational problem for $S_f(u,v)$, the following form is obtained, where the coefficients c_{mn} depend on the values for $R_f(k,l)$.

$$S_f(u,v) = \left[\sum_{m=-Q}^{Q} \sum_{n=-R}^{R} c_{mn} \; e^{-2\pi j(ku+lv)} \right]^{-1} \qquad -0.5 \leq u,v \leq 0.5 \quad (4.24)$$

It turns out that, just as the maximum entropy method and autoregressive models are related in one dimension, there is a similar connection in two-dimensions [New77]. Furthermore, the ME spectrum is also the spectrum of a Gauss-Markov random field [Woo76].

4.3. Paths in Two-Dimensional Fields

4.3.1. Ordering

One of the key difficulties shown in the previous Section is that of generalizing the one-dimensional notions of "past", "present", and "future" to two-dimensions. Different assumptions on this point have led to rather different random field models. It is a matter of argument as to which is the more "natural" generalization [Has80].

The imposition of path structures on a two-dimensional random field removes this ambiguity, however, at least as far as the random variables along such paths are concerned. Recall that a path was defined in Section 2.2.2 as an *ordered* sequence of random field nodes satisfying certain connectivity requirements. Let us denote the random field variables corresponding to a given path, $\mathbf{m} = [\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_n]$ as: $\mathbf{f} = [f_{\vec{r}_1}, \ldots, f_{\vec{r}_n}] = [f_1, \ldots, f_n]$. We then have the following obvious but important observation regarding the random field variables associated with this path.

Observation: The path $\mathbf{m}$ imposes an ordering on the variables $\{f_{\vec{r}_i}\}$:

$$f_{\vec{r}_1} < f_{\vec{r}_2} < \cdots < f_{\vec{r}_n} \qquad (4.25)$$

It is again natural to think of a *time series* in the observations with the terms "past", "present", and "future" possessing some meaning.

It will later become necessary to restrict our attention to a class of paths among all possible paths defined earlier. For this class, the nature of the ordering imposed by the path can be made more concrete. Consider pairs of nodes of the lattice I^2,

$$(\vec{r}_i\ ,\ \vec{r}_j) \ \in \ I^2 \times I^2 \qquad (4.26)$$

and some undefined root node $\vec{r}_0$. We denote by "$<$" the binary relation "distance from $\vec{r}_0$". Thus, $\vec{r}_i < \vec{r}_j$ if the Euclidean distance, $|\vec{r}_i - \vec{r}_0| < |\vec{r}_j - \vec{r}_0|$. The relation $<$ is a *partial order* [Bob74] on the set of lattice nodes $\{\vec{r}_i\}$, since $<$ is reflexive, antisymmetric, and transitive. If we now restrict the set of all paths with root node $\vec{r}_0$ to the set $Q_{\vec{r}_0}$ given by:

$$Q_{\vec{r}_0} = \left\{ \mathbf{m} \mid \vec{r}_i < \vec{r}_j \quad \text{for } all \quad i < j \right\} \tag{4.27}$$

then the set of nodes $\{\vec{r}_i\}$ comprising a path $\mathbf{m} \in Q_{\vec{r}_0}$ is *totally ordered* or *linearly ordered*. Finally, since there is an isomorphism between nodes of a path and the random field observations along that path, we see that the path $\mathbf{m}$ imposes a linear ordering on the set $\mathbf{f}$:

$$f_1 < f_2 < \cdots < f_n \qquad \mathbf{f} \in Q \tag{4.28}$$

where $\mathbf{f} \in Q$ is understood to mean the corresponding $\mathbf{m} \in Q_{\vec{r}_0}$ for some $\vec{r}_0$. To illustrate, Figure 4.2 shows that the set Q is merely those paths that do not "double back on themselves".

4.3.2. Dynamic Modeling of Path Observations

The random field observations along a given path in Q form a linearly ordered set. This fact suggests the use of a dynamic model for the variables. A well known class of such models is known variously in the literature as Linear Predictive, Autoregressive Moving Average (ARMA), or Predictive Discrete Wiener Filter models (see [Mak75] or [Kai74] for a general description). All of these models exhibit the wide-sense Markov property (in the classical one-dimensional sense) that the best prediction of the next observation in a time series is a linear function of the past p observations. Here "best" is in terms of minimizing the mean squared error. The orthogonality principle states that this error between the actual value of the next observation and the predicted value is orthogonal to the past observations.

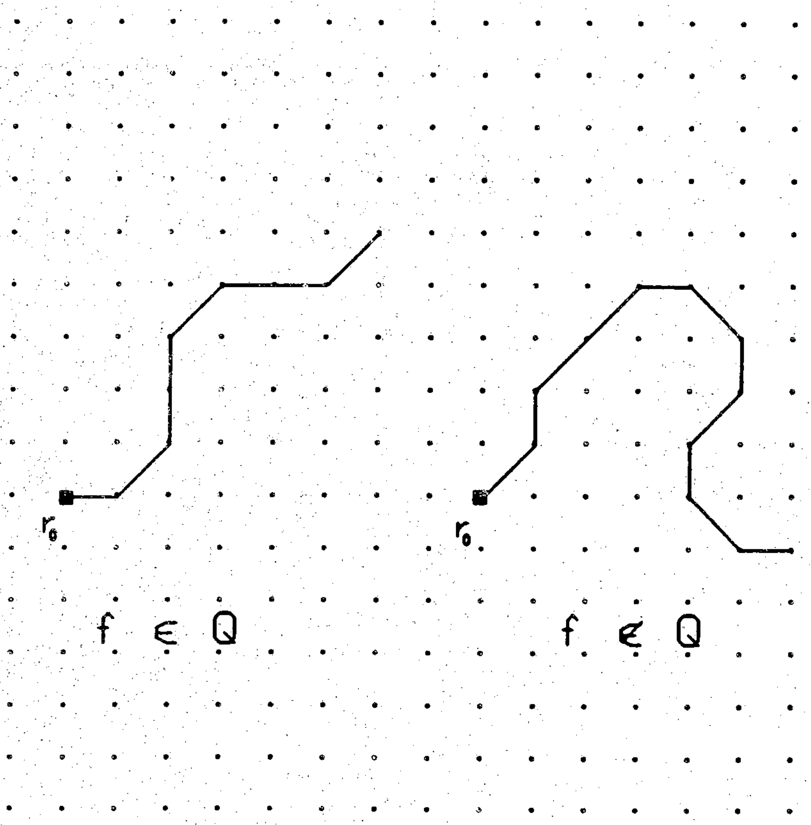

Figure 4.2: The path topology on the left is a member of the set Q but the one on the right is not. The right path loops back on itself so that, for example, the ninth node is closer to the root node than is the fifth node.

Adopting the ARMA model, the above assumptions may be stated in the form:

$$f_i - \sum_{k=1}^{p} \phi_k \, f_{i-k} = \xi_i - \sum_{l=1}^{q} \theta_l \, \xi_{i-l} \tag{4.29}$$

where the ϕ_k are the p autoregression coefficients and the θ_l are the q moving average coefficients. The process $\{\xi_i\}$, known as the *input* process or the *innovations* process is a sequence of orthogonal random variables:

$$E\left\{\xi_i \, \xi_j\right\} = \sigma_\xi^2 \, \delta_{ij} \tag{4.30}$$

Thus, the sequence of observations $\{f_i\}$ may be viewed as the output of a linear filter with transfer function

$$H(z) = \frac{1 + \sum_{l=1}^{q} \theta_l \, z^{-l}}{1 - \sum_{k=1}^{p} \phi_k \, z^{-k}} \tag{4.31}$$

and with input $\{\xi_i\}$.

4.3.3. Application to SEL

We discuss the important question of when the observations really do fit the model given above in the next Section, but here we are interested in how such a model may be exploited.

Let us assume the process $\{f_i\}$ is Gaussian and wide sense stationary with mean $\bar{f}$. We will denote by:

$$\left\{\tilde{f}_i\right\} : \quad \tilde{f}_j = f_j - \bar{f} \tag{4.32}$$

i.e. the zero mean process obtained from $\{f_i\}$ by subtracting $\bar{f}$. We further assume that $\{\xi_i\}$ is a zero mean, Gaussian, wide-sense stationary, orthogonal process such that:

$$\tilde{f}_i - \sum_{k=1}^{p} \phi_k \tilde{f}_{i-k} = \xi_i - \sum_{l=1}^{q} \theta_l \xi_{i-l} \tag{4.33}$$

Thus:

$$f_i - \bar{f} - \sum_{k=1}^{p} \phi_k \tilde{f}_{i-k} = \xi_i - \sum_{l=1}^{q} \theta_l \xi_{i-l} \tag{4.34}$$

Or:

$$f_i - \sum_{k=1}^{p} \phi_k \tilde{f}_{i-k} - \sum_{l=1}^{q} \theta_l \xi_{i-l} = \xi_i + \bar{f} \tag{4.35}$$

Denoting the left hand side of (4.35) by y_i, we have:

$$y_i = \xi_i + \bar{f} \tag{4.36}$$

Assume that at time i the previous q values of $\left\{y_j\right\}_{j=i-1}^{i-q}$ are known. Then, since the present observation, f_i, is known, the p previous observations are known (and therefore the $\{\tilde{f}_{i-k}\}$), and the q previous ξ_{i-l} values are known ($\xi_{i-l} = y_{i-l} - \bar{f}$), the value of y_i is known. This is then used to find ξ_i. In this way, the sequence $\{y_i\}$ may be recursively generated from the observations $\{f_i\}$.

Now, the random variables ξ_i are orthogonal and zero mean. Thus:

$$E\left\{y_i\, y_j\right\} = E\left\{\xi_i\, \xi_j\right\} + \overline{f}^2 = \overline{f}^2 = \left[E\left\{y_i\right\}\right]^2 \qquad (4.37)$$

so the y_i are uncorrelated. Since they are linear combinations of Gaussian random variables, they are also Gaussian and therefore independent. If we then define the p_0 and p_1 measures of Section 2.2.3 on these *virtual observations* instead of the actual observations f_i :

$$\begin{aligned} p_0(y_i) &= Pr\,(y_i \mid H_0) = Pr\,(y_i \mid y_i \textit{ is on a random path}) \\ p_1(y_i) &= Pr\,(y_i \mid H_1) = Pr\,(y_i \mid y_i \textit{ is on an edge path}) \end{aligned} \qquad (4.38)$$

then we may define the path branch metric in terms of the y_i also. Since the y_i are independent the important assumption of Equation (2.17) is true. The results obtained in Chapter 2 and 3 are therefore also valid, even though the original field f is correlated.

In short, imposing an ARMA model on the sequence of correlated observations comprising a path allows one to generate a sequence of independent virtual observations from the original sequence with a linear, causal, finite memory filter. This new sequence is used to calculate the metric for the path. Of course to do so, the searching algorithm must now keep a record of not only every path and its metric visited, but also the *state* of the observation process for each path. For an ARMA(p ,q) model, this state is composed of the last p variables $\{\tilde{f}_{i-k}\}$ and the last q variables $\{\xi_{i-l}\}$. This may add considerably to the memory requirements of the searching algorithm.

4.4. The D-SIM Random Field

It should be clear from the Section above that it is desirable to be able to model the sequence of observations along a path in a correlated random field as an ARMA process. It is not clear if or when such a model is appropriate or realistic. In this Section, we will propose a random field for which the model can be shown to explicitly hold. That is, we are able to specify a two-dimensional autocorrelation function such that the one-dimensional sequence of observations along any path in the class Q obeys a difference equation of the form shown in Equation (4.29).

4.4.1. Difficulties with Previous Models

We return to the various two-dimensional random field models summarized in Section 4.2. It is easy to construct counter-examples or arguments to show that, in general, these fields fail in the sense discussed above.

4.4.1.1. Wide-Sense Markov Fields

Recall that wide-sense Markov random fields satisfy:

$$E\left\{[f_{a,b} - \hat{f}_{a,b}]f_{i,j}\right\} = 0 \qquad \textit{all } f_{i,j} \in X_{a,b} \tag{4.10}$$

Let us consider the specific case of a zero mean, unit variance random field with autocorrelation function:

$$R_f(k,l) = e^{-a_1|k| - a_2|l|} \tag{4.39}$$

It can easily be shown [Ros76] that the Equation (4.10) is satisfied with the MMSE estimator given by:

$$\hat{f}_{m,n} = e^{-a_1} f_{m-1,n} + e^{-a_2} f_{m,n-1} - e^{-a_1-a_2} f_{m-1,n-1} \tag{4.40}$$

so this field is wide-sense Markov. Now consider the path $\mathbf{f} \in Q$ depicted in Figure 4.3 which is horizontal up to observation f_s and diagonal beyond. Before node s, the autocorrelation function of the observations along $\mathbf{f}$ satisfy:

$$R(k) - e^{-a_1} R(k-1) = 0 \tag{4.41}$$

Therefore:

$$E\left\{[f_i - e^{-a_1} f_{i-1}] f_j\right\} = 0 \qquad j < i < s \tag{4.42}$$

which implies the $\{f_i\}$ satisfy the difference equation:

$$f_i - e^{-a_1} f_{i-1} = \xi_i \qquad , i < s \tag{4.43}$$

$$E\left\{\xi_i \; \xi_j\right\} = 0 \qquad i \neq j$$

After node s, however, we have:

$$R(k) + e^{-a_1-a_2} R(k-1) = 0 \tag{4.44}$$

which leads to:

$$f_i + e^{-a_1-a_2} f_{i-1} = \xi_i \qquad , i > s \tag{4.45}$$

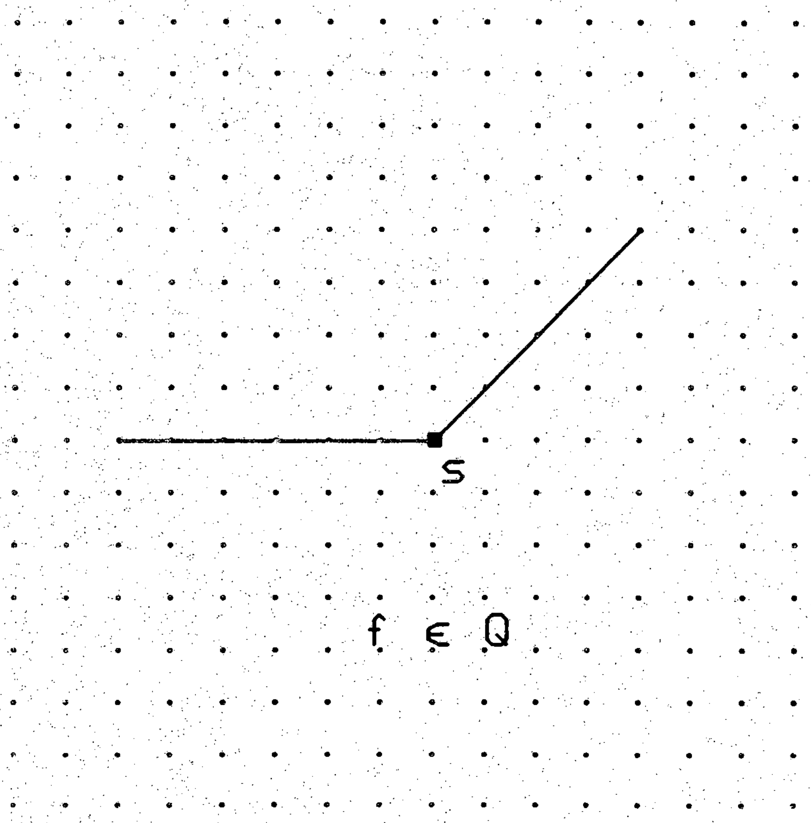

Figure 4.3: A path in the set Q that is horizontal up to node s and diagonal beyond.

$$E\left\{\xi_i\ \xi_j\right\} = 0 \qquad i \neq j$$

We see, therefore, that the second order statistics of the sequence $\{f_i\}$ are different before and after the term $i = s$. Even in this simple case, we see that the sequence of observations is not wide-sense stationary, although the random field from which they were drawn is homogeneous. Furthermore, we cannot even employ a ***locally stationary*** model [Sil57], as one can quickly see by considering paths in Q with many changes in direction (Figure 4.4).

4.4.1.2. Markov Mesh Models

Markov mesh random fields are those for which:

$$Pr\,(f_{a,b} \mid Z_{a,b}) \;=\; Pr\,(f_{a,b} \mid U_{a,b}) \qquad , U_{a,b} \subset X_{a,b}^{a,b} \tag{4.3}$$

The counter example given in the Section above serves for these models as well if we make the additional assumption that the field is Gaussian. This is because for Gaussian random fields, the wide-sense and strict-sense Markov definitions are equivalent [Ros76]. Thus, for the same reasons as before, the random field with autocorrelation function given by Equation (4.39) is an example of a Markov mesh for which a sequence of observations along an arbitrary path in Q does not satisfy Equation (4.29).

