May 1995 **LIDS-TH-2298**

Research Supported By:

Army Research Office
ARO DAAL03-92-G-0115

Automated Text Recognition

Mohamad A. Akra

May 1995

LIDS-TH-2298

Sponsor Acknowledgments

Army Research Office
ARO DAAL03-92-G-0115

Automated Text Recognition

Mohamad A. Akra

This report is based on the unaltered thesis of Mohamad A. Akra submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the Massachusetts Institute of Technology in August 1993.

This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with research support gratefully acknowledged by the above mentioned sponsor(s).

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Automated Text Recognition

by

Mohamad A. Akra

B.E., American University of Beirut (1986)
S.M., Massachusetts Institute of Technology (1988)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 1993

Author .....
Department of Electrical Engineering and Computer Science
August 6, 1993

Certified by .....
Sanjoy K. Mitter
Professor of Electrical Engineering
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students

Automated Text Recognition

by

Mohamad A. Akra

Submitted to the Department of Electrical Engineering and Computer Science
on August 6, 1993, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Automated text recognition is a difficult but important problem. It can be summarized as: how to enable a computer to recognize letters and digits from a predefined alphabet, possibly using contextual information. Various attempts at solving this problem, using different selections of features and classifiers, have been made. Human performance has been achieved in accuracy by automated text recognition systems, and has been bypassed in speed for the case of single size, single font, high quality, known layout, known background, text. When one or more of the above parameters are changed, the problem becomes increasingly difficult. In particular, attaining human performance in recognizing cursive script of varying size, varying style, unknown layout, unknown background is far from the reach of todays' algorithms, despite the continuous research effort for almost four decades. In this report, we analyze the problem in detail, present the associated difficulties, and propose a coherent framework for addressing automated text recognition.

Thesis Supervisor: Sanjoy K. Mitter
Title: Professor of Electrical Engineering

To:

The Engineer

Acknowledgments

"Then when (Solomon) saw it placed firmly before him, he said: "This is by the grace of my Lord! To test me whether I am grateful or ungrateful! And if any is grateful, truly his gratitude is (a gain) for his own soul; but if any is ungrateful, truly my Lord is Free of all Needs, Supreme in Honour!". Surah 27, Ayah 40, translated.

All praises and thanks belong to God, Most High and Most Great alone. It was solely through His beneficence that this thesis was completed. He is the one who inspired the intelligent ideas of this thesis, and He is the one who blessed me with the help of many people.

There are many people to thank for their help in completing this thesis. First, I acknowledge the help of my research supervisor Professor Sanjoy Mitter. I cite him first because I owe him much of the credit for this work. He showed me the way to innovation and taught me to be ever conscious of the fundamentals. His patience with me was beyond limits. Yet he also knew when it was appropriate to apply pressure. Choosing Professor Mitter as my thesis supervisor was one of the smartest choices I have made in my life.

I also appreciate the help of Professor Dahleh. I thank him for the time he spent reading my thesis, his encouragement, support, and his constructive comments on how to make the thesis more cohesive and structured. I still remember one of the gems of advice he gave me when I began my thesis: "The best way to do research on an old problem is to work on it *before* reading any related article!"

I am indebted to Professor Gallager who gave me excellent suggestions (one of them is to question the metric idea). He also insured that whatever I said about information theory was correct and precise. I am especially impressed by the care with which he reviews documents. He was even able to spot an error in the bibliography section!

When it comes to friends and colleagues, I am blessed with two sources of continuous support. The first source is Mahbub, who prayed for me sincerely, and proofread my thesis. His phone calls every morning were of unmatched value. The second source

is Walid, who spent long hours on the telephone (on his expense!), and on e-mail, discussing my thesis problem and encouraging me to continue, especially during the tough times...

I would also like to express my deepest gratitudes to my brothers Abu Zayd, Adham, Adnan, Afif, Emadeddine, Hussain, Ibrahim, Saad, Sabri, Suheil, Wassim, Yusuf, Ziad, and all my other good friends. They always remained supportive, even when I neglected to return their many, many phone calls in the last months of my PhD.

Three negative acknowledgements should be stated at this point. Ahmad, Seerah and Tasneem cooperated destructively to keep me awake both days and nights. However, their sheer joy and ecstatic welcome over my arrival home every night (or sometimes every morning!) makes up for all of this.

Finally, there is always the one person who hides behind the scenes, but who by far manages to provide the most. With all my due respect and consideration, I dedicate this thesis to the one person who suffered the most before it was accomplished: my wife.

Contents

List of Figures

List of Tables

Chapter 1

Automated Character Recognition

1.1 Introduction

The field of automated pattern recognition was originally stimulated by studies in optical character recognition. Optical character recognition means the ability of mapping grey-level images of characters into equivalent ASCII code[1]. The achievement of this goal implies the automation of time-consuming – yet important – tasks such as data entry, check processing and mail interpretation. However, the simplicity of the problem statement belies the complexity of the problem. This complexity is due – mainly – to two factors: the variability in writing styles and sizes across different writings, and the a priori uncertainty in the text layout. Two major approaches to the problem can be identified in the literature: the *statistical* approach and the *linguistic* approach. However, neither approach has yielded the desired performance needed in the above cited applications, especially if the documents are handwritten. The main difficulty encountered in both approaches is the question of *representation*, or *feature selection*. What is a character? a binary image? a collection of strokes? or a set of numbers resulting from morphological operations? Apparently, this question can only be answered through a consideration of the text generation process and the structure of the uncertainty present. We propose applying an *information theory*

[1] American Standard Code for Information Interchange.

framework to understand the problem and try evaluating previous approaches within such a framework. The second major encountered difficulty is the question of *similarity*. What similarity measure best models our perception? A weighted Euclidean distance? A Hausdorff distance? or a set of rules? It is true that the questions of representation and similarity are closely tied. Nevertheless, we argue that the classical notion of looking for a metric (in the mathematical sense) is not the proper notion for recognition.

This chapter is a general introduction to the problem, and a discussion of the information theory framework. This framework will be used in the next chapter to organize the literature survey. The comparative discussion between our work and other people's work will be scattered throughout the thesis.

1.2 Problem Description

Given some text written on paper, we would like to be able to scan the paper and recognize the text. In other words, we want an algorithm that transforms grey-level bitmapped images of text into its corresponding ASCII code. The criterion of success is the ability to achieve human-like performance in accuracy, and to do so with reasonable speed, where "reasonable" differs in meaning from application to application. The speed is measured as the number of characters, or predefined group of characters (such as words, addresses, etc.) recognized per second, whereas the accuracy is measured as the percentage of misclassified characters, or misclassified groups of characters.

1.2.1 Handwritten vs Machine Print

It is convenient to define two main classes of text recognition problems: *handwritten* and *machine printed*. The division is primarily due to the different types of variabilities encountered in each class. In machine printed text, variability is due to the vast collection of fonts as well as the quality of print in the printing mechanisms (dot matrix, inkjet, laser, etc.). In handwritten text, variability is due to the loss of

synchronism between the muscles of the hand as well as the variation of styles due to several factors, including but not limited to: education, mood, culture, etc.

1.2.2 On-Line vs Off-Line

The handwritten version of the problem can be further subdivided into two main categories: *on-line* and *off-line*. In the on-line category, the dynamic process of writing is captured, usually via a digitizer or a tablet, as is the case with current pen-based computer systems. In the off-line category, this dynamic information is lost and only the static information is available through a scanning process. While dynamic information can be very helpful in improving the performance of the recognition system, the true advantage of an on-line system is the possibility of *adaptation* that it provides. The writer adapts to the system, resulting in a significant improvement of the recognition accuracy.

1.2.3 Handprint vs Cursive

Alternatively, the handwritten text recognition problem can be subdivided into two other main categories: *handprinted* and *cursive*. In the handprinted category, the characters of a word are isolated and disconnected. In the cursive category, the characters are connected by ligatures. It turns out that *segmenting* the characters is not an easy problem, and that is the main reason why the performance results reported in recognizing handprinted text are higher than those reported in recognizing cursive text.