4.4.1.3. Maximum Entropy Fields

A maximum entropy (ME) field is defined in terms of its spectrum (Equation (4.24)), where the coefficients, c_{mn}, are Hermitian and determined by the values (or estimates) of the autocorrelation function $R_f\,(k\,,l)$ for $|k| \leq Q \quad |l| \leq R$. Newman

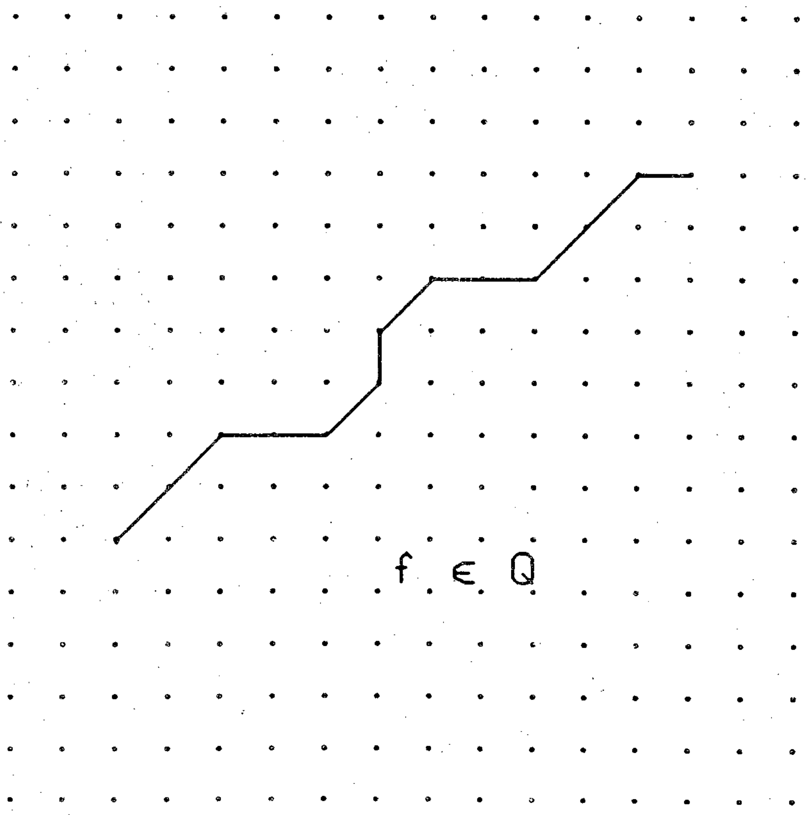

Figure 4.4: A path in the set Q that has many changes in direction.

[New77] has shown, however, that despite the difference in the way ME and ARM models are formulated, their power spectra have the same form. The correspondence between Equations (4.21) and (4.24) can be seen by setting:

$$c_{mn} = \sum_{i=max(0,-m)}^{\min(Q-m,Q)} \sum_{j=max(0,-n)}^{\min(R-n,R)} \gamma_{i+m,j+n} \gamma_{i,j}^{*} \tag{4.46}$$

Now, since we have assumed R_f is integratable, S_f is necessarily bounded and continuous. Also, R_f is bounded. Therefore, R_f is square integrable and so by the Plancherel Theorem, any two fields with the same spectrum necessarily have an identical autocorrelation function R_f. This along with Equation (4.46) implies that there exists an ME field with R_f given by Equation (4.39). So once again, this same R_f serves as a counter example to the claim that ME fields give rise to paths whose observations satisfy Equation (4.29).

4.4.1.4. Gauss-Markov Fields

For a counter example for Gauss-Markov fields, we turn to the original example of Woods [Woo72]. In this case the field is generated by the difference equation:

$$f_{i,j} = \alpha(f_{i,j+1} + f_{i+1,j} + f_{i,j-1} + f_{i-1,j}) + u_{i,j} \tag{4.47}$$

where $\{u_{i,j}\}$ is a zero mean Gaussian field and

$R_u(k,l) = \begin{Bmatrix} c & (k,l) = (0,0) \\ -c\,\alpha & k = 1,-1 \;\;, l = 1,-1 \\ 0 & elsewhere \end{Bmatrix}$ $|\alpha| < \frac{1}{4}$. Woods shows that the autocorrelation function for this field is:

$$R_f(k,l) \;=\; c \int_{-0.5}^{0.5} \int_{-0.5}^{0.5} \frac{e^{j2\pi(uk + vl)}}{1 - 2\alpha(\cos 2\pi u + \cos 2\pi v)} \, du \, dv \qquad (4.48)$$

Woods evaluated this integral numerically for $\alpha = 0.225$. Values out to $k = l = 4$ are listed in his Table I. It is clear from these values that, just as in the example of the Sections above, observations along diagonal paths have different second order statistics than do those along horizontal paths. Furthermore, in this example even straight paths do not have a sequence of observations whose autocorrelation values decay exponentially. This is a necessary condition for them to obey a difference equation.

4.4.2. The D-SIM Field

We have seen in Section 4.4.1 that the classical random field models do not give rise to paths whose observations satisfy Equation (4.29). We might go ahead and model the observations as a one-dimensional Markov Chain anyway, but it is unclear how to estimate the ARMA parameters ϕ_k and θ_l from $R_f(k,l)$. A more satisfactory solution is to find a random field model for which Equation (4.29) *does* hold, fit this model to the image and obtain the ϕ_k and θ_l parameters from the model.

In this Section we describe such a random field. It will be called a Discrete-Step Isotropic Markov (D-SIM) random field, and is defined directly in terms of its autocorrelation function. Here we will assume the field f is homogeneous, zero mean, and Gaussian.

4.4.2.1. Necessary Conditions on the Autocorrelation Function

The counter examples given in Section 4.4.1 provide some insight into two conditions that R_f must satisfy for the field to be D-SIM. The first is that the field must be isotropic. Roughly speaking, this means that $R_f\,(k,l)$ should be independent of direction and depend only on the separation of the variables. In most discussions of random fields, this condition is stated:

$$R_f\,(k,l) \;=\; R_f\,([k^2+l^2]^{\frac{1}{2}}) \;=\; R_f\,(r) \tag{4.49}$$

i.e. the autocorrelation function is a function of one variable, the Euclidean distance between the nodes. This concept of isotropy is furthermore intuitively satisfying. Given no other a priori information about an image, why should one expect the observations to be correlated any more or less in one direction than another?

The second necessary condition is that along any straight path (really *any* path, $\mathbf{f} \in Q$), the autocorrelation function of the variables should decay exponentially or as the sum of exponentials. Further, this exponential decay must be in *discrete steps*, i.e. it must be *geometrical*. This is necessary in order that the observation be modeled by a difference equation.

The simple and familiar isotropic autocorrelation function of Equation (4.49) does not meet this second condition. But Euclidean distance, while being possibly the most natural, is not the only way to measure distance on a square lattice. In particular, another distance function on I^2 which suggests itself in view of condition two is:

$$r(k,l) \;=\; |k| \;+\; |l| \tag{4.50}$$

known as the *city block* distance [Ros68]. We note in passing that city block distance, like Euclidean distance, its a metric in the topological sense [Nay82].

4.4.2.2. D-SIM Fields

We now define a p^{th}-order D-SIM random field as a homogeneous field with an autocorrelation function of the form:

$$R_f(k,l) = \sigma^2 \left[c_1 e^{-a_1(|k|+|l|)} + \ldots + c_p e^{-a_p(|k|+|l|)} \right] \quad (4.51)$$

where $\sigma^2 = R_f(0,0)$ and $\sum_{i=1}^{p} c_i = 1$. It is seen from Equation (4.52) that this autocorrelation function is nothing more than the autocorrelation function of a one-dimensional p^{th}-order wide-sense Markov sequence, with city block (two-dimensional) distance taking the place of (one-dimensional) integer lags.

A random field being D-SIM is a sufficient condition for paths in the class Q to satisfy an equation of the form of Equation (4.29). Recall that if $\mathbf{f} \in Q$, then there exists a root node, $\vec{r}_0$, such that:

$$|\vec{r}_i - \vec{r}_0| < |\vec{r}_j - \vec{r}_0| \qquad , all\ i < j \quad (4.52)$$

We remark that, although this definition was formulated using Euclidean distance as the distance function, we obtain precisely the same set of paths Q using city block distance in Equation (4.27), since both are topological metrics. Before we can prove the above assertion, we need the following theorem:

Theorem 4.1: A path $\mathbf{f} \in Q$ in a p^{th}-order D-SIM random field induces a super sequence $\mathbf{f}'$ such that Equation (4.29) holds for this $\mathbf{f}'$.

Proof:

Let $\mathbf{f} \in Q$ be:

$$\mathbf{f} = [f_{\vec{r}_1}, \ldots, f_{\vec{r}_i}, \ldots] = [f_1, \ldots, f_i, \ldots] \tag{4.53}$$

The problem encountered in dealing with $\mathbf{f}$ directly is that for some j's, $| r_{j+1} - r_j | = 1$ and for other j's, $| r_{j+1} - r_j | = 2$ depending on whether the path is locally running horizontally, vertically, or diagonally. Where the path is locally diagonal, say from f_j to f_{j+1}, we introduce a dummy variable $\tilde{f}_j$ such that:

$$| \tilde{r}_j - r_j | = 1 \quad \text{and} \quad | r_{j+1} - \tilde{r}_j | = 1 \tag{4.54}$$

and

$$r_j < \tilde{r}_j < r_{j+1} \tag{4.55}$$

(see Figure 4.5). We note that on a square lattice, if $\mathbf{f} \in Q$, this can always be done while preserving the ordering (4.55). We define the sequence $\mathbf{f}'$ to be just $\mathbf{f}$ with the ordered inclusion of the dummy variables $\tilde{f}_j$:

$$\begin{aligned} \mathbf{f}' &= [f_1, \ldots, f_j, \tilde{f}_j, f_{j+1}, \ldots] \\ &= [f'_1, \ldots, f'_i, \ldots] \end{aligned} \tag{4.56}$$

Because of (4.55), this sequence is also an element of Q. Furthermore, the distance between every variable of $\mathbf{f}'$ is, by construction, one. Now, since the field is D-SIM, we have:

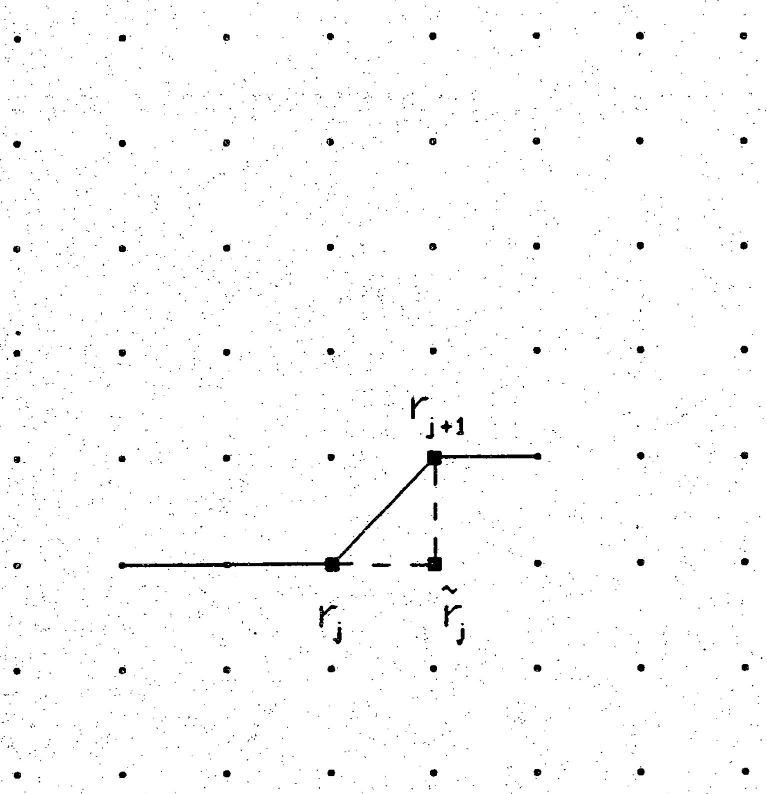

Figure 4.5: Where $\mathbf{f}$ is locally diagonal, a dummy observation $\tilde{f}_j$ is inserted in the sequence $\mathbf{f}'$ such that Equations (4.55) and (4.54) are satisfied.

$$E\left\{f'_j\, f'_{j+K}\right\} = \sigma^2\left[c_1 e^{-a_1|k|} + \ldots + c_p e^{-a_p|k|}\right] \qquad (4.57)$$

This autocorrelation function of the sequence f' is well known in time series analysis, Weiner Filter theory, etc. Since it is real, even, integrable, and positive definite, its Fourier Transform, $S_{f'}(u)$, is real, even, positive, and integrable. It may be expressed in the form:

$$S_{f'}(u) = \frac{A(u^2)}{B(u^2)} \qquad ; -0.5 \leq u \leq 0.5 \qquad (4.58)$$

That is, autocorrelation functions of the form shown in Equation (4.57) correspond to power spectra that can be expressed as ratios of polynomials in u^2. The polynomial $B(u^2)$ is of degree p (from Equation (4.57)) and the degree of A is less than p. Let degree(A) $= q < p$. Because A and B are polynomials in u^2, $S_{f'}(u)$ may be factored into two terms:

$$S_{f'}(u) = \frac{C(u)}{D(u)} \cdot \frac{C^*(u)}{D^*(u)} = H(u) \cdot H^*(u) \quad ; -0.5 \leq u \leq 0.5 \ (4.59)$$

where $H(u)$ and $H^*(u)$ are defined implicitly. Furthermore, this factoring may be accomplished in such a way that all of the complex poles and zeros of $H(u)$ lie in the upper half of the complex plane. Thus $H(u)$ is analytic in the lower half plane. While we have not proven the statements leading to (4.58) and (4.59), they are well known results. The interested reader is referred to [Dav58], [Ire], [Wol38], or other discussions of Weiner theory.

Using the change of variables, $z = e^{j2\pi u}$, Equation (4.59) becomes:

$$S_{\mathbf{f'}}(z) = H(z) \cdot H^*(z) = |H(z)|^2 \qquad (4.60)$$

where $H(z)$ is rational in z and analytic on and outside the unit circle. Therefore, the sequence $\mathbf{f'}$ can be thought of as the output of a filter $H(z)$ driven by white noise. Since all the poles of $H(z)$ are inside the unit circle, and all its zeros are on or inside the unit circle, $H(z)$ is *causal and minimum phase.* $H(z)$ is of the form:

$$H(z) = \frac{\sum_{l=0}^{q} \theta_l z^{-l}}{\sum_{k=0}^{p} \phi_k z^{-k}} \qquad (4.61)$$

so that $\mathbf{f'}$ satisfies:

$$f'_i - \sum_{k=1}^{p} \phi_k f'_{i-k} = \xi_i - \sum_{l=1}^{q} \theta_l \xi_{i-l} \qquad (4.62)$$

$$E\left\{\xi_i \xi_j\right\} = \sigma^2 \delta_{ij} \qquad (4.63)$$

which is what was to be proved.

This theorem tells us that the observations along a path $\mathbf{f} \in Q$ in a D-SIM field are a subsequence of a sequence $\mathbf{f'} \in Q$ which is ARMA. We now show that $\mathbf{f}$ itself possesses an ARMA-like quality.

In the following, we require some new notation. Let f_i be a variable in the sequence $\mathbf{f} \in Q$. Let the induced super-sequence be $\mathbf{f'}$. We will denote by $f'_{(i)}$ the variable $f'_j \in \mathbf{f'}$ corresponding to f_i. That is, $(i) = j \mid f'_j = f_i$.

Theorem 4.2: With $\mathbf{f}$ as in Theorem 4.1, it is possible to recursively construct a sequence $\mathbf{f}'$ from past values of $\mathbf{f}$ such that:

$$\hat{f}_i = \sum_{k=1}^{p} \phi_k \, f'_{(i)-k} + \sum_{l=1}^{q} \theta_l \, \xi_{(i)-l} \tag{4.64}$$

$$f_i - \hat{f}_i = \xi_{(i)}$$

$$E\left\{\xi_{(i)}\,\xi_{(j)}\right\} = \sigma_\xi^2 \, \delta_{(i)(j)}$$

Proof:

If there are no diagonal segments in the path $\mathbf{f}$, then $\mathbf{f} = \mathbf{f}'$ and the Theorem follows directly from Equation (4.62).