1.2.4 Layout and Background

Finally, there are two problems that add to the difficulty of off-line automated text recognition: the *layout* problem and the *background* problem. The layout problem is a consequence of the lack of a priori knowledge about the position and orientation of the text. A typical magazine page might have pictures, and the text might run across several columns in an unpredictable way. Similarly, an artist might design his text

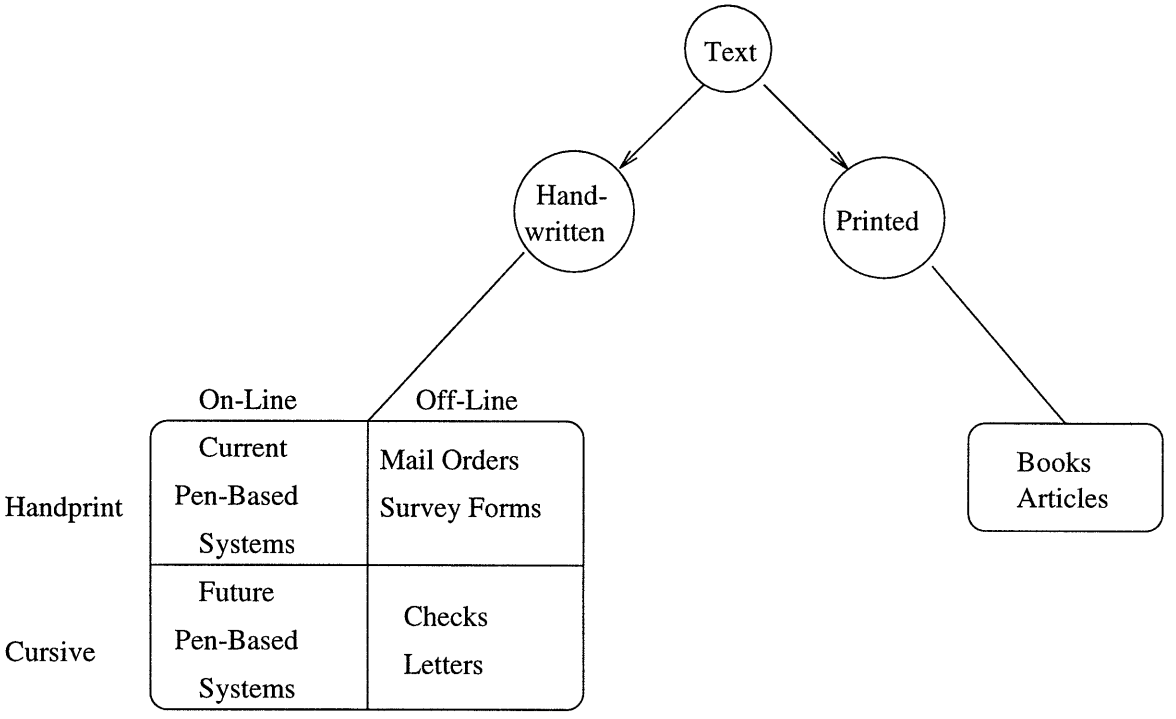


Figure 1-1: Text recognition : categories and applications

on a very complicated layout.

The background problem is a consequence of the different possible background colors in a typical page, and the existence of different image design techniques (e.g., partial reverse video, blurring, etc.), all of which complicate the extraction of the actual text.

1.2.5 Our Focus

In our research program, we do not attempt to solve the background problem. Rather, we will assume that the text is available in grey level, on a white background, and that the text layout is known.

1.3 Historical Background and Current Status

1.3.1 Brief History

The order of difficulty in text recognition increases as follows:

1. Off-line, machine printed, single font

2. Off-line, machine printed, multifont (or omnifont)

3. On-line, handprinted

4. Off-line, handprinted

5. On-line, cursive

6. Off-line, cursive

Historically, the research on different versions of the problem started in the same order. The lucid book "History of OCR"[41], describes the scientific and commercial achievements in this field, up to 1982.

1.3.2 Current Commercial Packages

Currently, PC commercial packages exist to recognize off-line, machine printed, omnifont text, such as Omnipage by Caere Inc., TypeReader by Expervision, and WordScan Plus by Calera[19]. There are also packages to recognize off-line handprinted text such as IDEPT by HNC, NestorReader by Nestor Applied Systems, Teleform by Cardiff Software, and PaperKeyboard by Datacap. However, off-line packages have not yet reached the level of accuracy that permits fully replacing the process of hand keying or typing the data.

As for commercial on-line systems, Scriptwriter by Data Entry Systems recognizes handprinted text. Also, recently, an application program written by Paragraph and implemented on Apple Newton, showed good performance in recognizing on-line cursive text.

Finally, the author is not aware of any commercial package that can reliably recognize off-line cursive text.

1.4 Applications

The importance of text recognition arises from several applications. In the past, the most common applications were "Reader for the Blind"[2] and "Data Entry". Today, "Data Entry" still remains an important application, but "Reader for the Blind" and such applications have been superseded by "Mail Interpretation" and "Check Processing".

1.4.1 Mail Interpretation

Nearly 166 billion mail pieces are collected yearly by the U.S.Postal Service. Almost 15% of this mail volume has handwritten addresses. Currently, the post office uses an optical character reader that can read printed addresses only, with 60-70% recognition rate . Nearly 100,000 employees are assigned to sort the handwritten-address mail, as well as the rejected 30-40% of the printed-address mail. While the productivity of the OCR reader is 5000 accepted pieces/man-hr, that of the manual sorting is only 650 pieces/man-hr. Automating the process of mail interpretation and sorting can result in a faster, more reliable and less expensive postal system. Estimated yearly savings from automating the handwritten addresses mail alone are $125 million[14].

1.4.2 Check Processing

Americans write 50 billion checks a year. Currently, banks require that two people review every check before deducting the amount on the check. Automating the process can – at least – reduce the number of reviewers to one, thereby speeding up banking transactions and reducing the processing cost[32].

1.4.3 Data entry

More than 200 million tax forms were filed in 1990. Nearly 112 million came from individual taxpayers filing Forms 1040, 1040A or 1040EZ. A little less than 40,000

[2]In fact, the first OCR patent awarded in US was in 1809, and it was for a reading device to aid the blind[41].

employees were working in the Data Processing Operations. In order to provide taxpayers the prompt and responsive service they desire, a Tax Systems Modernization (TSM) program was established in 1990. By using electronic filing, optical character recognition and other technologies over the next ten years, the IRS hopes to[42]:

- eliminate millions of unnecessary contacts with taxpayers needed now to correct name, address and social security number mismatches;
- provide copies of tax returns to taxpayers and IRS employees in less than one day rather than the current 45-day average;
- reduce by one quarter the time it takes to process cases.

Other data entry applications include: credit cards forms, mail orders, questionnaires, etc.

1.5 Suggested Solutions

1.5.1 Previous Attempts

Traditionally, two approaches are known for pattern recognition: the *statistical* approach and the *linguistic* approach. The idea in both approaches is to select a set of measurements called *features* (e.g., the density of black pixels, the number of loops in the character, the type and position of a stroke, etc.), and a set of decision rules, constituting a *classifier*. Due to the variability across different samples of the same pattern class, these features will be unknown, i.e., modeled as random variables. These variables define the *feature space*. In the statistical approach, the space is partitioned, according to the set of rules, into regions corresponding to different patterns, i.e., one region for A, another for B, etc. Given an unknown pattern, the procedure is to:

1. Extract the vector of features
2. Find which region it belongs to

3. Assign to the pattern the label of that region.

In the linguistic approach, a pattern class is considered to be a set of features generated by a non-deterministic finite state machine, a Markov process, or a push-down automata, etc. Given an unknown pattern, the procedure is:

1. Extract the set of features
2. Check which machine potentially generated it
3. Label the unknown pattern accordingly

Therefore, the main difference between the two approaches lies in the structure of the classifier, which, in a sense, relies on our view of what defines a character. In the statistical approach, possible classifiers are: *nearest mean, Fisher, neural network*, or *nearest neighbor*[11]. In the linguistic approach, the classifier is usually a machine matching classifier. The statistical approach is older and is more commonly used with simple patterns such as characters. The linguistic approach is preferred in the case of complex patterns, such as 3D images.

1.5.2 Encountered Difficulties

The difficulty in both approaches lies in choosing a useful set of features, which is sometimes termed the *problem of representation.* In a handwritten text recognition problem, for instance, choosing the pixel values of the text image as features is not a useful strategy. While the set of pixel values marks a complete representation, in the sense that any other representation can be derived from it, it is not a convenient representation to deal with. Variations in the samples of a handwritten pattern would cause a wide scale correlation among the features which, consequently, complicates the design and analysis of the classifier. We make the observation that the representation problem should be approached through an analysis of the noise source. By noise we mean the disturbance or inaccuracy incurred on the text during the text generation process, in either the handwritten or the machine-printed case.

1.5.3 Proposed Approach

Since we have reduced the problem of feature extraction to that of noise analysis, the analogy with signal transmission theory becomes apparent. This analogy, however, is not complete. While most of the signals studied in signal transmission theory are functions of one time variable t, a character is a function of two space variables x and y. Moreover, while in most signal applications, the interesting noise is amplitude noise (except in array signal processing), the most interesting noise in text is domain noise. Consequently, the theory of Bayesian detection and estimation that relies on an additive noise model does not carry over trivially to the case of text recognition. Nevertheless, we argue that several fundamental principles apply. The following is a non exhaustive list of these principles:

1. The recognition problem is best approached through a study of deformations of characters rather than through a study of the characters themselves. In other words, instead of searching the characters for features that are robust to noise, we propose to investigate the noise itself and see how it affects the various characters. This principle is explored in Chapter 3.

2. A hierarchy of abstractions is a useful tool to avoid computational complexity without severely degrading performance. This tool was used in coding theory when abstracting the triplet modulator/channel/demodulator as a discrete channel with certain transition probabilities. The abstraction principle is explored in Appendix A1.

   We note, however, that the hierarchy of abstractions is more than just a complexity management tool. It is an *inherent* component of the recognition process itself. Recognition, when viewed properly, is nothing but a representation at a suitably abstract level.

3. In coding theory, a major step in improving the decoder design was accomplished by merging the digital data demodulator with the channel decoder. This means that the whole waveform of bits is decoded directly into a set of

bits, whereas previously, the common strategy was to decode individual bits first, and then correct the erroneous ones by using information in other bits. The same concept is adopted in Chapter 4 where we try to decode the character from the set of pixels directly, i.e., without decomposing it into primitives, such as strokes or bays.

The ultimate performance would be achieved by decoding words directly from pixels. However, the number of words we have in the dictionary is usually very large, signaling the necessity of an abstraction at the level of the characters.