If there are diagonal segments in $\mathbf{f}$, then $\mathbf{f} \neq \mathbf{f}'$. Since $\mathbf{f}$ is a subsequence of $\mathbf{f}'$, every element of $\mathbf{f}$ has a corresponding element in $\mathbf{f}'$: $f_i = f'_{(i)}$, but not vice versa. We fill in the dummy variables $\tilde{f}$ in the following manner. We know from Equation (4.62) that:

$$f'_j - \sum_{k=1}^{p} \phi_k \, f'_{j-k} + \sum_{l=1}^{q} \theta_l \, \xi_{j-l} = \xi_j \tag{4.65}$$

Using vector notation, we will denote:

$$\begin{aligned}
\Psi_j &= \left[f'_{j}, \ldots, f'_{j-p+1}, \xi_j, \ldots, \xi_{j-q+1} \right]^T \\
&= \left[\mathbf{F}_j^T \vdots \Xi_j^T \right] \\
\mathbf{F}_j &= \left[f'_{j}, \ldots, f'_{j-p+1} \right]^T \\
\Xi_j &= \left[\xi_j, \ldots, \xi_{j-q+1} \right]^T \\
\Phi &= \left[\phi_1, \ldots, \phi_p \right]^T \\
\Theta &= \left[-\theta_1, \ldots, -\theta_q \right]^T
\end{aligned} \qquad (4.66)$$

Equation (4.65) can then be written (with k = j-1):

$$\xi_{k+1} = f'_{k+1} - \Phi \mathbf{F}_k - \Theta \Xi_k \qquad (4.67)$$

or:

$$\xi_{k+1} = f'_{k+1} - \left[\Phi^T \vdots \Theta^T \right] \Psi_k$$

with:

$$\begin{aligned}
\mathbf{F}_{k+1} &= \left[I \vdots 0 \right] \begin{bmatrix} f'_{k+1} \\ \cdots \\ \mathbf{F}_k \end{bmatrix} \qquad I \text{ is } p \times p \\
\Xi_{k+1} &= \left[I \vdots 0 \right] \begin{bmatrix} \xi_{k+1} \\ \cdots \\ \Xi_k \end{bmatrix} \qquad I \text{ is } q \times q
\end{aligned} \qquad (4.68)$$

Note that we have not employed the usual notation for state space equations here in order to show explicitly where the ξ_k come from.

Suppose that up to time i the state vector $\Psi_{(i)}$ is known. (We will assume that at time $i=0$, the state $\Psi_{(0)}$ is fixed and known.) If the next step along the path $\mathbf{f}$ from f_i to f_{i+1} is of length 1, then $f'_{(i)+1} = f'_{(i+1)} = f_{i+1}$. Therefore, the new innovation $\xi_{(i)+1}$ may be obtained from Equation (4.67) with $f'_{(i)+1}$ equal to the new observation f_{i+1}:

$$\xi_{(i)+1} = \xi_{(i+1)} = f_{i+1} - \left[\Phi^T \vdots \Theta^T\right] \Psi_{(i)} \tag{4.69}$$

The state vector $\Psi_{(i)}$ may be updated by Equations (4.68) with $f'_{(i)+1} = f_{i+1}$ and $\xi_{(i)+1}$ as above.

On the other hand, if the next step along $\mathbf{f}$ is of length 2, then $f'_{(i)+1} \neq f_{i+1}$. This is an occasion when a dummy variable $f'_{(i)+1} = \tilde{f}$ needs to be inserted in the process $\mathbf{f'}$. We *choose* $f'_{(i)+1}$ so that:

$$f'_{(i)+1} = \left[\Phi^T \vdots \Theta^T\right] \Psi_{(i)} + \tilde{\xi} \tag{4.70}$$

where $\tilde{\xi}$ is randomly chosen, independently of the ξ_i's, from a Gaussian distribution with mean zero and variance σ_ξ^2. Having obtained a dummy "observation" in this manner we find:

$$\begin{aligned} \xi_{(i)+1} &= f'_{(i)+1} - \left[\Phi^T \vdots \Theta^T\right] \Psi_{(i)} \\ &= \tilde{\xi} \end{aligned} \tag{4.71}$$

and we can update $\Psi_{(i)}$ as before.

This procedure generates a sequence $\{\xi_k\}$ which is uncorrelated. Furthermore, the sequences $\mathbf{f}$ and $\mathbf{f'}$ are identical up to the dummy observations, i.e. $f'_{(i)} = f_i$. Finally, $\mathbf{f'}$ obeys Equation (4.65). Thus:

$$f_i - \sum_{k=1}^{p} \phi_k \, f'_{(i)-k} + \sum_{l=1}^{q} \theta_l \, \xi_{(i)-l} = \xi_{(i)} \tag{4.72}$$

$$E\left\{\xi_{(i)}\,\xi_{(j)}\right\} = \sigma_\xi^2 \, \delta_{(i)(j)}$$

which completes the theorem.

In summary, Theorem 4.2 states the following. If a random field is D-SIM, then paths of the class Q have an ARMA-like structure. That is, it is always possible to recursively construct a super sequence $\mathbf{f}'$ from $\mathbf{f}$ such that Equation (4.72) generates a sequence of uncorrelated innovations. Where the path branches along $\mathbf{f}$ are of length 1, the f'_k are precisely the path observations $f'_{(i)} = f_i$. Otherwise, the dummy or "missing" observations may be filled in using past values by Equation (4.70). In this fashion, the projection of the next observation at time i, (f_i), on all the previous observations, $\{f_k \;\; k<i\}$, is always a linear function of the past p elements of $\mathbf{f}'$, $\{f'_k \;\; k<(i)\}$, and the past q elements of ξ, $\{\xi_k \;\; k<(i)\}$.

It is the special nature of the autocorrelation function of a D-SIM field that allows us to recursively generate a sequence of uncorrelated innovations $\{\xi_{(i)}\}$ from the path observations $\{f_i\}$. These in turn may be used in the calculation of branch metrics as discussed in Section 4.3.3. Since they are uncorrelated and Gaussian and therefore independent, the log-likelihood of a long path metric may be expressed as the sum of the log-likelihoods of the individual branches comprising the path, thus allowing recursive computation of the metric. We note that we have not shown *necessity*, only *sufficiency*. That is, it is sufficient that the field be D-SIM to use this procedure, but there may exist other fields that are not D-SIM for which paths of the class Q fit an ARMA model.

CHAPTER 5

EXPERIMENTAL RESULTS

5.1. Implementation Details

In this Chapter, we will discuss some experimental results from one implementation of the SEL algorithm. Our purpose is to experimentally verify the major claims made for the technique in the previous Chapters. These examples, a result of processing a variety of real image data, also provide an indication of the range of applications of the method. Before examining the results themselves, we will discuss some of the implementation details.

All the results discussed in this Chapter are obtained using the Z-J stack algorithm. The other algorithms discussed in Section 2.3 are either refinements on the stack algorithm or have been shown to yield essentially the same results in searching tree structures. Consequently, only the simplest, but not necessarily the fastest or most efficient, algorithm is tested here. The root nodes in these examples are obtained automatically by the algorithm by imposing a high threshold on the gradient magnitude output of the preprocessing edge operator in the manner described in Section 2.4. The first root node selected is simply that node with the highest magnitude value in the image. After the first edge path has been searched by the algorithm, subsequent root nodes are chosen if their associated gradient magnitude exceeds the threshold and they do not lie within some minimum distance of any previously detected edge path. Search termination is effected by one of four conditions: the best path looping on itself, the best path intersecting any previously detected path, the best path reaching an image boundary, or the internal stack filling up.

The Markov path transition probabilities for all cases, except where noted, are those given in the Appendix with the order of the process equal to two (k=2). The measures p_0 and p_1 used in the branch metrics were assumed to be a linear combination of Gaussians with parameters estimated from the magnitude map.

The edge operator preceding the SEL algorithm, except where otherwise noted, is a combination two-dimensional Gaussian low pass filter and gradient operator. The original image is convolved with two directional gradient masks to obtain an estimate of the gradient in the x and y directions. These gradient masks employ a Gaussian shaped window function to reduce the Gibbs effect in their spectral response as discussed in Section 2.2.4. The x operator has the mathematical function:

$$X = \nabla(G * I) \cdot \hat{i}_x \tag{5.1}$$

and the y operator has the function:

$$Y = \nabla(G * I) \cdot \hat{i}_y \tag{5.2}$$

where G is the two dimensional Gaussian, I is the original image intensity function and $\hat{i}_x$, $\hat{i}_y$ are unit vectors in the x and y directions respectively. The X and Y directional gradient estimates are then combined to form an estimate of the gradient magnitude, M, and direction, D by:

$$\begin{aligned} M &= |\nabla (G * I)| \\ &= \left[X^2 + Y^2\right]^{0.5} \\ D &= \frac{\nabla (G * I)}{|\nabla (G * I)|} \\ &= \tan^{-1}\left[\frac{Y}{X}\right] \end{aligned} \tag{5.3}$$

The magnitude and direction maps were both quantized to eight bits, which is the dynamic range of the original images. This gradient operator is functionally the same as the Canny directional derivative operator [Can83]. Of course, non-maximum suppression in the direction of the gradient and hysteresis thresholding are not performed here as they are in that technique. This operator enjoys a high computational efficiency owing to the separability of the Gaussian kernel.

Only the magnitude values enter into the calculation of the branch metrics in the current implementation. The direction map serves two functions. It is used to provide the initial search direction, $\vec{d}_0$, at the root node and to provide the sign of the search direction at subsequent nodes. Since edge contours are searched in a direction *orthogonal* to the gradient direction, two choices are available. This 180° ambiguity is resolved during the search by the direction map. Since root nodes generally lie in the middle of an edge contour, *two* search directions are possible from each root node. The algorithm searches first in the direction such that the gradient vector points to the *right* of the path and then begins again at the same root node searching in the other direction such that the gradient vector points to the *left* of the path. This second search is obviously deleted if the first path closes on itself at the root node. In this way, the entire edge contour on which the root node lies is searched by the algorithm. Furthermore, the gradient direction information ensures that the searching *proceeds* along the edge contour and does not double back in the direction from which it came.

5.2. Performance Comparisons

The first thing we are interested in verifying by experiment is whether or not the SEL algorithm is more successful at detecting edges from the output of an edge operator than are threshold techniques. From the start, our hope has been that sequential detection of edge contours would provide better connected edges, a lower probability of false detection at low signal-to-noise ratios, and freedom from the sensitive problem of threshold selection as compared to threshold classification techniques. We will examine this question from two approaches: using a quantitative performance measure on artificially created images and using qualitative comparisons on some real images.

5.2.1. A Quantitative Performance Measure

As a first cut, we will consider an artificial image with a vertical step edge of known contrast to which has been added zero mean i.i.d. Gaussian noise of known variance. Thus, the signal-to-noise ratio, defined as:

$$SNR \quad = \quad 20 \log_{10} \left[\frac{h}{\sigma} \right] \qquad dB \tag{5.4}$$

where h is the edge contrast and σ the noise variance, is a known parameter. An example of this type of test image is shown in Figure 5.1 at a SNR = 10 dB. Although this image is highly unrealistic in that: 1) the edge is a true step, 2) the two halves of the image have a perfectly flat mean, 3) the contrast is constant over the length of the edge, and 4) the noise is truly additive i.i.d. Gaussian, it is useful for two reasons. One is that the SNR is a known parameter under our control. The second is that it allows the use of a simple but effective quantitative performance measure known as the Pratt figure of merit [Pra78].

The Pratt figure of merit, denoted here by F, is a common experimental tool in the literature for making objective comparisons among edge detectors. It penalizes a detector for both declaring a point that is not on the edge as an edge point and for missing true edge points. Given the test image with a single vertical edge as described above, the edge detector is applied and the resultant edge map is used to compute F as follows:

$$F = \frac{1}{I_M} \sum_{i=1}^{I_D} \frac{1}{1 + \alpha l_i^2} \tag{5.5}$$

where:

$I_M = \max(I_D, I_I)$

I_I = number of ideal edge points

I_D = number of detected edge points

l_i = displacement of the i^{th} detected edge point from the ideal edge.

α = scaling parameter

F ranges in value form 0 to 1.0 (perfect). The merit value for classical edge detectors is a strong function of the classification threshold. In practice, the threshold is varied to obtain the best F value. This iterative optimization must be repeated for each test image (each SNR) desired, as the best threshold for one SNR is generally not optimum

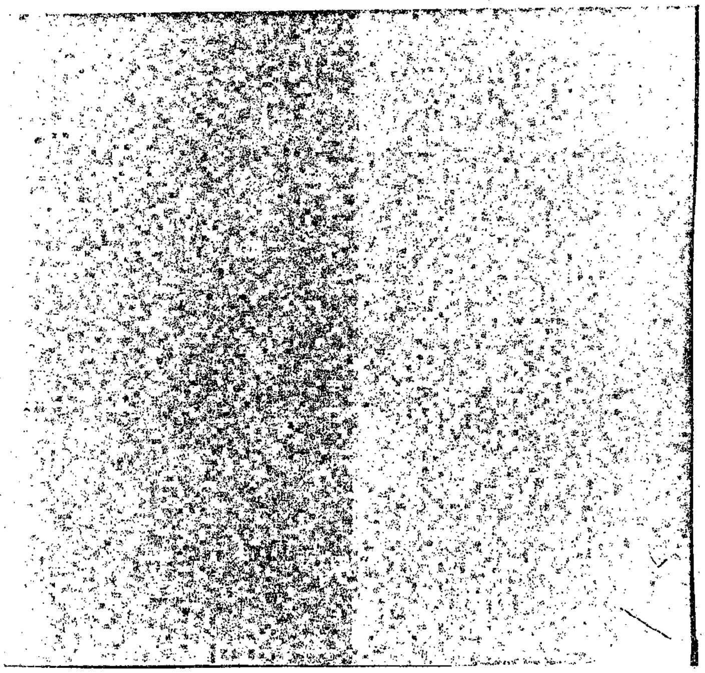

Figure 5.1: A Pratt figure of merit test image at a signal-to-noise ratio of +10 dB.

at others. As an example, Figure 5.2 illustrates the optimum decision thresholds as a function of image SNR for one type of edge detector. Figure 5.3 shows the resulting F value as a function of image SNR.

By comparison, there is no pointwise decision threshold for the SEL algorithm. The threshold based on complete path metrics is extremely robust. In fact, the SEL examples given throughout this Chapter all used the same threshold. This freedom from sensitive thresholds represents one of the distinct advantages of SEL.

The results of applying SEL to these artificial test images are contrasted with two threshold type detectors in Figure 5.4: the 5x5 Sobel [Dud73] is representative of small, gradient type detectors that appeared early in the literature; the Canny detector [Can83] is one of the latest of the large support detectors, using directional non-maximum suppression (equivalent to zero-crossing detection of the second derivative) and hysteresis thresholding. In the upper series of images, the edge operator used for *both* the threshold detector and SEL was the Sobel operator. The original test image at SNR = +7 dB is in 5.4a. 5.4b is the output of the 5x5 Sobel edge operator. Using the experimentally determined optimum threshold on this output, the edge map obtained is shown in 5.4c. The merit value is F = 0.59. At this SNR, a simple threshold on the Sobel operator output is quite obviously inadequate. Much of the true edge is missed and many false edge points are present. A lower threshold might improve connectivity along the true edge somewhat, but only by admitting even more false alarms. The SEL output ***using the same 5x5 Sobel magnitude map*** (5.4b) ***as input*** is shown in 5.4d. Its figure of merit is F = 0.99. This edge map clearly shows the advantage of sequential searching over thresholding. Connectivity of the detected edge is complete, the edge almost per-

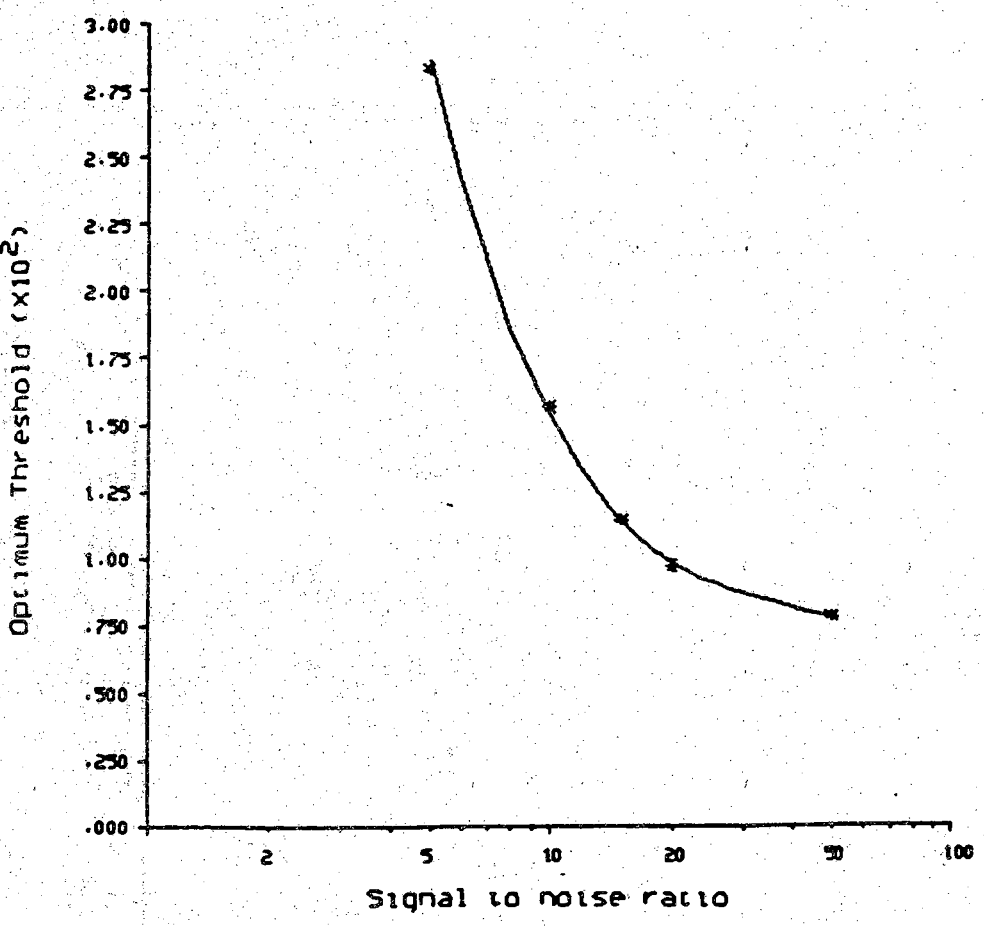

Figure 5.2: Optimum Pratt figure of merit threshold vs. signal-to-noise ratio for one 3x3 operator.

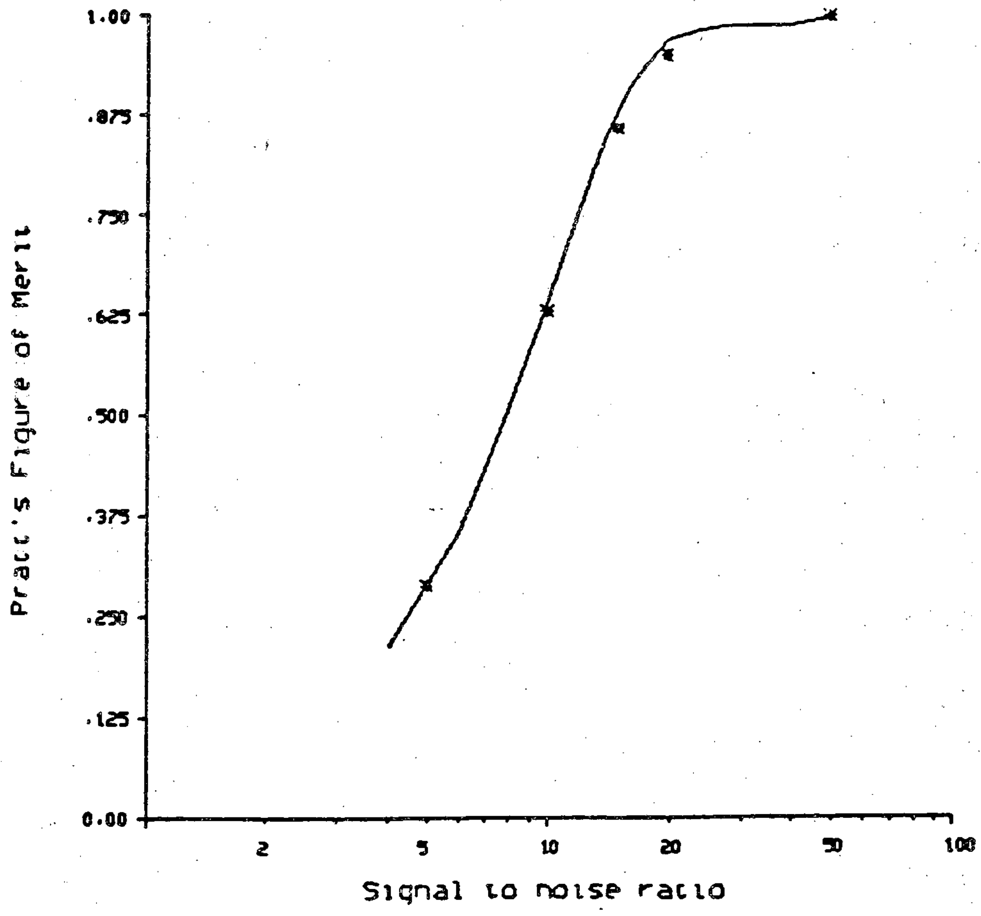

Figure 5.3: Pratt figure of merit value vs signal-to-noise ratio for one 3x3 operator.

fectly coincides with the true edge, and there are *no* false alarms.

At lower signal-to-noise ratios, small 5x5 operators are inadequate. The lower series of images in Figure 5.4 consider a much larger and more sophisticated edge operator. Here, the test image has SNR = -7 dB (5.4e). The magnitude map of the operator is given in 5.4f. This is the magnitude of the gradient of the Gaussian smoothed image, M, as described in Section 5.1. For this example, the standard deviation of the G filter was $\sigma = 3.5$ pixels. The edge map of Figure 5.4g was obtained by the Canny method from the magnitude and direction maps (before quantization to eight bits). That is, non-maximum suppression in the direction of the gradient is performed, followed by thresholding with hysteresis [Can83]. Even with this sophisticated method, it is again impossible to choose hysteresis thresholds such that good connectivity and few false alarms are achieved. In 5.4g, despite the many false edges, the connectivity along the true edge is still not complete. The merit value for 5.4g is F = 0.26. When SEL is applied to the *same* operator output, the resulting edge map is as in 5.4h with F = 0.52. Although there is one false edge segment at this low SNR, the true edge is detected with a *completely connected* contour. Furthermore, there is no trial and error selection of thresholds as in 5.4g.

5.2.2. Qualitative Comparisons

Even more striking comparisons then in the foregoing Section may be made with real images. This results from the fact that the artificial images are inadequate models of reality in many respects. Chief among these are the independence of the additive noise and the homogeneity of these images. Both of these characteristics tend to make threshold-type detectors appear better than they are. In a homogeneous image with

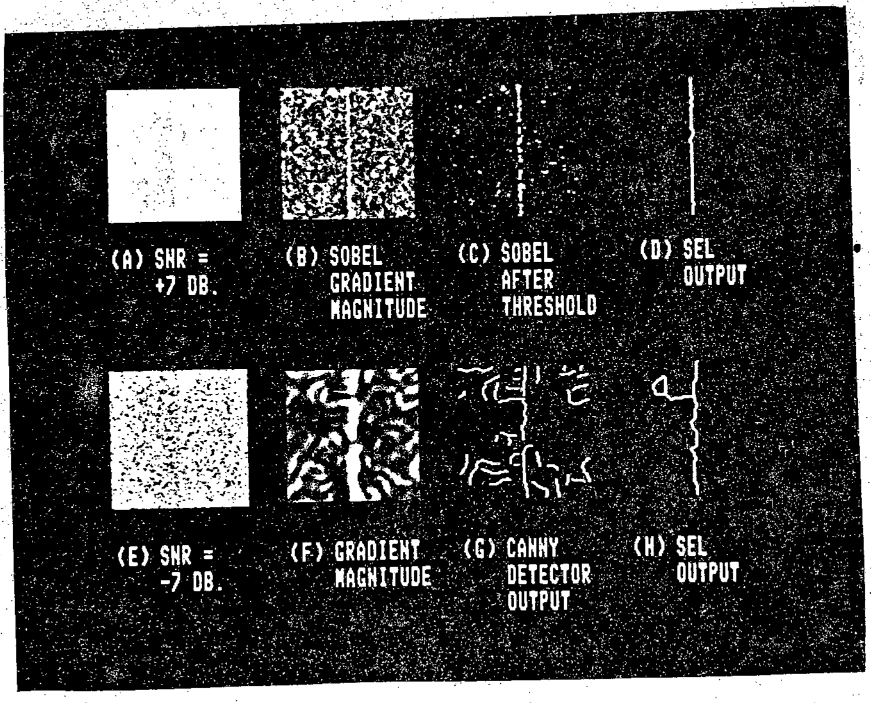

Figure 5.4: (a) Pratt figure of merit test image at SNR = +7 dB. (b) Gradient magnitude map obtained from a 5x5 Sobel operator. (c) Sobel detector output after application of a decision threshold to (b). (d) SEL output using (b) as input. (e) Pratt figure of merit test image at SNR = -7 dB. (f) Gradient magnitude map obtained from a Gaussian weighted gradient operator with $\sigma = 3.5$ pixels. (g) Output of Canny directional derivative detector. (h) Output of SEL algorithm.

independent noise, the best threshold to use at one location of the image is also optimum everywhere else. However, in *real* images, such coincidences do not occur in general. The relative merits of sequential searching over thresholds then becomes even more important.

To illustrate, we will again compare SEL to the two edge detectors considered in Section 5.2.1. For the small support Sobel operator we will consider the fairly high contrast image of Figure 5.5a. The Sobel operator output for this image is shown in Figure 5.5b. After application of a threshold, the edge map appears in 5.5c. This simple example clearly illustrates the classic trade off between connectivity and sensitivity. In 5.5c, many of the principal edges have dropout regions where the contour is broken. Note especially the lowest contrast part on the right of the image. And yet, despite the dropouts, other strong edges have multiple pixel widths and there are many spurious false edge points. The SEL edge map, using the *same* Sobel operator input data, shows completely connected, single pixel width edge contours (Figure 5.5d). The low contrast part on the right is detected as well as the higher contrast parts. The good connectivity results in object bounding contours that are closed in every case. This characteristic may be of great importance to a subsequent processing step such as shape analysis. Execution time for the SEL algorithm on a VAX 11-780 processor was 1.29 CPU seconds to obtain 5.5d.

The digital subtraction angiography image of Figure 5.6a is a challenging test for the large support Gaussian weighted gradient operator as the edges in that image are of low contrast and the signal-to-noise ratio is poor. This image is obtained by making two radiographs of the same area of the body in succession. Before the second exposure is

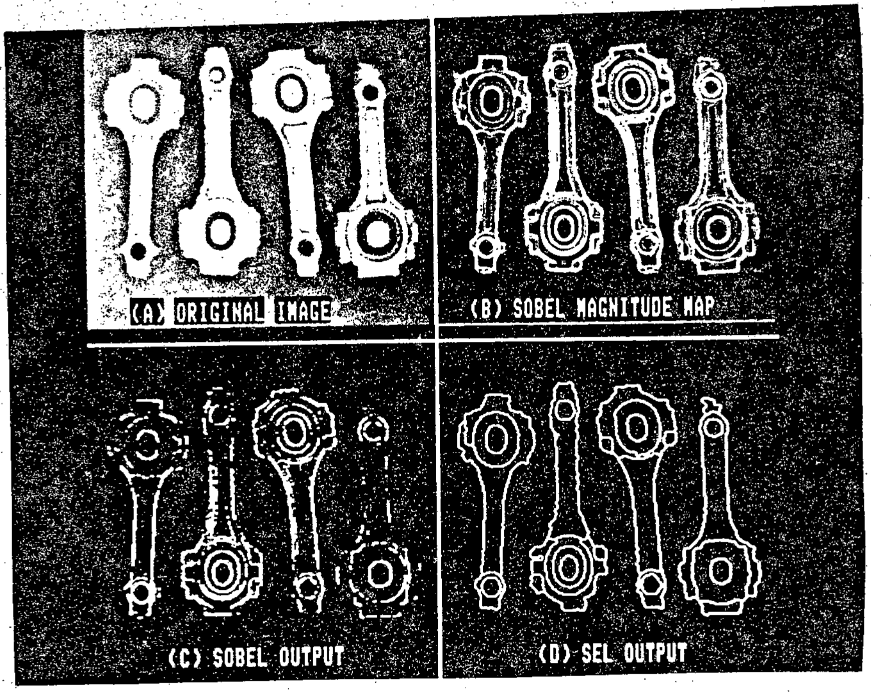

Figure 5.5: (a) Original machine parts image. (b) Gradient magnitude map obtained from a 5x5 Sobel operator. (c) Edge map after application of decision threshold to (b). (d) Edge map from SEL algorithm using (b) as input.

made, a contrast dye with a high attenuation to X-rays is introduced into the blood. When the two exposures are subtracted, the vessels carrying the contrast dye show up more clearly relative to background structures than in either single exposure. Figure 5.6b shows the magnitude of the gradient for an operator with Gaussian standard deviation $\sigma = 2.0$ pixels. When the Canny detector is applied to this angiogram image, the resulting edge map is as shown in 5.6c. Here, the poor connectivity and threshold sensitivity is very apparent. We again see the many false edge contours and yet some true edge points such as in the right branch of the artery are missed. In this image, the edge contrast and SNR change over a wide range which exacerbates the task of threshold selection. The SEL results using the same gradient information is given in 5.6d. The *edge operator* is exactly the same in both 5.6c and 5.6d. The difference is that instead of performing non-maximum suppression and thresholding with hysteresis on the operator output to get the edge map, the SEL algorithm is used. The edge connectivity and false edge rejection characteristics compare favorably with 5.6c. Execution time for the SEL algorithm for this example was 6.59 CPU seconds on the VAX 11-780 processor.

A comparison with another edge detector prominent in the literature is made in Figure 5.7. Here, the SEL algorithm is compared to the facet model operator of Haralick [Har84]. That operator also uses non-maximum suppression in the direction of the gradient. A threshold is applied to a measure of the edge contrast taken from a fitted parametric model of a local neighborhood. In this Figure, the output of an 11x11 size neighborhood operator is shown in b, and the SEL output using a $\sigma = 1.5$ gradient operator is shown in c. This size operator was chosen in this comparison because the number of coefficients in the operator window is the same as for the 11x11 facet operator. Again, the SEL output exhibits better connectivity and rejection of false edges, and

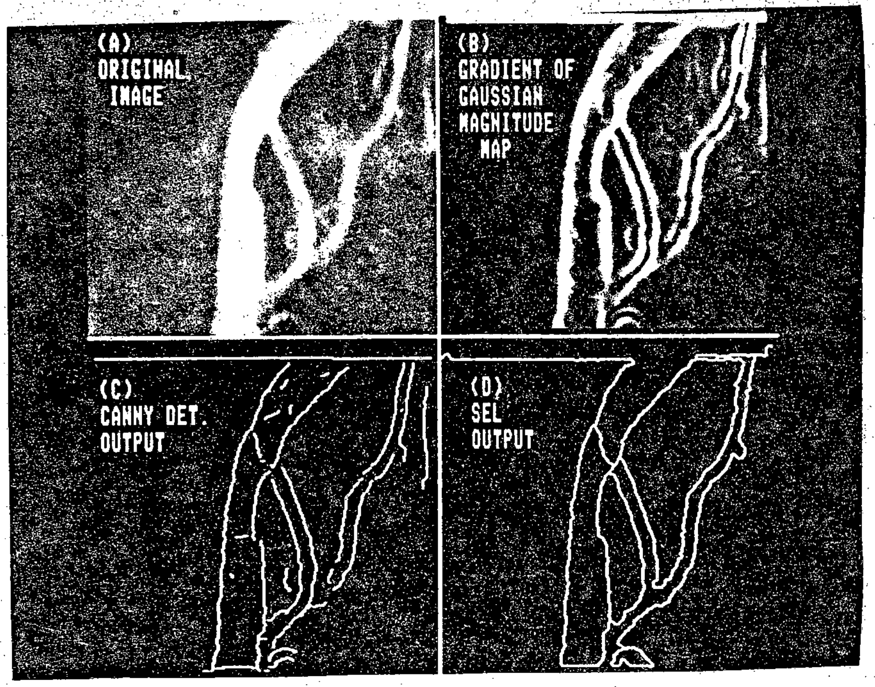

Figure 5.6: (a) Original arterial angiogram image. (b) Gradient magnitude map obtained from a Gaussian weighted gradient operator with $\sigma = 2.0$ pixels. (c) Edge map obtained from Canny directional derivative detector. (d) Edge map from SEL algorithm using (b) as input.

does not require the interactive selection of a decision threshold.

These examples illustrate the advantage of using a non-local sequential search procedure over a local threshold-based decision strategy. Quite apart from any considerations of the edge operator, at low signal to noise ratios, decisions involving large numbers of edge points are more easily made than when only a few edge points are considered. Instead of attempting to choose a decision threshold to make the edge / no edge classification on a pixel by pixel basis, the SEL algorithm makes a decision based on the many edge points contained in a long hypothesized path. It is this ability to utilize the soft information over many points rather than forcing hard decisions at every point that gives the technique its advantage in noisy situations.

5.3. Short Error Segments and the Markov Path Model

As noted in Sections 3.2.4 and 3.3.2, the Chernoff-based bounds on the distribution of computation and long error segment probability are not sufficiently tight to be directly useful for numerical results. Rather, that analytical work is useful for establishing the *characteristics* of sequential searching in images; to answer qualitative questions about the search behavior.