4. If the information to be transmitted is higher than the capacity of the channel, error free recognition is impossible. An illustrative example of this principle in the context of text recognition is cited at the end of this chapter.

1.5.4 A Note on the Literature

By reviewing the literature, we find that more than 900 papers were published in the last decade only(1983-1992). An additional 550 articles were published in the period before 1965[29]. By simple interpolation, we estimate that around 550 articles were published in the period (1966-1982). This makes the estimated overall number of articles above 2,000. However, the mass of publications is plagued by the fact that authors usually concentrate on only a few aspects of the text recognition system and often name their approach after the aspects on which they concentrated. This habit of researchers may lead the reader to think that the morphological approach, for instance, is necessarily different than the neural network approach. Careful study reveals that morphology is used in the preprocessing and the feature extraction phase, whereas neural networks are used in the classification phase. In that respect, they are clearly not mutually exclusive. Hence, a framework for comparative analysis becomes, not only a helpful tool, but also a vital necessity. In the following sections, we will attempt to establish that framework, and use it to survey the relevant literature.

1.6 The Big Picture

1.6.1 Description

Our main idea is to view the task of text recognition as part of a larger system: a system consisting of a source, a channel, and a destination. The source is the entity that generates the text, which could be a human mind or an electronic storage device. The channel is a combination of the medium that disturbs the text (such as the human hand or the printer) and the input device (such as the scanner or the digitizer). The destination is usually a storage device, receiving the text for further processing.

Casting the character recognition problem in the above framework helps us evaluate critically previous work on character recognition. The information theory framework also points to us that it is important to define what the channel is in the character recognition problem and how we should model the uncertainties present in the channel.

1.6.2 Survey

Although not new, the above idea has been commented on by only a few researchers, and their analyses have been very brief. In the context of handwriting recognition, Mermelstein and Eden say[29]:

> We assume that the handwriting generating system of the hand and arm acts as a transducer, receiving by means of the nerve fibers a description of the graphical pattern to be executed. The system is noisy in that it introduces variations in the outputs corresponding to input signals thought to carry the same information.

Other authors came close to the same idea. J.Kulikowski[27] drew a block diagram of the same flavor as the one we have in Figure 1-2. M. J. Usher[46] dedicated a complete section for OCR as an application of information theory. Recently, P. A. Chu wrote a PhD thesis on the "Applications of Information Theory to Pattern Recognition"[6]. Several articles (e.g., Casey[4]) used some elements of information theory, such as

entropy and mutual information, in their course of analysis. Finally, an extensive literature on pattern recognition can be found in journals dealing with information theory such as *IEEE Transactions on Information Theory.*[3]

1.6.3 Missing Link

However, in all of the above references, with the exception of Chu[6], the observations made regarding the relation between information theory and text recognition were not supported with the detailed analysis they deserve. Some authors (e.g., Vossler[47], Hull[21], Forney[9], and Neuhoff[35]), for instance, attribute the noise to the optical character recognizer! without any mention of the role of the hand or the printer. Nowhere could we find a conceptual view and detailed undertaking of the total text recognition problem in the light of information theory.

Concerning the work of Chu[6], it is mostly concerned with the design of classification trees for general pattern recognition problems. It does not make use of the aspects peculiar to the text recognition problem, such as the modeling of uncertainties present in the generated text.

1.7 Framework Overview

1.7.1 Concepts and Sources

Usually, text is generated to convey information about a certain *concept.* When a person writes a check, he writes some text that indicates the amount to be paid. When a person wants to send a letter, he writes some text that describes the destination to which this letter should be delivered. When a computer prints an english sentence, the text is used to describe a particular idea. In each of the above, we expect the source output to be a *concept* drawn from a *class* of concepts, e.g., a check amount

[3]The reader should observe that in his mathematical theory of communication, Shannon was using the channel to model the transmission media, be it wires or air. He was not using the channel to model the vocal tract, for speech recognition, or the hand, for text recognition. That is why we have not cited his work in this section.

between $1 and $10,000.00, an address out of 118 million delivery points in the US, or a collection of english words subject to certain grammatical rules.

1.7.2 Characters and Source Encoding

The generated text consists of *characters* laid out on the page. A character can be a letter, a digit, a blank space, or a punctuation mark. The same collection of characters is used to describe different concepts. In information theory, this idea is called *source encoding*[4], and it helps in representing a very large set, e.g., the set of US addresses, using strings of relatively few symbols, e.g., the english alphanumerics. Note that characters need not be written on the same baseline, and characters that have different size should not be treated the same as the normal size characters (imagine a small print on a bank check). However, no information about the geometric layout was used in the literature in a structured way, other than using few heuristics in preprocessing to eliminate these undesirable cases. In short, text was viewed simply as a *string of characters.*

1.7.3 Feature Selection and Channel Encoding

Each character is described, not necessarily uniquely, as a geometrical shape in the plane. These clean geometrical shapes are not observed on paper, except in cases of neat printing or handwriting. What is observed usually is a noisy version of these shapes. Ideally, one should deal with these shapes directly, applying - if we may call it - *geometric* information theory. However, little is known in the theory of two dimensional vector noise, i.e., noise having two components[5], both functions of x and y. Because of that, researchers have attempted to transform the problem into a one-dimensional problem, and to apply classical pattern recognition techniques (effectively dealing now with t instead of x and y). They argued that, despite the noisy variations, there seems to be certain aspects of these geometrical shapes that are invariant to

[4]More precisely, it is called *outer channel encoding.* See Forney[10].

[5]This noise should not be confused with the amplitude noise studied in image processing, which has one component only.

noise, like having an ascender in the letters b, d, or t, or a descender in p, q, or g. These aspects were considered very useful, and were usually called *features.* The problem of feature selection is still an open one, except in the case of selection from a predetermined, finite set of features. The process of identifying the character as a string of features is equivalent, in information theory, to *channel encoding.*

1.7.4 Writing, Digital Data Modulation, and Channel

We emphasize here that researchers in this field were not really formalizing the problem as we are doing now, but they merely proposed reducing the problem to determining the feature space and classification. Our attempt to formalize their work in the context of information theory is only to show the weak parts of the classical arguments. One weak part has already been exposed twice, namely the "serialization" of a two dimensional problem at two levels. However, we will continue in what follows along this path, for the purpose of surveying the literature in an organized manner.

To proceed, the input to the channel was not viewed as a geometrical shape, but rather as a vector of features. The features (and not the geometrical shape) were considered translated into paper by the motion of either a pen or a printer head, a process equivalent to *digital data modulation* in information theory. During this process, inaccuracies are observed. In the case of handwriting, these inaccuracies are due, to a certain extent, to the coordination inaccuracies of the muscle system involved. The more degrees of freedom involved, the larger the departure from the planned trajectory[37]. In the case of machine printing, the inaccuracies are due to the imperfect response of the printer head to sudden changes in speed, bad ribbons, bent keys, etc.[6]

1.7.5 Feature Extraction and Digital Data Demodulation

At this stage, the character in consideration is deformed. To recognize the character, the classical approach was to start by *feature extraction*, i.e., measuring from the

[6]Even though these accuracies may not be even detected by the human eye.

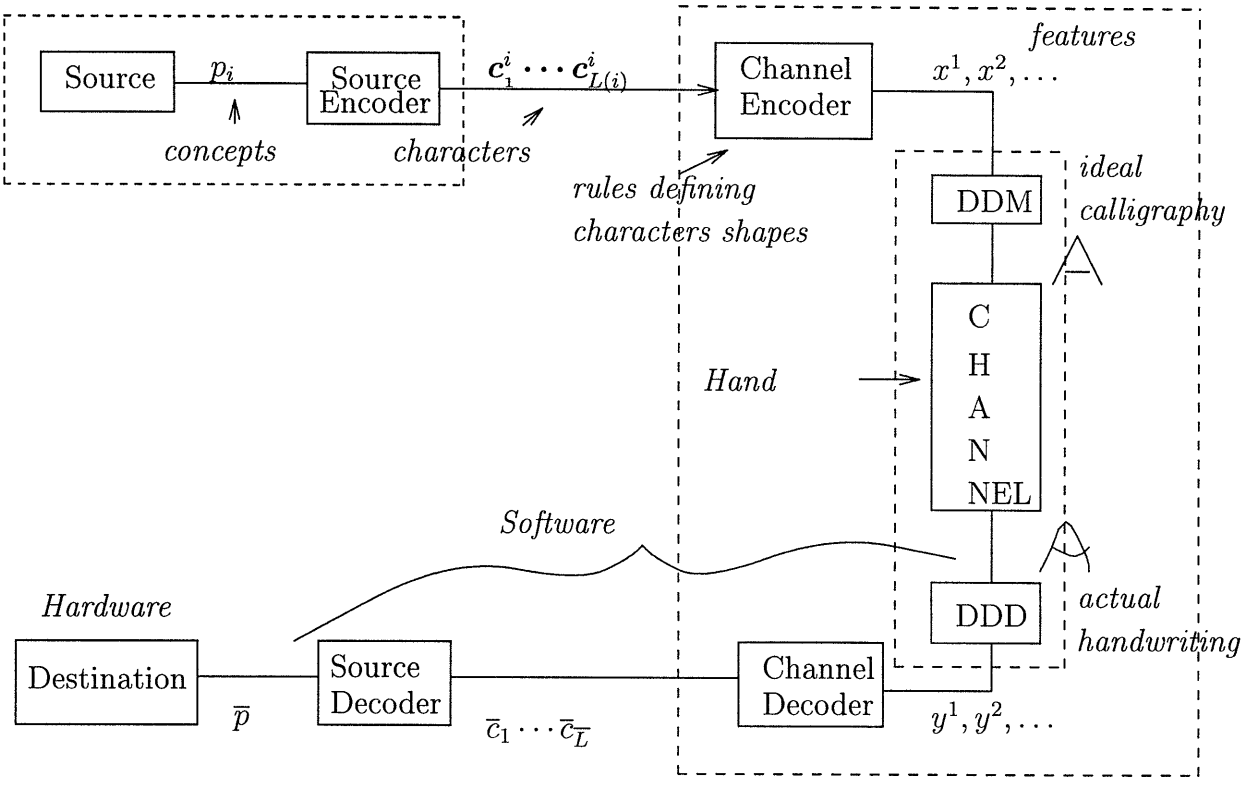


Figure 1-2: Mathematical model for text generation and interpretation

deformed geometrical shape those attributes that were identified as being noise invariant. This step was done in a module equivalent to the one known, in information theory, as a *digital data demodulator*. By inspecting the extracted features, a decision was made as to what character they belong. The set of rules used in the decision process constituted a *classifier*, which was equivalent to a *channel decoder*.

1.7.6 Postprocessing and Source Decoding

Consecutive characters are, generally, *not* independent. This contextual dependence was exploited to improve recognition, by correcting for characters that were erroneously deleted, inserted, or substituted. This *postprocessing* phase was carried by what is equivalent, in information theory, to a *source decoder*[7].

[7]More precisely, to an *outer channel decoder*.

1.7.7 Destination

The resulting concept from the output of the source decoder is sent to a *destination*, which is usually a storage device (such as in data entry applications), a control device (such as in mail sorting), or a processing device (such as in banking applications).

The previous discussion is summarized in Figure 1-2. In the next chapter, the various modules of our framework will be discussed in more detail.

1.8 Consequences of the Framework

1.8.1 Intelligibility: Definition

Quite a few intuitive facts can be explained within our framework of analysis. Consider, for instance, *intelligibility* of people writings. This is, traditionally, considered to be a subjective concept that cannot be quantified. In our framework, it is natural to associate this concept with hand *capacity*. More precisely, we could define *intelligibility* to be $C - H(X)$, where C is the hand capacity and $H(X)$ is the source entropy per concept.

1.8.2 Illustration 1: Doctor Example

To illustrate this association, we will consider three cases:

1. By virtue of the converse to the coding theorem[13], if the channel capacity is less than the source entropy, it is impossible to correctly decode the source message. Consider a random person trying to read (i.e., decode) a prescription written by a doctor. Because the doctor's hand capacity is less than 1 bit/char, the entropy of the english language[8], it will be impossible for a general person to understand the prescription.

2. On the other hand, when a pharmacist attempts to read the same prescription, the source will be the english language constrained in the medical context.

[8]See Shannon and Weaver[43] for a series of experiments to estimate the entropy of english text.

This constraint decreases the source entropy below the doctor's hand *capacity*. Hence, it becomes possible for the pharmacist to decrypt the prescription.

3. Finally, when the doctor's secretary tries to read his writing, the source for her will be the english language, the same as for the random person. However, she has the advantage of partial information about the noise correlation in the doctor's writings (she has seen a lot of his prescriptions). Hence, for her, the capacity of the doctor's hand model will be higher, which results in a higher intelligibility.

Therefore, it is meaningful to define the intelligibility of handwriting as the *capacity* of the hand that generated the writing minus the source entropy. Further justification of this definition results if we note that trying to write faster produces a less intelligible writing. This is again a consequence of the converse to the coding theorem.

1.8.3 Illustration 2: On-Line vs Off-Line Performance

Another intuitive fact that can be explained in our framework is the fact that system performance in on-line text recognition should be higher than the performance in the corresponding off-line version. Since in an off-line context, the dynamic information is lost, the capacity of the combined channel (hand/page) is less than the capacity of the combined channel (hand/tablet). Using the noisy channel theorem, we conclude that this decrease in the channel capacity leads to a deterioration in the recognition performance.

1.9 Conclusions

Information theory is a powerful tool. When used as a framework for investigating the field of text recognition, the benefits are: conceptual understanding and a unified language. The next chapters confirm this assertion.

1.10 Thesis Outline and Contributions

The character recognition problem as formulated earlier in the thesis, is viewed as a hierarchical problem in two levels of abstraction. The first level is classification of observed data into characters. The second level is the contextual analysis of consecutive characters to correct for errors and determine the true words.

We believe that the second level problem has been handled somewhat better in the literature than the first level problem. The difficulty in the character recognition problem arises – in our view – from two fundamental issues:

1. How should one model the uncertainties in the channel (the hand/printer)?

2. How should one measure similarity and dissimilarity between characters?

It is these two issues that we discuss in depth in the thesis. In the process we conclude that the standard paradigm of pattern recognition viz. feature extraction and subsequent partitioning of the space in which the characters live (classification) is not the correct one for this problem. For example, the segmentation problem may well be ill-posed. We develop a new paradigm in the form of certain principles (laws) which should govern the design of pattern recognition problems and which appear to have certain affinities to ways human beings recognize patterns. Our view is an internalist one (in the Chomskean sense) where primary emphasis is given to the representation of ideal characters (templates) and their deformations, representing uncertainties in the channel, which are in some sense universal. We try to understand the data on the basis of our knowledge (templates) and their deformations, which may be data dependent. Indeed, we attempt to avoid any processing of the observed data other than filtering the amplitude noise. We expect that pattern recognition systems built according to the principles enunciated in this thesis would require much less training than existing systems.

The thesis has five chapters. We have chosen to present the thesis as it developed historically during the past three years. Chapter 2 is a literature *understanding* in the light of information theory, while pointing out some of the weaknesses in the

field. We explore the metric idea (in the mathematical sense), the backbone of most similarity measures, in Chapter 3 for the case of time signals when both domain and amplitude deformations are present. We discuss the problem of detection for these models and present a theorem on the appropriate choice of a metric on the signal space. In subsequent chapters, we – in effect – abandon the *metric* philosophy, because one of the important contributions of this thesis is that a measure of similarity does not necessarily correspond to a metric. Indeed, there is essential asymmetry in the recognition process which needs to be captured in the similarity measure. This asymmetry is probably closely tied to the internalist view. Namely, one should work with templates and their deformations to look for patterns in the data rather than the other way around. We view recognition as a process of searching for patterns rather than complete matching between the observed data (samples) and the prior knowledge (templates). Nevertheless, the developments in Chapter 3 were instrumental in directing us to the view of pattern recognition as developed in Chapter 4, and that is why we have chosen to present the material in its historical order.

In Chapter 5 we describe the tools used in the recognition algorithm and report the results of implementation. Finally, in the appendix we describe a contextual analysis algorithm for detecting addresses (in the classical framework) from individual characters available as output of the classifier, by correcting replacement errors as well as erasures and insertions.

Chapter 2

Literature Review

2.1 Introduction

The framework of information theory suggests two levels of abstractions, with possible feedback between them. At the lower level, each character is identified individually. At the higher level, contextual information (in the form of acceptable groups of characters) is used to correct the errors. The advantage of the divide-and-conquer approach in managing the complexity has been recognized in several fields, including data networks and operating systems. The field of pattern recognition is no exception. More importantly, as we pointed out earlier, the recognition process is itself an abstraction at a suitable level. The following sections elaborate on this hierarchy of abstractions in reading text.

2.2 Contextual Analysis

In advanced optical text-recognition systems, the performance of a recognition system which consists only of a single-character recognition unit is not sufficient. Linguistic, contextual, or statistical information can be used to resolve ambiguities of characters – which have similar shapes – to detect errors or even correct them[24].

To focus on the source decoder or the contextual analysis module, all the blocks between the source encoder and the source decoder are lumped into one discrete

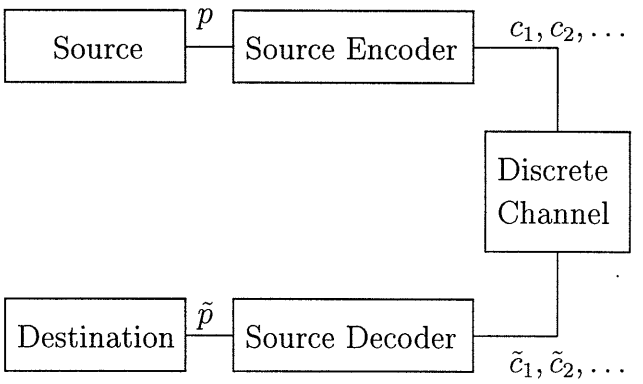


Figure 2-1: Block diagram of a system for contextual analysis

channel (see Figure 2-1). The source encoder maps each concept to a variable-length string of characters $c_1, c_2, \ldots$, which then forms an input to the channel. The source decoder collects the channel output $\tilde{c}_1, \tilde{c}_2, \ldots$ and guesses the transmitted concept.

2.3 Source Encoder Models

A concept p_i is encoded as a string of characters $c_1^i, c_2^i, \ldots, c_{L(i)}^i$, of variable length ($L$ is a function of i). The characters are elements of the union set of the english alphabet, the numerals, hyphenation marks and the space mark. For convenience, we add an End-Of-String (EOS) marker and denote the size of the resulting set by K.