However we can illustrate the effect of the Markov Chain model for paths on the occurrence of short error segments. As discussed in Section 3.3.3, the use of the Markov model ought to make noise induced, small, erratic deviations from smooth straight edges less frequent. This is because the transition probabilities bias the branch metrics in favor of smooth edges which have higher a-priori probabilities. This may be experimentally verified by using an artificially generated image with straight edge segments at

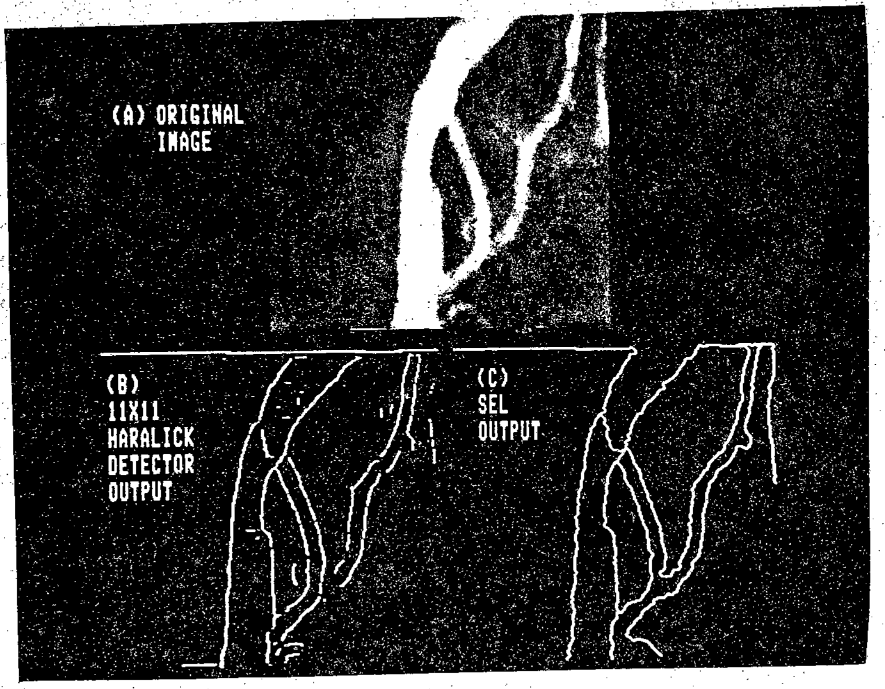

Figure 5.7: (a) Original arterial angiogram image. (b) Edge map obtained from 11x11 Haralick facet model directional derivative detector. (c) Edge map from SEL algorithm using a $\sigma = 1.7$ pixel (11x11) Gaussian weighted gradient operator as input.

known locations.

In Figure 5.8a we show the test image used for this purpose. The central region is of higher intensity than the surrounding background, the dividing border being a square with sides of length 64 pixels. White Gaussian noise has been added to bring the SNR to 10 dB in 5.8a and -3 dB in 5.8d. 5.8b and 5.8c show the results of the SEL algorithm on 5.8a with and without the Markov path model respectively. For the latter case, instead of the Markov transition probabilities entering the path branch metric, all path transitions were made equi-probable. That is, path transitions of left, straight, or right were simply set equal to one-third. Thus, all paths of the same length have equal a-priori probabilities. As can be seen from 5.8b and 5.8c, the path model has no effect at high SNR. The likelihood ratios of observations in the path metric completely dominate the path transition probabilities, be they Markov or equi-probable, so the detected edge coincides with the correct edge in either case.

The situation at low SNR is quite different, however. Figures 5.8e and 5.8f make the same comparison as before but now at -3 dB SNR. In this case, the high noise power corrupts the likelihood ratios to such a point that the detected edge in 5.8f (equi-probable paths) shows many small, erratic curves and bends where the true edge runs straight. The incorporation of the Markov model in the path metric (Figure 5.8e) has the effect of reducing these deviations. Notice that when the true edge *does* turn, as at the corners of the square, the searching algorithm responds appropriately. Here, the likelihood of the observations along a straight but incorrect path into the background is so low compared to that of the observations around the corner that the path with the highest metric follows the true edge. Thus, we see that the path branch metric derived

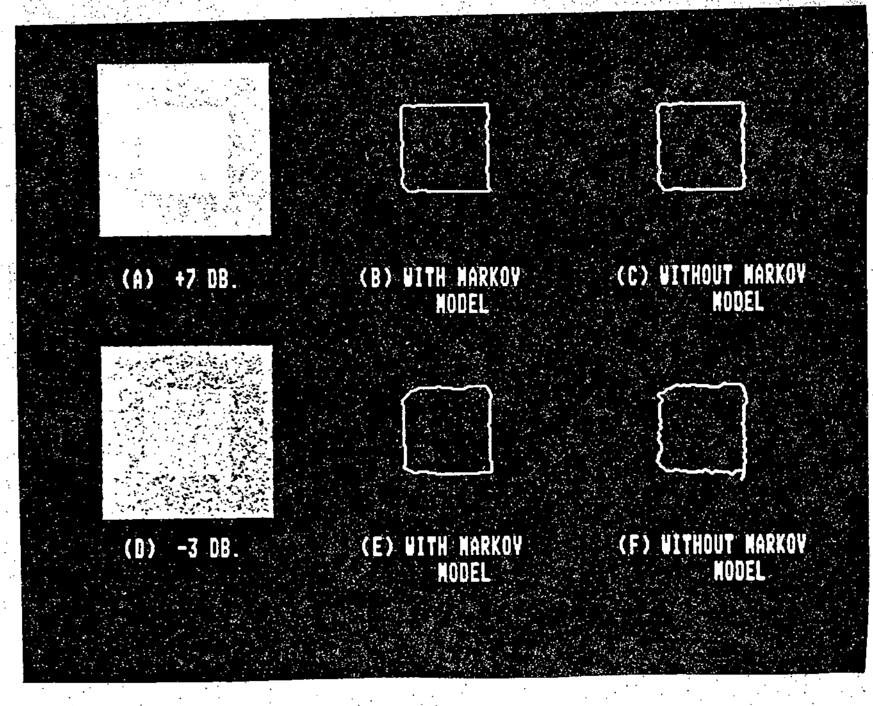

Figure 5.8: (a) Test image at SNR = +7 dB. (b) SEL output with Markov edge model incorporated in the algorithm. (c) SEL output without Markov edge model incorporated in the algorithm. (d) Test image at SNR = -3 dB. (e) SEL output with Markov edge model incorporated in the algorithm. The preprocessing edge operator is a Gaussian weighted gradient with $\sigma = 3.0$ pixels. (f) SEL output without Markov edge model incorporated in the algorithm.

in Section 2.2.3 strikes a subtle balance between the gradient observations and the a-priori probabilities of path topologies.

5.4. Correlated Random Fields

The premise advanced in Chapter 4 is that a particular model of the correlation among the random variables of a field implies that observations along paths searched by SEL are ARMA. This allows the searching metric to be based on an *uncorrelated* innovations sequence that results from passing the actual observations through a causal linear filter. Three questions remain unanswered. First, how appropriate is the D-SIM model? After suitable fitting of the model to a real image, how accurately does it reflect the actual correlation among pixels in the image? Second, how uncorrelated are the innovations obtained along search paths in an actual image? Finally, in what way, if at all, does this processing aid in the detection of edges by the SEL algorithm?

We may answer the first two questions by examining the autocorrelation function of observations and innovations along search paths. The model we have used is the second order D-SIM:

$$R_f\ (k,l) \ = \ \sigma^2 \left[c_1\, e^{-a_1(|k|+|l|)} + c_2\, e^{-a_2(|k|+|l|)} \right] \qquad (5.6)$$

Therefore, the observations along paths in the field obey (by Theorem 4.2):

$$f_i - \phi_1 f'_{(i)-1} - \phi_2 f'_{(i)-2} \ = \ \xi_{(i)} - \theta_1\, \xi_{(i)-1} \qquad (5.7)$$

$$E\left\{ \xi_{(i)}, \xi_{(j)} \right\} \ = \ 0 \qquad i \neq j$$

For the sake of computational efficiency and stability we have considered only the autoregressive terms. The moving average coefficient θ_1 is assumed to be zero. To fit the model to an image, the parameters ϕ_1 and ϕ_2 must be estimated. This is accomplished by generating a random path through the image. The observations along the path are used to estimate the values of the autocorrelation function by time averages. The autoregressive coefficients can then be estimated from the autocorrelation function by standard methods [Box70]. The fitted ARMA model is then used to obtain the innovations sequence along search paths in the manner of Section 4.3.3.

To judge the effectiveness of the ARMA model in reflecting the actual autocorrelation function of path observations as well as its ability to remove correlation in the generated innovations sequence, we find the time averaged estimates of the autocorrelation function along actual detected edge paths after the search algorithm is terminated. The autocorrelation function of the generated innovations along those paths can also be estimated. These two functions are then compared to the theoretical autocorrelation function predicted from the ARMA model. These results are shown in Figure 5.9. The long- dashed curve in Figure 5.9 is the experimentally determined autocorrelation function of observations along detected edge paths in an image. The solid curve is the autocorrelation function predicted from the ARMA model fitted to the image as above. The short-dashed curve is the autocorrelation function of the innovations sequence along the same detected edge paths. Two observations may be made. One is that the fitted ARMA model does a creditable job of modeling the actual correlation among observations along a path. The second is that the autocorrelation function of the innovations process, while not purely impulsive as it would be if it were truly uncorrelated, is nevertheless much less correlated than the original observations.

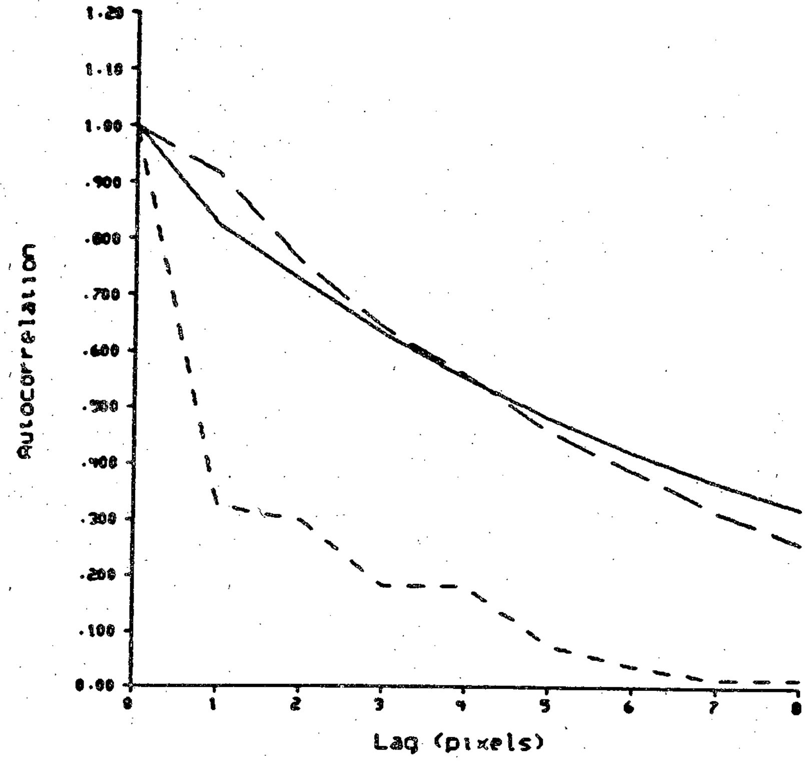

Figure 5.9: Long dashed line: measured autocorrelation function along paths in a real image. Solid line: autocorrelation function along paths in the fitted D-SIM random field model. Short dashed line: autocorrelation function of the innovations process along the same paths in the real image.

The effect of using this predictive filter in the searching algorithm is dramatic. Beyond its theoretical importance, the ability to predict the next path observation from the previous observations reduces the amount of incorrect path searching, reducing the average computation time. This is easy to see in light of the results obtained in Chapters 3 and 4. From Equation (5.7) we see that variance of the innovations process must be less than that of the original observations. This means that there is necessarily less overlap in the conditional densities p_1 and p_0 (see Figure (3.9) and Equation (3.29)) since the two processes have the same mean. Consequently, the quantity $D(\alpha\rho)$ is smaller so the bound on the distribution of computation is lower.

This effect is clearly visible in the processing of real images. We determine the effective search rate, i.e. the ratio of detected edge nodes to the total number of nodes searched. This can be graphically illustrated by performing the sequential searching with a limited stack size. Identical edge contours are searched with and without the predictive filter. Because of the limited stack, if the search rate is low, e.g. when the predictive filter is not used, only a portion of the edge contours are detected before the stack fills up. When the filter is used, however, many fewer incorrect paths are searched allowing more of the true edge contours to be detected. Figures 5.10 and 5.11 are two such examples. The original images are in the upper left of the Figures. In the lower right and lower left we see the resulting edge contours as determined by the SEL algorithm with and without the predictive filter, respectively. The stack size is limited to 2000 nodes in each case and, other than the filter, they are processed in identical fashion. The first three start nodes as determined by the algorithm are in the vicinity of the numbers and are the same for both variations of each image. In Figure 5.10, the number of nodes detected by the algorithm for the three paths shown is 81, 41, and 49 when the filter is not used. When the predictive filter is used, 393 edge nodes are found from the first start node, 319 from the second, and 310 from the third. In fact, as can be seen from the Figure, the entire edges associated with those start nodes are detected when the filter is in use. To put these numbers in perspective, consider the fraction of detected nodes to total nodes searched (4000 nodes per root node). When the filter is not in use this fraction is about 1/70, but it jumps to roughly 1/12 with the filter. Thus, the use of the predictive filter is seen to increase the effective search rate (number of detected nodes / number of nodes searched) by a factor of 6 for this image. In Figure 5.11, 162, 133, and 68 nodes are detected starting at the root nodes shown without the filter. Using the filter, the corresponding numbers are 553, 164, and 550. If we ignore the second contour in each case, the filter is seen to increase the search rate from about 1/40 to 1/8 or by a factor of 5. (The reason we do not consider the second contour is because the search is terminated early when the first contour is intersected for the case the filter is used, but no such intersection occurs when the filter is not used.)

Before leaving this topic, it is interesting to see a picture of *all* the nodes searched in an image. This gives a visual impression of how the amount of computation varies spatially over an image. In Figure 5.12 we again show the angiogram image of Figure 5.6, but here the overlay consists of *every* node ever visited by the search algorithm rather than just the final edge paths decided upon. As can be seen in this Figure, in those portions of the image where the contrast is high, very little incorrect searching is performed. However, where the contrast is low (e.g. at the artery junctions in the lower right), many additional nodes have been hypothesized as the algorithm attempts to find the best path.

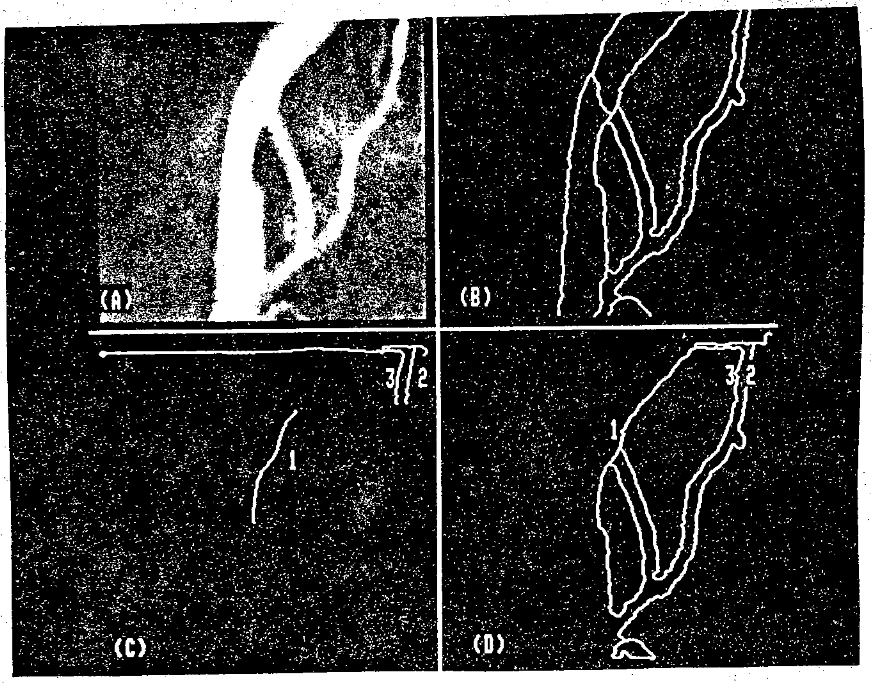

Figure 5.10: (a) Original image. (b) Final edge map from SEL. (c) SEL detected edges from the three indicated root nodes without the use of the prediction filter. (d) SEL detected edges from the same three indicated root nodes with the prediction filter in use.

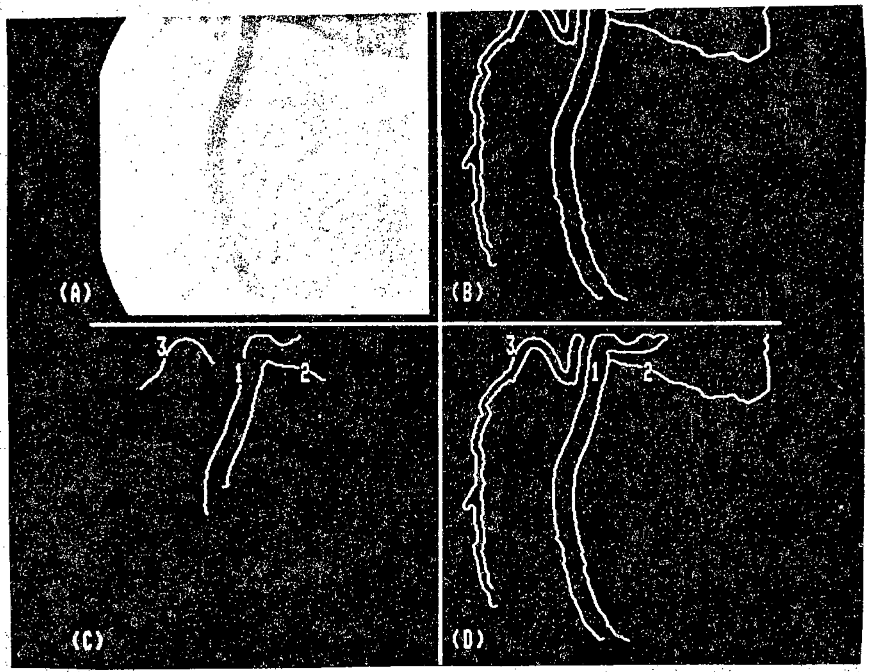

Figure 5.11: (a) Original image. (b) Final edge map from SEL. (c) SEL detected edges from the three indicated root nodes without the use of the prediction filter. (d) SEL detected edges from the same three indicated root nodes with the prediction filter in use.

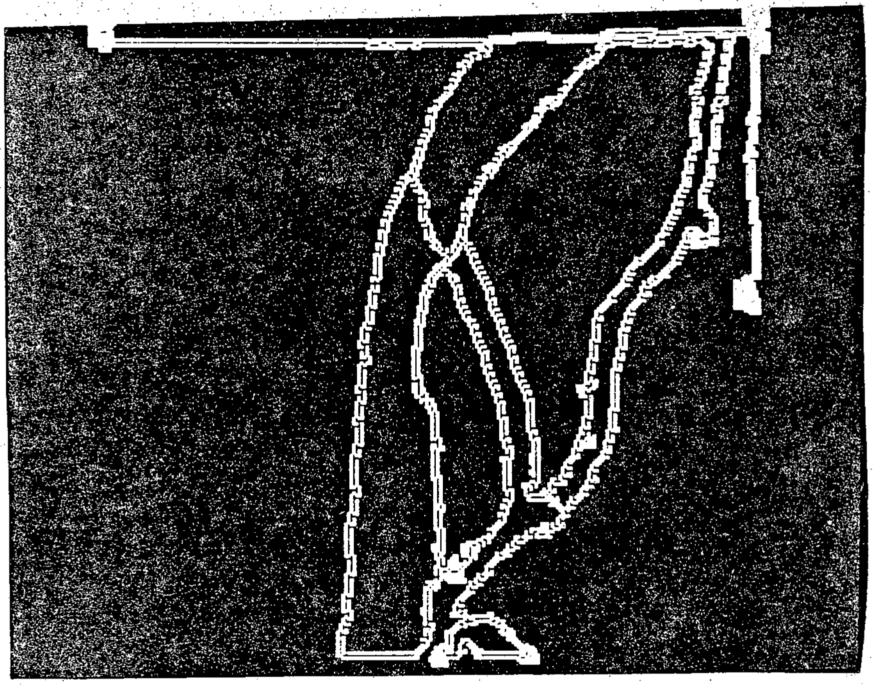

Figure 5.12: Image showing all the nodes searched by the SEL algorithm in the image of Figure 5.6a. The edge map of Figure 5.6d generated by the algorithm is overlaid in black. The white areas are all the other, incorrect nodes visited by the algorithm.

5.5. Examples

In this Section we present a number of experimental results to demonstrate the efficacy of the SEL approach. To demonstrate its usefulness in detecting linear features other than intensity edges, we have included an example of ramp edge detection and of line detection.

5.5.1. Ramp Edge Detection

Ramp edges in images are characterized by an intensity profile across the edge that rises linearly to a point and then decreases linearly thereafter. Because they are generally very wide and because the intensity gradient is nearly the same over this width, gradient-type enhancement operators are ineffective on these edges. The location of the edge is generally taken to be the top of the ramp. Since the gradient *direction* changes by π at this point, one possible enhancement operator suggested by Machuca and Gilbert [Mac81] calculates the integral:

$$\gamma = \int_C d\Theta(t) \tag{5.8}$$

on a closed curve C around every point in the image. In Equation (5.8), $\Theta(t)$ is called an angular function of the gradient direction field D (Equation (5.3)) and is given by:

$$\Theta(t) = D(t) - D(t_0) \tag{5.9}$$

where t_0 is any starting point on the curve C. The integral γ is therefore just the integral of the changes in gradient direction around a closed curve. If C surrounds a point lying on an intensity step edge, the gradient direction is constant everywhere around the point so $\gamma = 0$. If, however, the point is on a ramp edge for which the gradient direction differs by π radians on either side of the edge, then $\gamma = 2\pi$.

For this example, our test image is the artificial range image of Figure 5.13a. Here intensity is proportional to *range* from the viewer. It is of a three dimensional cube in which holes perpendicular to the faces have been drilled. The image exhibits *step edges* at the block boundaries due to depth discontinuities. It exhibits purely *ramp edges* where the front face meets the left face, where the front face meets the top face, and where the left face meets the top face. There are also more complicated combinations of step and ramp edges at the rims of the holes. Because the range (intensity) is *continuous* at the ramp edges, they do not show up at all in the gradient magnitude map of Figure 5.13b.

We implement a simple ramp edge enhancement operator by approximating γ in Equation (5.8) by:

$$\gamma = \sum_{1}^{8} \Theta_{t_i} = \sum_{1}^{8} \left[D_{\vec{r}_{i+1}} - D_{\vec{r}_i} \right] \qquad (5.10)$$

Here, $\vec{r}_i$, $i \in [1,8]$ is the set of eight nodes surrounding a point in a 3x3 neighborhood. (Note that $\vec{r}_9 = \vec{r}_1$) The γ is the sum of changes in gradient direction going around those eight neighbors of a point. When this operator is applied to the gradient direction map D, the result is shown in Figure 5.13c. Unlike 5.13b, the ramp edges are clearly visible here. Figure 5.13c also shows fainter ramp edges on either side of step edges in 5.13b because of the change in gradient direction caused by low pass filtering the step edges.

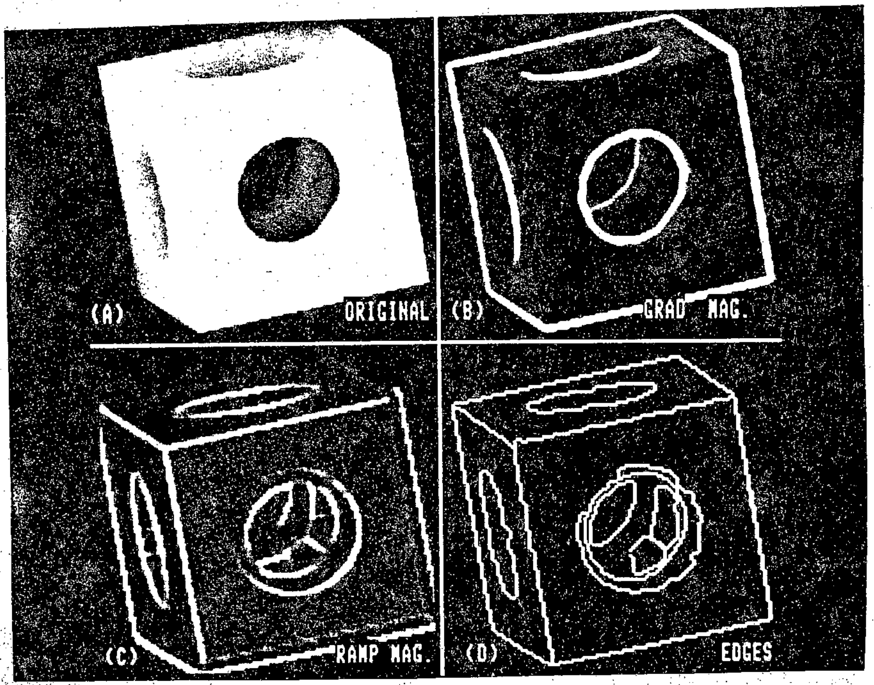

Figure 5.13: (a) Original range image of a 3-D cube. Intensity is proportional to distance from viewer. (b) Step edge magnitude map. (c) Ramp edge magnitude map. (d) SEL edge map using both step edge and ramp edge magnitude maps as input.

To obtain the edge map in Figure 5.13d, the SEL algorithm was first applied to the gradient magnitude map of 5.13b to obtain the step edges and then to the γ map of 5.13c to obtain the ramp edges. Thus, the same SEL algorithm is seen to detect two different types of linear features (step and ramp edges) simply by using different preprocessing operators.

5.5.2. Line Detection

Lines are another linear feature of interest in images. Figure 5.14a shows an aerial photograph containing roads which have a line like intensity profile. Using simple horizontal and vertical line operators suggested by Rosenfeld and Thurston. [Ros71]:

$$l_h = \begin{bmatrix} \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \\ 1 & 1 & 1 \\ \frac{-1}{2} & \frac{-1}{2} & \frac{-1}{2} \end{bmatrix}$$

$$l_v = \begin{bmatrix} \frac{-1}{2} & 1 & \frac{-1}{2} \\ \frac{-1}{2} & 1 & \frac{-1}{2} \\ \frac{-1}{2} & 1 & \frac{-1}{2} \end{bmatrix} \tag{5.11}$$

combined as: $L = [l_h^2 + l_v^2]^{0.5}$, we obtain the "line magnitude map" of Figure 5.14b. This magnitude map may be searched by the SEL algorithm in the same fashion as for other linear features. The resulting detected line map is shown in Figure 5.14c. While the line operator used here is not particularly sophisticated, this example and that in the previous Section demonstrate that SEL is effective for the detection of linear features in two-dimensional arrays independent of the nature of the feature. Only the preprocessing

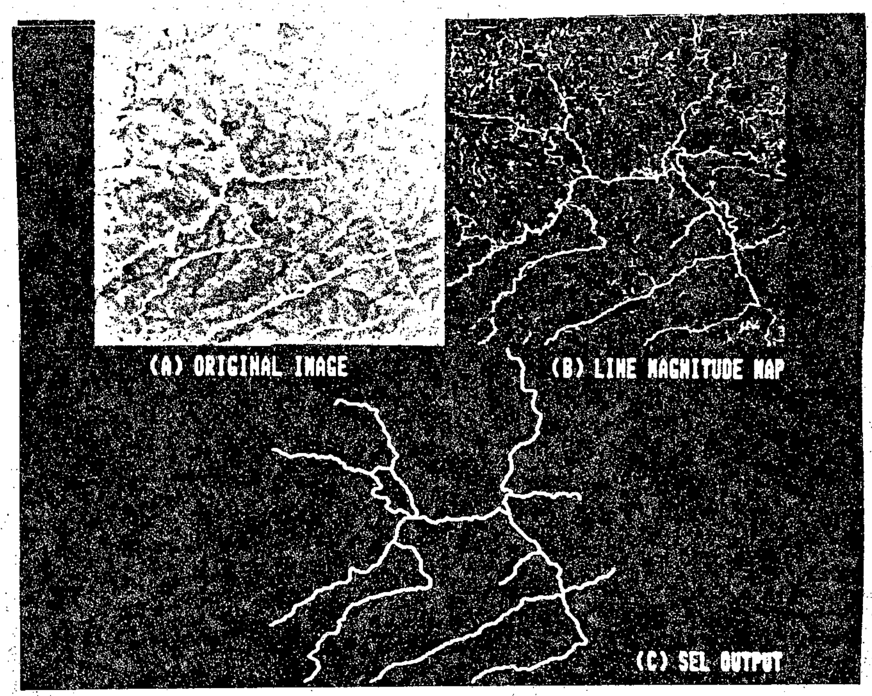

Figure 5.14: (a) Original aerial image of landscape. (b) Line magnitude map obtained using line operators described in text. (c) SEL output using (b) as input.

operator must be tailored to emphasize the desired feature.

5.5.3. Other Examples

Finally, in this Section, we present some results from applying the SEL technique to a variety of images with intensity step edges. In all of these examples, the preprocessing operator is the Gaussian weighted gradient and the SEL particulars are as given in Section 5.1. The gradient operator is of width $\sigma \in [1,2]$.

The Figures are divided into two types. In the first type a number of medical images have been examined. Figures 5.15 through 5.17 are arterial angiogram images as described in Section 5.2.2. These are followed in Figures 5.18 and 5.19 by heart ventricle images generated by a similar procedure. It is important to note in all of these examples that no human operator interaction is necessary to obtain the results; the processing is completely automatic.

The second type of example, Figures 5.20 and 5.21, is of machine parts. Again the processing is totally automatic. The excellent connectivity and completely closed bounding contours in the edge maps are desirable attributes since edge detection applied to such images is generally part of a much larger computer vision scheme. Subsequent processing tasks such as shape analysis and object recognition are greatly facilitated by high quality edge maps.

5.6. Conclusions

In this thesis, we have advanced an argument for performing the detection of linear features in images and two-dimensional random fields in a sequential manner. We have

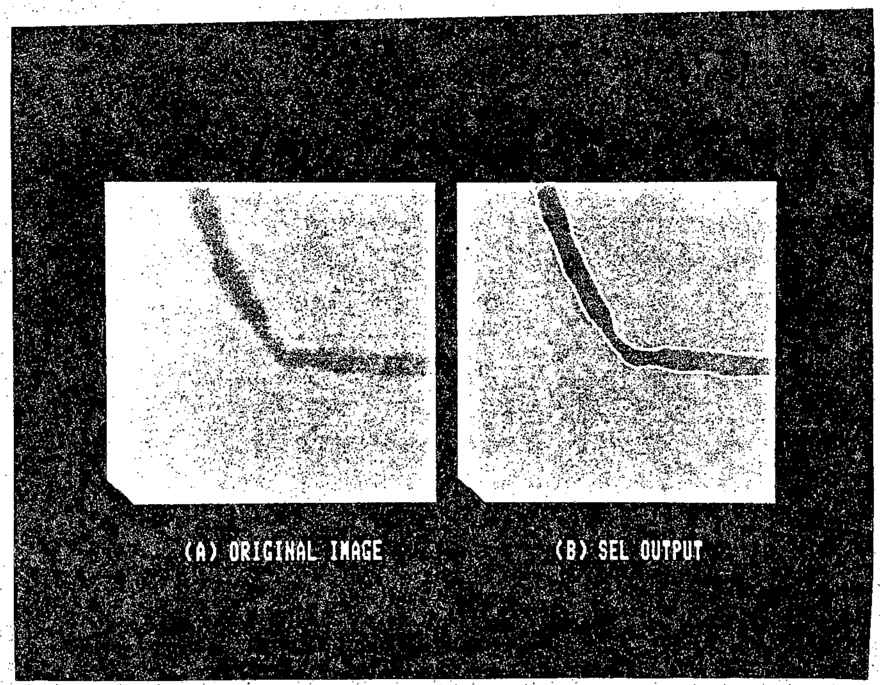

Figure 5.15: (a) Original angiogram image. (b) SEL edge map overlaid on (a).

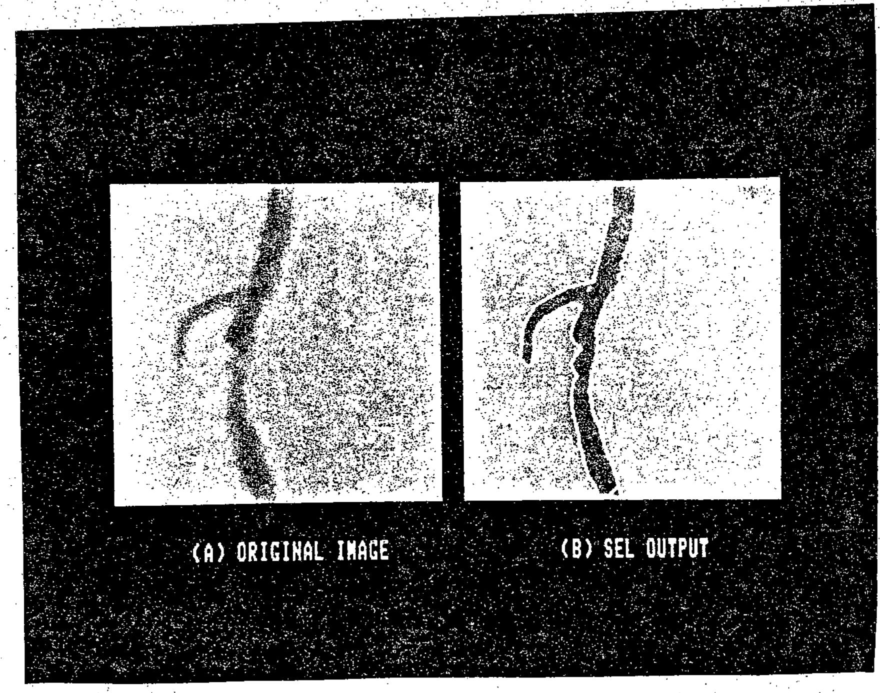

Figure 5.16: (a) Original angiogram image. (b) SEL edge map overlaid on (a).