Assume the concepts to be statistically independent. The source can be merged with the source encoder to obtain a new module. This module would be a random source generating statistically independent strings from a dictionary. Note that although the consecutive strings might be modeled statistically independent, the consecutive characters cannot. Such being the case with the new source, it is not reasonable to model it as a memoryless source. A more appropriate model would be a Markov process.

The great advantage of using a Markov model for the source is that the amount of computation needed in the postprocessing phase is *independent* of the size of the dictionary. On the other hand, the reported correction accuracy is only around 50 percent[35]. This accuracy can be improved by increasing the order of the model.

However, the number of states needed grows *exponentially* with the order of the model.

In the light of the above discussion, it was well understood by researchers[32] that to improve the correction accuracy, one had to use the entire input dictionary and not merely its statistical model. In other words, one had to represent the source as a stochastic tree.

2.3.1 Other Source Encoder Models

The attentive reader would note that, along the trip from the discrete memoryless model to the dictionary Markov model, error performance was traded for storage requirements. While using a large dictionary of concepts "as is" resulted in the best error performance, the required storage was excessive, hence the need for reducing *complexity*. While a Markovian representation is the mostly used compression technique in the field of text recognition, other techniques also exist, and they all come under the title of *Hashing* techniques[1].

Before proceeding, it is worth mentioning that what is known in the literature as a *binary n-gram* representation is only a simplified Markov model, where all permissible transitions are assumed equiprobable. The real advantage of this representation is that it is extremely compact, and can be handled efficiently. On the other hand, what is known as a *positional binary n-gram* representation is a simplified dictionary model, where all permissible transitions are assumed equiprobable. It is more efficient to handle than the dictionary model, at the expense of higher error rates (and not larger storage as claimed in Riseman[40]), unless non-consecutive indices are being handled as well. Figure 2-2 displays the various speed-storage-accuracy tradeoffs between the different source encoder models.

[1]Note that hashing is actually a form of compression since it represents the dictionary using a smaller table, the hash table.

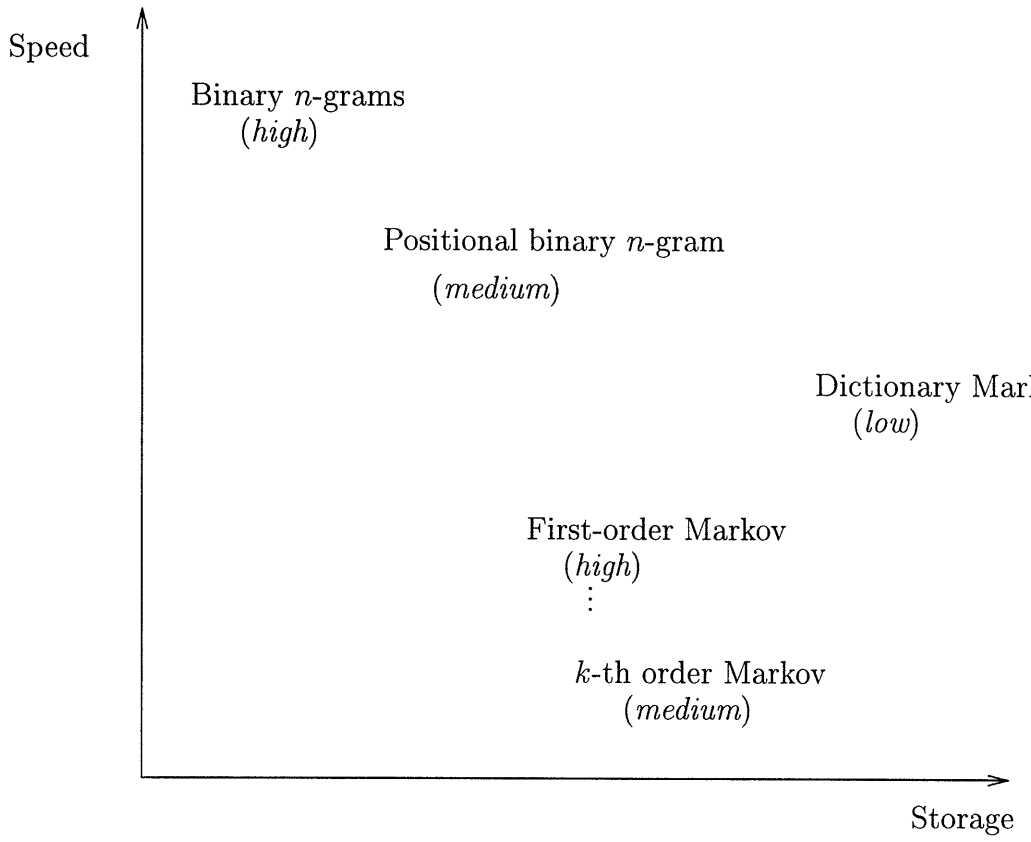


Figure 2-2: Speed-storage-accuracy tradeoffs between the different source encoder models

2.4 Channel Models

2.4.1 Hand/Printer = Channel

The media (hand/printer) involved in the transmission of characters are not perfect. Even in the case of the best artist or the best printer, inaccuracies are incurred. What is obtained on paper (or on a digitizer) is not the character, but rather a *deformed* version of it. The uncertainty of deformation results in some detection errors ($\tilde{c} \neq c$). Therefore, it is reasonable to model the (hand/printer+character recognition algorithm) as a discrete channel in the *information-theoretic* sense.

In what follows, we follow the classical path of thought, and we assume that the channel input is the string $c_1, c_2, \ldots$, and the channel output is the string $\tilde{c}_1, \tilde{c}_2, \ldots$

2.4.2 DMC Channel

The simplest channel model, at the level of contextual analaysis, is the discrete memoryless channel. A DMC channel is characterized by the transition probabilities $P(c_i/c_j), \forall i, j$. The main reason for using a DMC channel model is simplicity.

2.4.3 DSC Channel

If the writing is constrained, e.g., all the characters we are attempting to read are written inside boxes, the channel can be modeled as a discrete *synchronous* channel (DSC). In that case, $\tilde{c}_1$ corresponds to c_1, $\tilde{c}_2$ corresponds to c_2, and so on. Such a situation occurs partially in the french mail, where envelopes have five boxes in which to write the ZIP Code. It also occurs in reading most questionnaires and data entry forms.

Synchronous channels are completely described by specifying the set of possible inputs to the channel, the set of possible outputs and, for each input and output history, a probability measure on the set of outputs. Using our notation, a DSC channel is characterized by

$$P(\tilde{\boldsymbol{c}}/\boldsymbol{c}) = \Pi_{i=1}^{N} P(\tilde{c}_i/c_1, c_2, \cdots, c_i, \tilde{c}_1, \tilde{c}_2, \cdots, \tilde{c}_{i-1})$$

2.4.4 EIC Channel

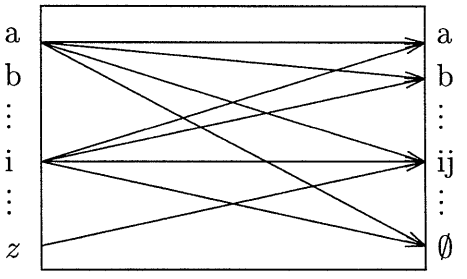


Figure 2-3: Erasure-insertion channel

On the other hand, US addresses are handwritten without constraints. Similarly, banks refuse to put any constraints on where and how to write the check amount.

This results in segmentation problems in both of the above applications. In that case, asynchronism might occur, and the channel may add or delete characters.[2] A standard example is the misinterpretation of the touching pair "rn" as "m", and vice versa. To account for this behavior, we have to alter the classical channel model as follows: We add to the output alphabet a special symbol $\emptyset$ called the *empty symbol*, together with all the possible pairs of characters. The probability of incorrectly deleting a character c is represented as $P(\emptyset/c)$, and the probability of incorrectly reading a character k as the pair ij is represented as $P(ij/k)$. The resulting erasure-insertion channel (EIC) is shown in Figure 2-3. A major difference between the erasure in this case and the erasure in classical binary-erasure channel (BEC) is that, in the EIC case, the channel output gives no indication as to which letters were deleted or inserted.

Note that our version of an EIC channel (first presented by Gallager[12]) is better than the one provided in Kashyap[25]. In his model, Kashyap uses an empty symbol at the input of the channel to describe insertions. This approach, however, causes difficulty in estimating the insertion probabilities. Furthermore, it does not map naturally the cases encountered in practice, such as recognizing "m" as "rn" (printed), or "m" as "nn" (handwritten), or "w" as "vv", etc. Each such case will be counted in his model as two errors, whereas we count it only as one.

2.4.5 Conclusion

At the level of concept recognition, richer channel models arise. Examples include: discrete memoryless channels suitable for simplicity; discrete synchronous channels suitable for machine print, questionnaires, and data forms; and finally, erasure-insertion channels suitable for cursive writing and touching characters.

[2] A third kind of error, the transpose of characters, commonly encountered in typing, is not accounted for in this model.

2.5 Source Decoder/Postprocessor

In what follows, we discuss the classical decoding strategies for various choices of source encoders and channels. We consider dictionary sources, i.e., those represented as trees, and Markov sources, i.e., those represented as trellises. Together with these sources we consider DMC channels and EIC channels.

2.5.1 Dictionary Source and DMC channel: Top-Down Decoding

An example of such a case is reading a string whose characters are printed inside boxes. The observed string may or may not be in the dictionary. In the latter case, we want to find the closest string in the dictionary according to some prechosen measure. Consider the tree representation of the dictionary. We can imagine laying the given string along every possible path through the tree to find the best agreement, as follows:

1. Let the observed string be $s_o = c_1 c_2 \ldots c_L$.

2. Consider only the subdictionary of strings having same length as the observed one. Let M be the number of such strings.

3. For every string $s_d = l_1 l_2 \ldots l_L$ in the subdictionary, calculate the distance from s_o. The distance is defined as:

$$d(s_o, s_d) = -\log(P(s_o/s_d) = -\sum_{i=1}^{L} \log(P(c_i/l_i))$$

4. Select the string with the smallest distance.

Traditionally, any function that satisfies the *distance* properties could be used as a candidate for d. The above function, however, results in a maximum likelihood decoding rule.

The complexity of the algorithm is $O(ML)$. It is possible that M, the dictionary size,

be exponential in L (i.e., $M = O((\alpha K)^L)$, where K is the size of the alphabet). In this case, sequential algorithms, which are fast algorithms to find the minimum cost path in a tree, become a better alternative.

Sequential algorithms are strategies for searching only the likely paths through a tree (or a trellis). They usually leave pending most decisions to drop paths permanently. Occasionally, a sequential algorithm will decide to back up and extend a path previously ignored. There are two classes of sequential algorithms in use, the stack algorithm and the Fano algorithm. We will not describe any of them in this section, partly because we will discuss the Fano algorithm later, and also because there is enough literature about them, especially in the theory of convolutional coding. The interested reader should consult Gallager[13] for a rigorous analysis, or Blahut[2] for a readable introduction.

Finally, if we assume that all transitions are equiprobable (i.e., a positional binary n-gram representation), then an efficient decoding scheme as explained by Hull[21] can be used. To determine the error location we check which positional n-grams rejected the string, since they must contain the location(s) of the error(s) among their position indexes. To correct the error is simple if there is only one index. If there is more than one index, we assume a single error and check for consistency in the table. The procedure can be generalized for more than single errors.

2.5.2 Markov Source and DMC channel: Bottom-Up Decoding

If the source is represented as a Markov model and the channel as a DMC, the observed string hides some Markov properties in it, hence the term *Hidden Markov model.*

A kind of graph called a *trellis* can usefully describe the output sequence of any finite-state machine. In general, a trellis is a graph whose nodes are in a rectangular grid, semi-infinite to the right; the number of nodes in each colomn is finite. The configuration of the branches connecting each column of nodes to the next column of

nodes on the right is the same for each column of nodes. Figure 2-4 shows a Trellis corresponding to a 3 state Markov process.

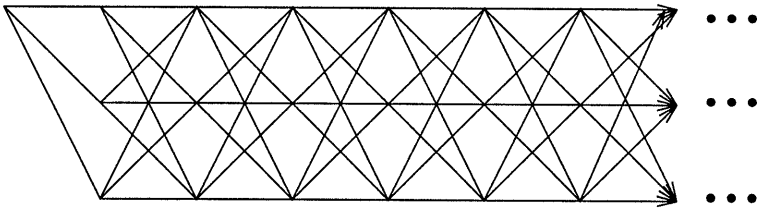


Figure 2-4: A trellis that corresponds to a 3 state Markov process

We can imagine simply laying the observed word along every possible path through the trellis. This procedure has complexity exponential in L, the length of the word. In particular, if K is the number of states in the Markov model, the procedure requires $O(K^L)$ computations. The Viterbi algorithm, which we describe next, will efficiently search the trellis in $O(K^2L)$.

The Viterbi algorithm, proposed in 1967, uses forward dynamic programming to find a running set of candidates for the best path through a trellis[2]. In particular, it can be viewed as a solution to the problem of maximum *a posteriori* probability (MAP) estimation of the state sequence of a finite-state discrete-time Markov process observed in memoryless noise[9]. A tutorial description of the Viterbi algorithm can be found in Forney[9]. According to Blahut[2], it may be helpful to think of the Viterbi algorithm as a window through which a portion of the trellis may be viewed. One can see only a finite-length section of the trellis, and on this is marked the surviving paths, each labeled with a cost. As time goes by, the trellis slides to the left. As new nodes appear on the right, some paths are extended to them, other paths disappear, and an old column of nodes is lost on the left side.

The Viterbi algorithm has become the *de facto* decoding strategy in the case of Markov source and DMC channel. Neuhoff[35] investigated its ability and commented favorably on its simplicity and accuracy. While the optimal algorithm is exponential, suboptimal versions of it are reasonably fast. One suboptimal version is the beam search Viterbi algorithm (Shinghal and Toussaint[44]), in which d paths – at most – are kept at every stage.

2.5.3 Summary for DMC

The attentive reader would have noted that a tree can be viewed as a trellis. So, when should we use the Viterbi algorithm instead of the Fano algorithm? The answer depends on K, the size of the alphabet, L, the length of the word, and M, the size of the dictionary. Viterbi requires, roughly, $2LK^2$ additions and LK^2 comparisons[35], irrespective of the noise introduced by the channel. On the other hand, the Fano algorithm is very fast when there are no errors, and is slow when there are numerous errors. Furthermore, its optimality is asymptotic, i.e., when long sequences are involved. These characteristics suggest that the Viterbi algorithm should be used with dictionaries consisting of short words, or with Markov sources - with limited number of states - whose output is transmitted through a high noise channel. On the other hand, the Fano algorithm should be used with a dictionary consisting of long strings (e.g., the dictionary of US addresses), or with Markov sources transmitting through a near noiseless channel.

2.5.4 Dictionary Source and EIC Channel

The Levenshtein distance (LD) between two strings is defined as the minimum number of edit operations on the individual symbols needed to convert one string into the other. The edit operations are insertion, deletion and substitution[25]. By using different weights for the three different edit operations, we get the weighted Levenshtein distance. Okuda (1976) reports in [36] several experiments using Levenshtein distance, and Masek (1980) presents a fast algorithm for computing the distance. The Levenshtein distance is used as a criteria for decoding when we have a dictionary source transmitting through an EIC channel, a situation that ideally models mail interpretation.

The Levenshtein distance requires a dynamic programming type of algorithm to minimize. This means that even in the case of no errors, lots of computations are to be made. We ask: is there a Fano decoding type of algorithm which decodes quickly when there is no errors but takes longer when errors are present? Indeed,

Appendix A1 describes a modification of Fano to accomadate this situation. We see here another advantage of using the information-theoretic framework. We were able to quickly spot areas in the field that haven't been completely studied.

2.6 Summary

Table 2.1 summarizes the natural decoding mechanism for several encoder-channel pairs.

Channel/Encoder	Dictionary(Tree)	k-Markov(Trellis)
DMC	Fano	Viterbi
EIC	M. Fano, LD	Uninvestigated

Table 2.1: The decoder choice for several encoder-channel pairs

2.7 Feature Selection and Classification

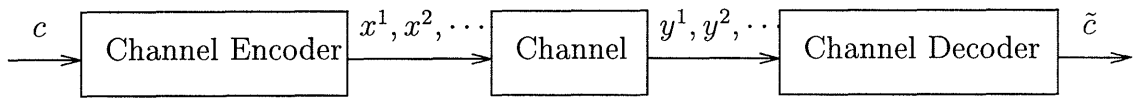


Figure 2-5: Block diagram of a system for feature selection and classification

To understand how -classically- the features were selected, it is better to lump together all the blocks that are between the channel encoder and the channel decoder into one discrete channel (see Figure 2-5). The channel encoder maps each character to a collection of features $x^1, x^2, \ldots$, which is then input to the channel. The source decoder collects the channel output $y^1, y^2, \ldots$, and guesses the transmitted character. The major problem here is synchronization or segmentation. Later in the thesis, we discuss better ways of dealing with the segmentation problem.

2.8 Channel Encoder Models

The characters of any alphabet contain attributes (features) that are resistant to channel noise (i.e., hand or printer inaccuracies). Researchers thought of the text generation process as sending these particular features *only* over the channel. They argued that this is a sensible approach since these features are the only information used by the decoder (i.e., the recognition algorithm).

Later in the thesis, we will adopt the view that features are the aspects of the character that are hit by the noise. Since the noise is mostly domain noise, we will take for features the (x, y) positions of the black pixels.

2.8.1 Features Definition

But, what is a feature formally? In the literature, a feature is defined to be a mapping f^i, from $\mathcal{S}$, the set of planar patterns, to R, the set of real numbers. Mathematically,

Definition 2.1 *Let D be a bounded subset of the plane. Let S be a surface defined over D such that the height of S over D is strictly positive. Then, S is a geometric planar pattern whose domain is D. Let RD be the smallest rectangle (with sides parallel to the axis) containing D. Then, RD is the rectangular domain of S.*

Definition 2.2 *Let $\mathcal{S}$ be the set of geometric planar patterns. Furthermore, let f^i be a real mapping defined over $\mathcal{S}$, i.e.,*

$$\begin{aligned} f^i : \mathcal{S} &\longrightarrow R \\ c &\longmapsto f^i(c) \end{aligned}$$

Then, f^i is a feature.