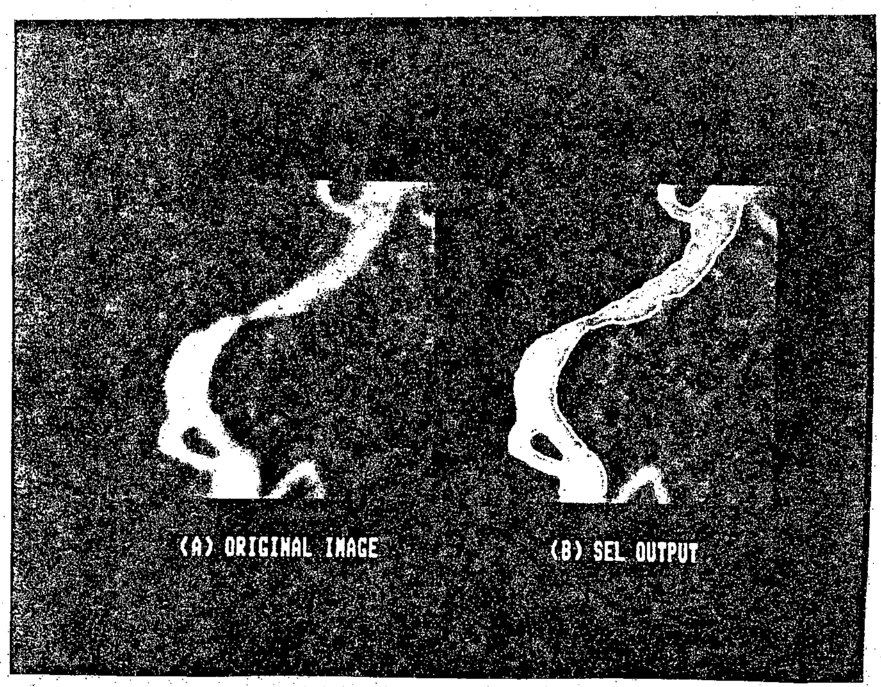

Figure 5.17: (a) Original angiogram image. (b) SEL edge map overlaid on (a).

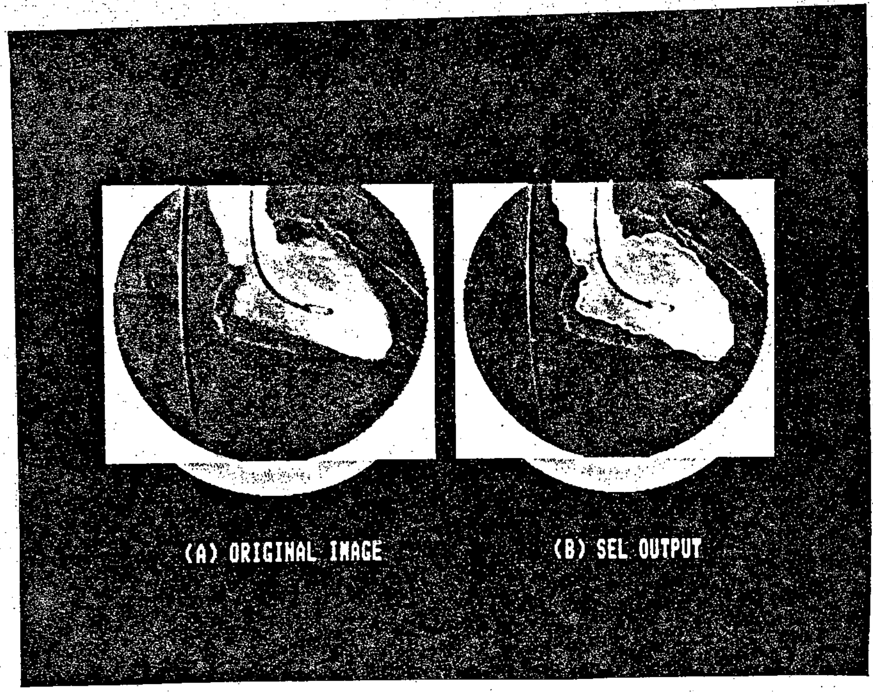

Figure 5.18: (a) Original ventricle image. (b) SEL edge map overlaid on (a).

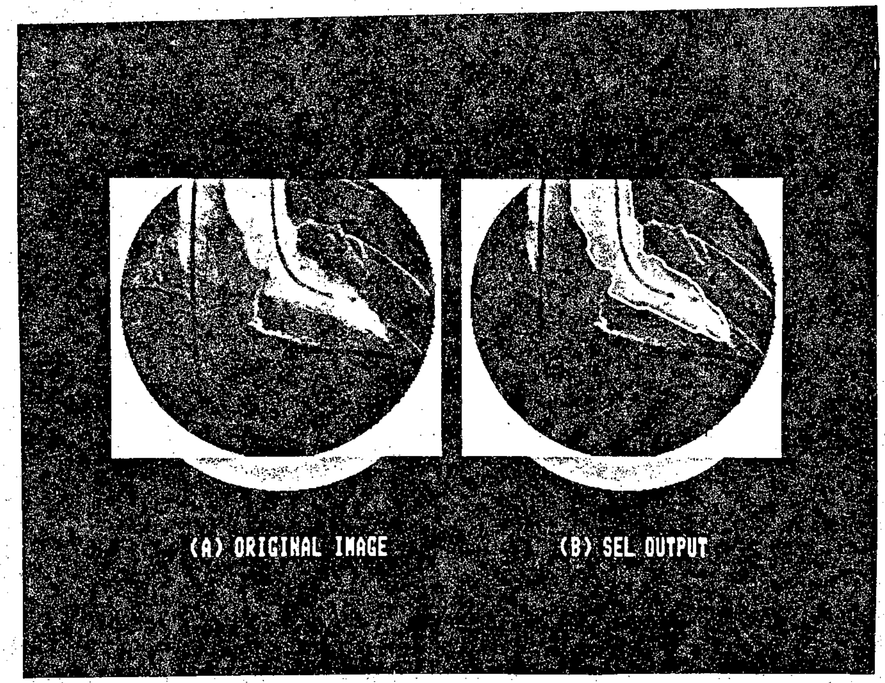

Figure 5.19: (a) Original ventricle image. (b) SEL edge map overlaid on (a).

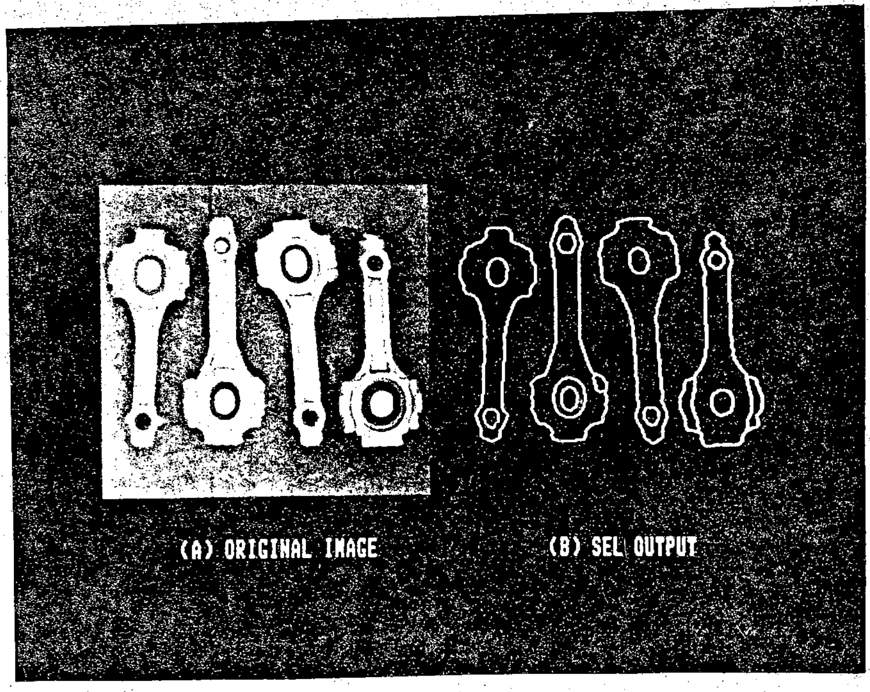

Figure 5.20: (a) Original parts image. (b) SEL edge map.

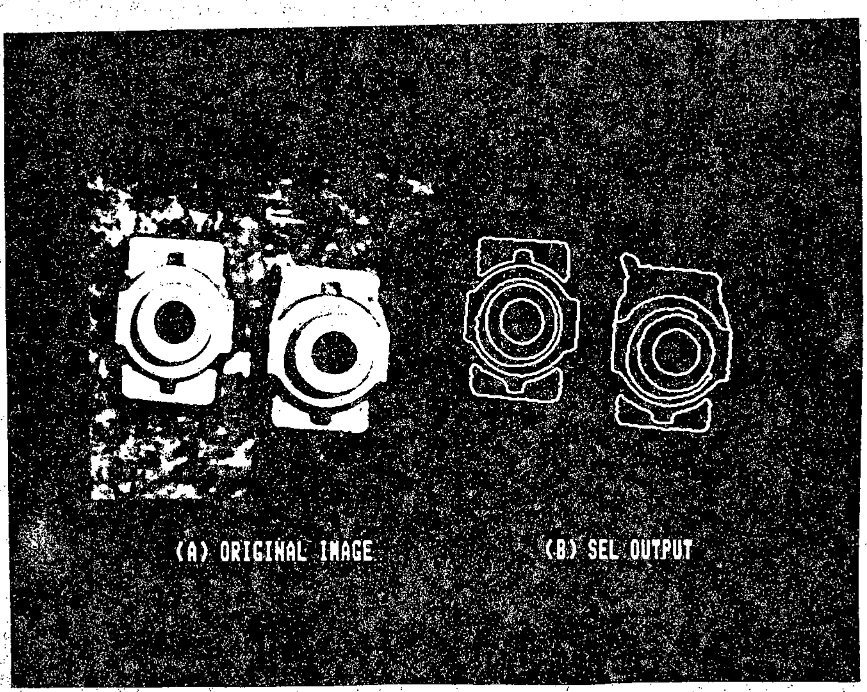

Figure 5.21: (a) Original parts image. (b) SEL edge map.

formulated the problem as a best path tree search, proposed a Markov model for paths in two-dimensional random fields, and derived a path branch metric. This metric was shown to possess an important conditional drift characteristic for sequential searching. Next, theoretic bounds were found on the algorithm's distribution of computation and probability of error. These bounds are useful in describing the qualitative behavior of the searching algorithm as a function of the parameters of the random field model. This theory was then generalized to the case of correlated random fields. A Markov random field model was proposed for which it was shown that observations along search paths obey a finite order difference equation. Therefore, a causal linear filter may be constructed such that optimum (in the mean squared error sense) predictions of future path observations may be calculated to aid in the search process.

The sequential detection technique proposed here was implemented in software and used for a number of experiments. These experiments were designed to illustrate and substantiate the analytical results. The method was compared to several current techniques in the literature on the basis of both a quantitative performance measure and subjective quality using real images. The effect of the Markov path model and of the predictive filter for correlated observations was explored. Finally, examples of the use of this technique for ramp edge detection, line detection, and step edge detection were provided.

This method of detecting linear features was shown to produce very well connected edge maps, with a low false edge detection probability, and with a minimum of user interaction. These characteristics are all extremely important in many computer vision applications.

There are numerous possible avenues for future research. On the analytic side, the path branch metric is only as good as the model for the conditional measures p_0 and p_1. While these were merely assumed to be Gaussian with parameters estimated from the image in the current implementation, future work might be directed toward finding more accurate models for these distributions. Also, the whole question of vector observations is open. On the algorithmic side, several possibilities exist. Only the stack algorithm was investigated here. In particular, the Fano search algorithm (Section 2.3.5) is an attractive alternative as it would allow a very fast hardware implementation. Knowledge-based root node selection and termination strategies might also be examined. These are appropriate where some prior knowledge of the image content is available. Finally, a knowledge-based component to the path metric in the manner of Ashkar and Modestino [Ash78] may prove very useful where much is known about the image.

APPENDIX

APPENDIX

Markov Transition Probabilities for Paths

In Section 2.2.2 we introduced a Markov model for paths in a two-dimensional rectangular lattice. According to this model, a path is denoted by a start or root node, a starting direction, and a sequence of direction transition letters:

$$\mathbf{m} = \vec{r}_0 \times \vec{d}_0 \times \left[a_1 , , , a_n\right]$$
$$a_i \in A = \left[L ,S ,R\right]$$

The *state* of the process is defined to be the last k transition letters:

$$s_i = \left[a_{i-1} \cdots a_{i-k}\right]$$

and the a-priori probability of such a path is given by:

$$Pr\ (\mathbf{m}) = \prod_{i=1}^{n} Pr\ (s_i \mid s_{i-1})$$

In this appendix we will address the question of how the state transition probabilities are estimated.

The state transition probabilities may be estimated from a collection of images in the following manner. An edge detector is applied to the images in the collection and the resulting edge maps obtained. For the present purposes, complete connectivity is not essential, but edge thinness is. Therefore, edges with multiple pixel width are discarded from the edge maps. The connected edge segments in these edge maps are then traced, recording the sequences of direction changes along their length. If a direction change of $\pi / 2$ or greater is observed, the sequence is terminated at that point and the rest of the edge is treated as a new edge segment. It may be noted that this is a very rare occurrence. These sequences of direction changes for edges in the images of the collection can then be used to find the corresponding state sequences, since, given an initial state, s_0, a state sequence is uniquely determined from the the direction change sequence (Equation 2.4). The initial state is assumed to be the "all straight" state:

$$s_0 = \left[S\ ,\ ,\ ,\ S\right]$$

Finally, the probabilities of the state transitions $Pr\ (s_i \mid s_j)$ are estimated as the relative frequency of those transitions in the observed state sequences. Thus, the state transition matrix is estimated from the relative frequencies of state transitions along edge segments of the images in the collection.

The collection of images used for this purpose include man-made parts as well as natural scenes. Two tables of transition probabilities are shown in Figures A.1 and A.2 for $k = 2$ and $k = 3$ order Markov models. Note for example that the state transition from "all straight" to "all straight" has a much higher probability than the transitions "all straight" to [S,S,R] or [S,S,L] (or [S,S,S,R], [S,S,S,L]). Also, very erratic states such as [S,R,L,R,] have a low probability of occurrence.

State i:	State i+1:	Transition Prob:	Prob of State i:
L,L	L,L	0.1958	
L,L	L,S	0.2028	0.0712
L,L	L,R	0.6014	
L,S	S,L	0.1399	
L,S	S,S	0.5267	0.0605
L,S	S,R	0.3333	
L,R	R,L	0.2894	
L,R	R,S	0.3293	0.1247
L,R	R,R	0.3812	
S,L	L,L	0.2140	
S,L	L,S	0.0856	0.0640
S,L	L,R	0.7004	
S,S	S,L	0.0942	
S,S	S,S	0.8116	0.3594
S,S	S,R	0.0942	
S,R	R,L	0.7004	
S,R	R,S	0.0856	0.0640
S,R	R,R	0.2140	
R,L	L,L	0.3812	
R,L	L,S	0.3293	0.1247
R,L	L,R	0.2894	
R,S	S,L	0.3333	
R,S	S,S	0.5267	0.0605
R,S	S,R	0.1399	
R,R	R,L	0.6014	
R,R	R,S	0.2028	0.0712
R,R	R,R	0.1958	

Figure A.1: State transition probabilities and state probabilities for a $k = 2$ Markov path model.

State i:	State i+1:	Transition Prob:	Prob of State i:
L,L,L	L,L,L	0.3922	
L,L,L	L,L,S	0.1176	0.0134
L,L,L	L,L,R	0.4902	
L,L,S	L,S,L	0.1600	
L,L,S	L,S,S	0.3200	0.0131
L,L,S	L,S,R	0.5200	
L,L,R	L,R,L	0.1369	
L,L,R	L,R,S	0.2500	0.0440
L,L,R	L,R,R	0.6131	
L,S,L	S,L,L	0.3529	
L,S,L	S,L,S	0.1471	0.0089
L,S,L	S,L,R	0.5000	
L,S,S	S,S,L	0.0960	
L,S,S	S,S,S	0.6560	0.0327
L,S,S	S,S,R	0.2480	
L,S,R	S,R,L	0.6329	
L,S,R	S,R,S	0.1139	0.0207
L,S,R	S,R,R	0.2532	
L,R,L	R,L,L	0.2302	
L,R,L	R,L,S	0.2590	0.0364
L,R,L	R,L,R	0.5108	
L,R,S	R,S,L	0.2704	
L,R,S	R,S,S	0.6164	0.0416
L,R,S	R,S,R	0.1132	
L,R,R	R,R,L	0.7353	
L,R,R	R,R,S	0.1118	0.0445
L,R,R	R,R,R	0.1529	
S,L,L	L,L,L	0.1500	
S,L,L	L,L,S	0.5750	0.0105
S,L,L	L,L,R	0.2750	
S,L,S	L,S,L	0.2000	
S,L,S	L,S,S	0.4500	0.0052
S,L,S	L,S,R	0.3500	
S,L,R	L,R,L	0.2326	
S,L,R	L,R,S	0.4651	0.0450
S,L,R	L,R,R	0.3023	
S,S,L	S,L,L	0.1484	
S,S,L	S,L,S	0.0469	0.0335
S,S,L	S,L,R	0.8047	
S,S,S	S,S,L	0.0758	
S,S,S	S,S,S	0.8484	0.3007

Figure A.2: State transition probabilities and state probabilities for a $k = 3$ Markov path model.

S,S,S	S,S,R	0.0758	
S,S,R	S,R,L	0.8047	
S,S,R	S,R,S	0.0469	0.0335
S,S,R	S,R,R	0.1484	
S,R,L	R,L,L	0.3023	
S,R,L	R,L,S	0.4651	0.0450
S,R,L	R,L,R	0.2326	
S,R,S	R,S,L	0.3500	
S,R,S	R,S,S	0.4500	0.0052
S,R,S	R,S,R	0.2000	
S,R,R	R,R,L	0.2750	
S,R,R	R,R,S	0.5750	0.0105
S,R,R	R,R,R	0.1500	
R,L,L	L,L,L	0.1529	
R,L,L	L,L,S	0.1118	0.0445
R,L,L	L,L,R	0.7353	
R,L,S	L,S,L	0.1132	
R,L,S	L,S,S	0.6164	0.0416
R,L,S	L,S,R	0.2704	
R,L,R	L,R,L	0.5108	
R,L,R	L,R,S	0.2590	0.0364
R,L,R	L,R,R	0.2302	
R,S,L	S,L,L	0.2532	
R,S,L	S,L,S	0.1139	0.0207
R,S,L	S,L,R	0.6329	
R,S,S	S,S,L	0.2480	
R,S,S	S,S,S	0.6560	0.0327
R,S,S	S,S,R	0.0960	
R,S,R	S,R,L	0.5000	
R,S,R	S,R,S	0.1471	0.0089
R,S,R	S,R,R	0.3529	
R,R,L	R,L,L	0.6131	
R,R,L	R,L,S	0.2500	0.0440
R,R,L	R,L,R	0.1369	
R,R,S	R,S,L	0.5200	
R,R,S	R,S,S	0.3200	0.0131
R,R,S	R,S,R	0.1600	
R,R,R	R,R,L	0.4902	
R,R,R	R,R,S	0.1176	0.0134
R,R,R	R,R,R	0.3922	

Figure A.2 (continued).

REFERENCES

REFERENCES

[Abd79] I. Abdou and W. Pratt, "Quantitative design and evaluation of enhancement/thresholding edge detectors," *Proc. of IEEE,* vol. 67, pp. 753-763, May 1979.

[Abe65] K. Abend, T. Harley, L. Kanal, "Classification of binary random patterns," *IEEE Trans. Inf. Theory,* vol. IT-11, pp. 538-544, 1965.

[Aho74] A. Aho, J. Hopcroft, and J. Ullman, *The Analysis and Design of Computer Algorithms,* Reading Mass.: Addison-Wesley, 1974.

[Ash78] G. Ashkar and J. Modestino, "The contour extraction problem with biomedical applications," *CGIP,* vol. 7, pp. 331-355, 1978.

[Bar69] T. Barnard and J. Burg, "Analytical studies of techniques for the computation of high-resolution wavenumber spectra," *Advanced Array Research Report #9,* Texas Instruments, Inc., May 1969.

[Bas81] M. Basseville, B. Espiau, J. Gasnier, "Edge detection using sequential methods for change in level - part I: a sequential edge detection algorithm," *IEEE Trans. Acoustics, Speech & Sig. Proc.,* vol. ASSP-29, pp. 24-31, Feb. 1981.

[Bob74] L. Bobrow and M. Arbib, *Discrete Mathematics: Applied Algebra for Computer Science and Information Science,* Philadelphia: W. B. Saunders, 1974.

[Box70] G. Box and G. Jenkins, *Time Series Analysis Forcasting and Control,* San Francisco: Holden-Day, 1970.

[Can83] J. Canny, "Finding edges and lines in images," *MIT AI-TR-720,* June 1983.

[Che77] P.R. Chevillat and D.J. Costello, Jr., "A multiple stack algorithm for erasure-free decoding of convolutional codes," *IEEE Trans. Commun.,* col. COM-25, pp. 1460-1470, Dec. 1977.

[Chi74] Y. Chien and K. Fu, "A decision function method for boundary detection," *CGIP,* vol. 3, pp. 125-140, 1974.

[Coo78] D. Cooper and H. Elliot, "A maximum likelihood framework for boundary estimation in noisy images," *Proc. IEEE Comp. Soc. Conf. Pattern Rec. & Image Proc.,* Chicago, pp. 25-31, May 31-June 2 1978.

[Coo79] D. Cooper, "Maximum likelihood estimation of markov-process blob boundaries in noisy images," *IEEE Trans. PAMI,* vol. PAMI-1, pp. 372-384, Oct. 1979.

[Cor71] D. Corcoran, *Pattern Recognition,* Baltimore MD: Penquin, 1971.

[Cor70] T. Cornsweet, *Visual Perception,* N.Y.: Academic Press, 1970.

[Dav58] W. Davenport and W. Root, *Random Siganls and Noise,* N.Y.: McGraw-Hill, 1958.

[Dav72] L. Davisson, "Rate distortion theory and application," *Proc. IEEE,* vol. 60, pp. 800-808, July 1972.

[Dud73] R. Duda and P. Hart, *Pattern Classification and Scene Analysis,* N.Y.: Wiley, 1973.

[Edw73] J. Edward and M. Fitelson, "Notes on maximum-entropy processing," *IEEE Trans. Inf. Theory,* vol. IT-19, pp. 232-234, Mar. 1973.

[Fan63] R. Fano, "A hueristic discussion of probabilistic decoding," *IEEE Trans. Inf. Theory,* vol. IT-9, pp.64-74, Apr. 1963.

[Fis73] M. Fischler and R. Elschlager, "The representation and matching of pictorial structures," *IEEE Trans. Computers,* vol. C-22, pp. 67-92, Jan. 1973.

[For73] G. Forney, Jr., "The Viterbi algorithm," *Proc. IEEE,* vol. 61, pp. 268-278, Mar. 1973.

[For74a] G. Forney, Jr., "Convolutional codes II: maximum likelihood decoding," *Inf. Control,* vol. 25, pp.222-266, July 1974.

[For74b] G. Forney, Jr., "Convolutional codes III: sequential decoding," *Inf. Control,* vol. 25, pp.267-297, July 1974.

[Gal65] R. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. Inf. Theory,* vol. IT-11, pp. 3-18, Jan. 1965.

[Gal68] R. Gallager, *Information Theory and Reliable Communication,* N.Y.: Wiley, 1968.

[Gei73] J.M. Geist, "Search properties of some sequential decoding algorithms," *IEEE Trans. Commun. Technol.,* vol. COM-19, pp.415-419, Aug. 1973.

[Gri73] R. Griffith, "Automatic boundary and feature extraction for the estimation of left ventricular volume in man from serial selective angiocardiograms," D. Eng. thesis, Rensselaer Polytechnic Institute, Aug. 1973.

[Hab72] A. Habibi, "Two-dimensional Bayesian estimate of images," *Proc. IEEE,* vol. 60, July 1972.

[Hac75] D. Haccoun and M.J. Ferguson, "Generalized stack algorithms for decoding convolutional codes, *IEEE Trans. Inf. Theory,* vol. IT-21, pp.638-651, Nov. 1975.

[Ham83] R. Hamming, *Digital Filters,* N.J.: Prentice Hall, 1983.

[Han82] F. Hansen and H. Elliot, "Image segmentation using simple Markov field models," *CGIP,* vol. 20, pp. 101-132, 1982.

[Har68] P. Hart, N. Nilsson, B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Systems Sci. Cybernet.,* vol. SSC-4, pp. 100-107, 1968.

[Har84] R. Haralick, "Digital step edges from zero crossing of second directional derivatives," *IEEE Trans. PAMI,* vol. PAMI-6, pp. 58-68, Jan. 1984.

[Has80] M.Hassner and J. Sklansky, "The use of Markov random fields as models of texture," *CGIP,* vol. 12, pp. 357-370, 1980.

[Hue73] M. Hueckel, "A local visual operator which recognizes edges and lines," *J. ACM,* vol. 20, pp. 634-647, Oct. 1973.

[Huy80] N. Huyn, et al., "Probalistic analysis of the complexity of A^*," *Artif. Int.* vol. 15, pp. -254, Dec. 1980.

[Jel69] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM J. Res. and Dev.,* vol. 13, pp.675-685, Nov. 1969.

[Kai74] T. Kailath, "A view of three decades of linear filtering theory," *IEEE Trans. Inf. Theory,* vol. IT-20, pp. 146-181, Mar. 1974.

[Kan80] L. Kanal, "Markov mesh models," *CGIP,* vol. 12, pp. 371-375, 1980.

[Kir71] R. Kirsch, "Computer determination of the constituent structure of biological images," *Comp. and Biomed. Res.,* vol. 3, June 1971.

[Kit81] L. Kitchen and A. Rosenfeld, "Edge evaluation using local edge coherence," *IEEE Trans. Sys. Man & Cyber.*, vol. SMC-11, pp. 597-605, Sept. 1981.

[Kul59] S. Kullback, *Infomation Theory and Statistics*, N.Y.: Wiley, 1959.

[Kun82] M. Kunt, "Edge detection: a tutorial review," *Proc. IEEE Intern. Conf. on Acoustics, Sp. and Sig. Proc.*, vol. 2, 1982.

[Lin83] S. Lin and D. Costello Jr., *Error Control Coding: Fundamentals and Applications*, N.J.: Prentice-Hall, 1983.

[Mac81] R. Machuca and A. Gilbert, "Finding edges in noisy scenes," *IEEE Trans. PAMI*, vol. PAMI-3, pp. 103-111, Jan. 1981.

[Mak75] J. Makhoul, "Linear prediction: a tutorial review," *Proc. IEEE*, vol. 63, pp. 561-580, Apr. 1975.

[Mar76a] D. Marr, "Early processing of visual information," *Phil. Trans. R. Soc. Lond.*, vol. B 275, pp.483-524, 1976.

[Mar76b] A. Martelli, "An application of heuristic search methods to edge and contour detection," *Commun. ACM*, vol. 19, pp. 73-83, 1976.

[Mar80] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc. Lond.*, vol. B 207, pp.187-217, 1980.

[Mas72] J.Massey, "Variable - length codes and the Fano metric," *IEEE Trans. Inf. Theory*, vol. IT-18, pp.196-198, Jan. 1972.

[McE77] R. McEliece, *The Theory of Information and Coding*, Reading, MA: Addison-Wesley, 1977.

[Mel78] J. Melsa and D. Cohen, *Decision and Estimation Theory*, N.Y.: McGraw-Hill, 1978.

[Mod76] J. Modestino, G. Ashkar, R. Fries, H. Kaufman, "Computer evaluation of dynamic left ventricular volume from serial angiograms," *Proc. Computers in Cardiology*, St. Louis MO, pp. 211-218, Oct. 7-9 1976.

[Mod77] J. Modestino and R. Fries, "Edge detection in noisy images using recursive digital filtering," *CGIP*, vol. 6, pp. 409-433, 1977.

[Nav80] R. Navatia and K. Babu, "Linear feature extraction and description," *CGIP*, vol. 13, pp.257-269, July 1980.

[Nay82] A. Naylor and G. Sell, *Linear Operator Theory in Engineering and Science*, N.Y.: Springer-Verlag, 1982.

[New77] W. Newman, "A new method of multidimensional power spectral analysis," *Astron. Astrophys.*, vol. 54, pp. 369-380, 1977.

[Nil71] N. Nilsson, *Problem-Solving Methods in Artificial Intelligence*, N.Y.: McGraw-Hill, 1971.

[Nis65] A. Nishikawa, R. Massa, and J. Mott-Smith, "Area properties of television pictures," *IEEE Trans. Inf. Theory*, vol. IT-11, pp. 348-352, 1965.

[Pet54] W. Peterson, T. Birdsall, W. Fox, "The theory of signal detectability," *IRE Trans. Inf. Theory*, vol. PGIT-4, 1954.

[Pra78] W. Pratt, *Digital Image Processing*, N.Y.: Wiley, 1978.

[Rob65] L. Roberts, "Machine perception of 3-dimensional solids," *Optical and Electro-Opt. Inf. Processing J.*, pp. 159-197, 1965.

[Ros68] A. Rosenfeld and J. Pfaltz, "Distance functions on digital pictures," *J.ACM*, vol. 13, pp.471-494, 1968.

[Ros70] A. Rosenfeld, "Connectivity in digital pictures." *J.ACM*, vol. 17. pp. 146-160, Jan. 1970.

[Ros71] A. Rosenfeld and M. Thurston, "Edge and curve detection for visual scene analysis," *IEEE Trans. Computers*, vol. C-20, pp. 562-569, 1971.

[Ros76] A. Rosenfeld and A. Kak, *Digital Picture Processing*, N.Y.: Academic Press, 1976.

[Ros83] S. Ross, *Stochastic Processes*, N.Y.: Wiley, 1983.

[Ros84] A. Rosenfeld, "Picture processing: 1983," *CVGIP*, vol. 26, pp. 347-393, June 1984.

[Sav66] J.E. Savage, "Sequential decoding - the computation problem," *Bell Syst. Tech. J.*, vol. 45, pp.149-175, Jan. 1966.

[Sch80] M. Schwartz, *Information, Transmission, Modulation and Noise,* N.Y.: McGraw-Hill, 1980.

[Sil57] R. Silverman, "Locally stationary random processes," *IRE Trans. Inf. Theory,* vol. IT-3, pp. 182-187, Sept. 1957.

[Suc82] R. Suciu and A. Reeves, "A comparison of differential and moment based edge detectors," *Proc. IEEE Comp. Soc. Conf. on Patt. Recog. and Image Proc.,* pp. 97-102, June 1982.

[Van68] H. Van Trees, *Detection Estimation and Modulation Theory, Part I,* N.Y.: Wiley, 1968.

[Vit79] A.J. Viterbi and J.K. Omura, *Principles of Digital Communication and Coding,* N.Y.: McGraw-Hill, 1979.

[Wol38] H. Wold, *A Study in the Analysis of Stationary Time Series,* Uppsala: Almquist and Wiksells, 1938.

[Wil71] L. Wilkins and P. Wintz, "Bibliography on data compression, picture properties, and picture coding," *IEEE Trans. Inf. Theory,* vol. IT-17, pp. 180-199, 1971.

[Won71] E. Wong, *Stochastic Processes in Information and Dynamical Systems,* Huntington, N.Y.: Krieger, 1971.

[Woo72] J. Woods, "Two-dimensional discrete Markovian fields," *IEEE Trans. Inf. Theory,* vol. IT-18, pp. 232-240, 1972.

[Woo76] J. Woods, "Two-dimensional Markov spectral estimation," *IEEE Trans. Inf. Theory,* vol. IT-22, pp. 552-559, Sept. 1976.

[Woz61] J.M. Wozencraft and B. Reiffen, *Sequential Decoding,* Cambridge, MA: MIT Press, 1961.

[Woz65] J. Wozencraft and I. Jacobs, *Principles of Communication Engineering,* N.Y.: Wiley, 1965.

[Zig66] K. Zigangirov, "Some sequential decoding procedures," *Probl. Peredachi Inf.,* vol. 2, pp. 13-25, 1966.

[Zuc76] S.W. Zucker, R.A. Hummel, and A. Rosenfeld, "An application of relaxation labeling to line and curve enhancement," *IEEE Trans. Comp.,* vol. C-26, pp.394-403, Apr. 1976.

[Zuc78a] S.W. Zucker, E.V. Kirshnaumurthy, and R. Harr, "Relaxation processes for scene labeling: convergence, speed, and stability," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp.41-48, Jan 1978.

[Zuc78b] S.W.Zucker and J.L. Mohammed, "Analysis of probabilistic relaxation labeling processes," *Proc. IEEE Conf. on Image Processing and Pattern Recognition*, Chicago, IL, pp.307-312, 1978.