Note that if the functional f^i is a feature, then the value of that functional at c, $x^i = f^i(c)$, is sent across the channel. Given N features, c is *encoded* as a feature-word of length N, $x^1, x^2, \cdots, x^N$ (denoted $\boldsymbol{x}$), and transmitted over the channel. This block encoding technique is the one used in the statistical recognition school. It usually

assumes that there is no inherent relation among the features. This assumption is valid if the features are conditionally independent, which is almost never the case. To exploit the dependence among the features, relations of various sorts, such as Markovian property, were assumed by the linguistic recognition school,

2.8.2 Historical Survey

Despite the historically huge list of proposals for a suitable feature set, i.e., channel encoding, it is possible to group the proposed feature sets into two major categories: global and local. A feature is called *global* if it is extracted from every point of the planar geometric shape. Examples of global features are moments, transforms, number of cusps, loops, bays, etc. A feature is called *local* if it is defined in terms of local elementary geometrical or topological shapes. Examples of local features are morphological features, strokes, T-junctions, X-junctions, etc., and their positions in the frame bounding the character[24].

2.8.3 Desired Properties

What are the properties of a good feature set? Researchers have sought two main properties: discrimination and invariance. *Discrimination* indicates that, in the feature space, samples of *different* characters are far from each other. *Invariance* indicates that, in the feature space, samples of the *same* character are close to each other.

2.9 Channel Models

The channel at this level encompasses three modules: the ideal image features generation module, the hand/printer, and the feature extraction module. While in the contextual analysis layer, the channel models were exclusively discrete in amplitude (e.g., DMC, EIC, DSC), this restriction does not apply to the classification layer. Additional important channel models that arise at this stage are additive white Gaussian

noise channels (for primitive analysis), additive white noise channels (for simplified analysis), and fading channels (for hand fatigue). Note that these models were not used for the hand/printer. They were used (or at least some of them) for the combined feature generation-hand/printer-feature extraction channel. We are not aware of any models of the hand/printer that were incorporated in a recognition system.

2.9.1 The Hand/Printer

Later in the thesis, we envision the hand/printer as incurring three types of deformations: Global, local and point deformation. Global deformations include: slanting, translation, scaling, etc. with the deformation parameter confined to a certain range. Global deformations can be described as deformations preserving the topology of the character. Local deformations include: displacement of a stroke in the character A, scaling of a stroke in the character 4, etc. Local deformations are global deformations operating only on predefined subsets of the character called primitives. Point deformations are slight disturbances of te individual points. Point deformations makes the difference between a line drawn by hand and a line drawn by a ruler.

2.9.2 Feature Extraction

To extract the features, one necessary prerequisite is to estimate the boundaries of the character. This is a direct consequence of the requirement that a metric should be used to measure the similarity between the template and the sample. This classical view of similarity forces us to treat the template and the sample equally. It does not handle in any structured manner the problem of blobs, scratches, or touching characters. In the case of neat handprinting, this problem does not pop up. On the other hand, if the characters are touching or have scratches or blobs, isolating them from the noisy surroundings becomes a challenging task.

The above argument leads us later in the thesis to simply abandon the metric approach, and to search in the noisy observation for the template we propose. In effect, we want to put more confidence in our knowledge (template processing) rather

than dividing this confidence equally between our knowledge and the observed data (samples) as suggested by the metric approach.

2.10 Channel Decoder/Classifier

Back to the classical view of character recognition. A Bayesian, or maximum a posteriori rule (MAP) is the optimum rule for classification assuming a certain stochastic model. Given an observed feature-word $\boldsymbol{y}$, we choose the character c_j that maximizes $P(\boldsymbol{x}_j/\boldsymbol{y})$, where $\boldsymbol{x}_j$ is the feature vector corresponding to c_j. If the characters are assumed equiprobable, MAP reduces to the maximum likelihood rule: choose c_j that maximizes $P(\boldsymbol{y}/\boldsymbol{x}_j)$.

Unfortunately, it is simply intractable to estimate the joint pdf $P(\boldsymbol{y}/\boldsymbol{x}_j)$. $O(K^N)$ samples are needed to estimate the joint histogram in N dimensions when using K bins per dimension. There are three ways to get out of this problem: either (1) assume independence of features, which means $P(\boldsymbol{y}/\boldsymbol{x}_j) = \Pi_{i=1}^{N} P(y^i/x_j^i)$ or, (2) assume the features are Gaussian, in which case there are efficient means of estimating the required parameters of the joint Gaussian pdf or, (3) settle for less and use suboptimal procedures that make use of the marginal pdfs (which can be easily estimated)[3]. Commonly used classifiers in the literature are: K-nearest mean, K-nearest neighbor[11], Fisher pairwise, and neural networks[20].

A major step in information theory was to realize that by merging the digital data demodulator and the channel decoder in one module, better performance can be achieved. Later, we resort to the same technique and we attempt to recognize the character directly from the (x, y) positions of the observed black pixels. Instead of detecting primitives and then detecting characters, we do the opposite. We deform the character template globally until maximum matching is achieved with a subset of the data, then we deform the template locally until better matching is attained. Finally, we deform the points to get exact matching. All the above deformations have to be within the permissible range, otherwise we consider the template not to exist

[3]The techniques in (3) are the same as in (1). Only the point of view is different.

in the observed data.

2.11 Summary and Conclusions

Handwritten character recognition is an old field of research, where various approaches have resulted in a vast literature. Yet, no solid line of research has emerged. It seems that a framework is needed. Under this framework, the weak points can be highlighted, the important questions can be formulated, and the achievable limits can be found. In the previous chapters, we have attempted to develop such a framework using information theory as a backbone. We pointed at certain flaws governing the classical approach, such as the "serialzation" of an inherently geometric problem, and the insistance to use a metric as a measure of similarity. In the next chapter, we will explore the traditional viewpoint of using a metric to see how much we can push it. We will consider the simple case of one dimensional signals corrupted with both amplitude *and* domain noise, and we will state a theorem on an appropriate metric to deal with this situation.

Chapter 3

Simplification: The Case in One Dimension

3.1 Introduction

In what follows, we will be discussing the problem of recognizing single dimensional, real-valued, functions. While this might seem, at first glance , a distraction from the course of the thesis, it is actually very instrumental in pointing out the kind of difficulties that one faces when dealing with both domain and amplitude deformation. We will assume that the functions under consideration are of bounded support (actually defined on the unit interval), and we will work out the problem of determining the proper metric.

To proceed, let H be the space of functions of bounded variations, defined over the unit interval I. Let $h_1, h_2, \ldots, h_M$ be elements of H called *hypotheses*. Let $g \in H$ be the *received* signal. Determine a plausible detection rule to determine the most likely h_i that was transmitted.

The traditional approach, in signal transmission theory, is to assume that g was obtained from one of the h_i's through amplitude deformation (additive or multiplicative). In other words, we make M hypothesis $H_1, H_2, \ldots, H_M$, where under hypothesis

H_i (assuming additive noise):

$$g(t) = h_i(t) + n(t) \quad 0 \leq t \leq 1$$

and $n(t)$ denotes the amplitude noise, lying in some suitable space. Next, we assume as cost a functional of $n(t)$. Finally, we select the hypothesis that minimizes the cost.

If the L_2 metric was used as a cost on H, for instance, the detection rule would be to choose h_i that minimizes $\int (g(t) - h_i(t))^2 dt$. To use a probabilistic interpretation, this is the rule that corresponds to maximum likelihood detection when $n(t)$ is modeled as Gaussian white noise.

Unfortunately, comparing signals based on their amplitude difference only does not satisfy our intuition. Consider, for instance, the following example:

$$\begin{aligned} h_1(t) &= 0.125 \quad \text{for} \quad 0.1 \leq t \leq 0.99 \text{ and 0 otherwise} \\ h_2(t) &= 1 \quad \text{for} \quad 0.5 \leq t \leq 0.51 \text{ and 0 otherwise} \\ g(t) &= 1 \quad \text{for} \quad 0.52 \leq t \leq 0.53 \text{ and 0 otherwise} \end{aligned}$$

In this example, g is closer to h_1 in the L^2-sense, whereas it is closer to h_2 in any intuitive sense (see Figure 3-1).

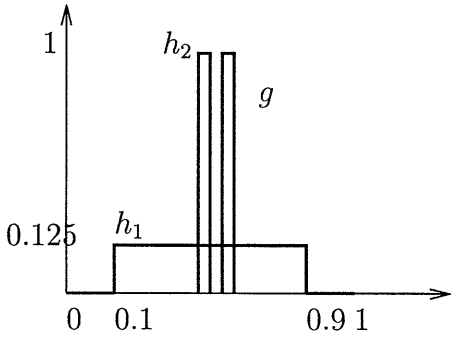


Figure 3-1: Counter example showing weakness of L_2 metric

Where is the catch? L^2 metric assigns a large cost for domain deformation. So, it seems that we should formulate the problem in a way that accounts for domain deformation in addition to amplitude deformation.

Let us concentrate for the moment on domain deformation only. Later, we see how

both deformations can be brought together. A naive attempt to solve the problem is to say: under hypothesis H_i we have

$$g(t) = h_i(t + w(t)) \tag{3.1}$$

and then use the cost function $\int w(t)^2 dt$ as a criteria for similarity. However, this function has annoying asymmetry. The reason is, if we write $g(t) = h_i(t + w_i(t))$, and alternatively, $h_i(t) = g(t + w_i'(t))$, then it is not true that $\int w_i(t)^2 dt = \int w_i'(t)^2 dt$. Furthermore, if we decide later to shift our interest to functions having infinite support, it is possible for a small domain shift to cause a blow-up in the cost function (consider $\cos(t)$ and $\cos(t + \epsilon)$), which means it is not translation invariant. A more convenient relation than Equation 3.1 is

$$g(t) = h_i(x(t)) \quad \text{or} \quad h_i(t) = g(x^{-1}(t)) \tag{3.2}$$

assuming x is invertible. Using Equation 3.2 we can immediately see that for a cost functional to be symmetric, it has to remain the same if x is replaced by x^{-1}. These ideas will be made more concrete in the following section.

3.2 Domain Deformation : Theory

Assume that under hypothesis H_i:

$$g(t) = h_i(x(t)) = (h_i \circ x)(t)$$

where x(t) is an order-preserving homeomorphism of the unit interval I onto itself. We will show later that a solution x for the above equation exists if and only if g and h_i have the same sequence of extrema.

The reader should note that, throughout the coming discussion, we will be dealing with three different spaces: X, H, and W. First, we will define X, and then we will present a few lemmas to gain some understanding about the space X.

Let X be the space of all order-preserving homeomorphisms of the unit interval I onto itself. Then, the following two lemmas are well known.

Lemma 3.1 *A function x is an element of X if and only if it can be represented as a continuous, strictly increasing, function joining the origin to the point (1,1) of $I \times I$ (see Figure 3-2). The horizontal axis in Figure 3-2 (also called the t-axis) is the domain of g, and the vertical axis (also called the x-axis) is the domain of h_i.*

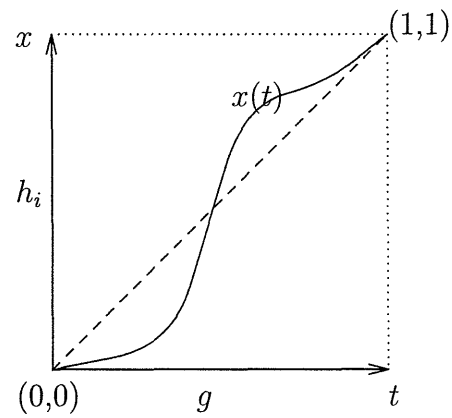


Figure 3-2: Domain deformation shown as a curve in the product domain of g and h_i

Note that the functions g and h_i are to be visualized normal to the plane of Figure 3-2.

Lemma 3.2 *The pair $(X, \circ)$ is a group.*

Note that the inverse x^{-1} is a reflection of x around the diagonal of $I \times I$.

Lemma 3.3 *The space X, viewed as a set, is convex.*

Proof: By convex we mean, if x_1 and x_2 belong to X, then so do $\alpha x_1 + (1-\alpha)x_2$ for all α in $[0,1]$. Let $x = \alpha x_1 + (1-\alpha)x_2$. Then x is continuous, being a weighted sum of continuous functions. Letting $t_1 < t_2$, we get

$$\begin{aligned} x(t_2) &= \alpha x_1(t_2) + (1-\alpha)x_2(t_2) \\ &> \alpha x_1(t_1) + (1-\alpha)x_2(t_1) \\ &= x(t_1) \end{aligned}$$

Finally, $x(0) = 0$, and $x(1) = 1$. Hence, x is in X.

Next, we define the second space of interest, H, and its subspace, H_g:

Definition 3.4 *Let H be the space of real functions of bounded variations defined over I. Let g be an element of H. H_g is defined to be the set of all functions in H which can be obtained from g through order-preserving, homeomorphic, domain deformation. In other words,*

$$H_g = \{f : g = f \circ x, x \in X\}$$

Note that we already have an onto mapping from X to H_g, mapping x to $g \circ x^{-1}$. However, this mapping is not one-to-one in general. Consider, for instance, the constant function $g(t) = 1$. Then, H_g is the singleton $\{g\}$, since $1 \circ x^{-1} = 1$ for all x in X. By removing from X the "redundant" x's, the onto mapping becomes a bijection. For the case $g = 1$, the redundant deformations are all the deformations except $x = i$. The following lemmas characterize the redundant domain deformations for a general function g.

Lemma 3.5 *Let f and g be both strictly increasing (decreasing) functions. Assume that the relation $g = f \circ x$ has a solution in X. Then, that solution is unique.*

Proof: Assume the contrary. Let x_1 and x_2 be two different solutions of the equation $g = f \circ x$. Then, there exists a point a at which they differ. Let $x_1(a) = b_1, x_2(a) = b_2$. However, $g(a) = f[x_1(a)] = f[x_2(a)]$. Therefore, $g(b_1) = g(b_2)$ for $b_1 \neq b_2$. This is a contradiction, since g is strictly monotonic. Hence, the solution x is unique.

Lemma 3.6 *Let $f, g \in H$. Let $g = f \circ x$ has a solution in X. Then, x is a bijection between the local maxima(minima) of g and the local maxima(minima) of f.*

Proof: Let a be a point at which g has a local maximum. Then, there is an open nbhd of a, $B(a) \subset I$, such that $g(t) \leq g(a)$ for all t in $B(a)$[1] Consider $x[B(a)]$. It is

[1]To handle the extreme points 0 and 1, view I as a subspace of R, where the topology is inherited by taking the intersections of open sets of R with I.

open since x is a homeomorphism, and contains $x(a)$. For all u in $x[B(a)]$, we have $f(u) \leq f[x(a)]$, otherwise $x^{-1}(u)$ would be a point in $B(a)$ such that $g[x^{-1}(u)] \geq g(a)$. Hence, $x(a)$ is a local maximum of f. Similarly, if b is a local maximum of f , we can prove that $x^{-1}(b)$ is a local maximum of g. Finally, the minima can be treated in an analogous way.

Theorem 3.7 *Suppose that g is not constant on any subinterval of I, and that the relation $g = f \circ x$ has a solution in X. Then, the solution is unique on the whole interval.*

Proof: Let $a_1 < a_2 < \ldots$ be the points at which g has extrema. Let $b_1 < b_2 < \ldots$ be the points at which f has extrema. Let $g = f \circ x$ has a solution. Then x in Figure 3-2 should pass through the points $(0,0), (a_1, b_1), (a_2, b_2), \ldots, (1,1)$, otherwise x would not be increasing. Furthermore, between any two consecutive points $(a_i, bi), (a_{i+1}, b_{i+1})$, f and g are either both increasing or decreasing. As a consequence of Lemma 3.5, the solution is unique in $[a_i, a_{i+1}]$. Therefore, the solution is unique on the whole interval.

Corollary 3.8 *Suppose that g is not constant on any subinterval of I. Then there exists a bijection between H_g and X, mapping f in H_g to the unique x in X, satisfying $g = f \circ x$.*

Corollary 3.9 *Let g be constant on the ordered intervals $(c_1, c_2), (c_3, c_4), \ldots$ Let f be constant on the ordered intervals $(d_1, d_2), (d_3, d_4), \ldots$ Then,*

1. *Any valid solution of $f = g \circ x$ should map (c_i, c_{i+1}) to (d_i, d_{i+1}) for all i.*

2. *Outside these subintervals, x is uniquely determined.*

Throughout the work, we will choose the solution x that maps (c_i, c_{i+1}) to (d_i, d_{i+1}) *linearly.* As a consequence of this, the solution to $g = f \circ x$ becomes unique. We denote by X_g, the subset of X obtained after deleting every x which is not linear in the constant subintervals of g.

Our goal has been to define a distinction function d^* over $H_g \times H_g$, which handle

domain deformation in a way analogous to the way L_2 or other metrics handle amplitude deformation. The direct consequence is that d^* becomes a metric over H. The second consequence is that $d^*(f, f \circ x)$ becomes a function of x only.

The second consequence implies that $d^*(f_1, f_2)$ should not change if we deform both domains by the same deformation (invariance to composition). The equivalent assumption in the classical case is to say the amplitude noise cost is dependent on the difference $g(t) - h_i(t)$ only (invariance to translation).
Let us denote $d^*(f, f \circ x)$ by $< x >$. Then, a more compact way to write the properties of a distinction function is:

1. $x \in X$ is the identity function if and only if $< x >= 0$.

2. If $x \in X$, then $< x >=< x^{-1} >$.

3. If x_1 and x_2 are in X, then $< x_1 > + < x_2 > \geq < x_1 \circ x_2 >$.

Clearly, if $< x >= d^*(f, f \circ x)$ is a distinction function, then so is $\lambda < x >$ for any positive constant λ.
Note that the distinction function d^* defined on H_g induces a metric on X_g. In other words, if x_1 and x_2 are in X_g, then $d_{X_g}(x_1, x_2) =< x_1 \circ x_2^{-1} >$ is a metric on X_g. Furthermore, the two metric spaces H_g and X_g become very similar in the sense of the following theorem:

Theorem 3.10 *Let d^* be a function over $H_g \times H_g$. let d be a function over $X_g \times X_g$ defined as follows:*

$$d(x_1, x_2) = d^*(f_1, f_2) \quad \textit{such that} \quad g = f_1 \circ x_1 = f_2 \circ x_2$$

Then, d^* is a distinction function if and only if d is a composition invariant metric. Furthermore, if d is a composition invariant metric on X, the mapping

$$\begin{aligned} b : (H_g, d^*) &\longrightarrow (X, d) \\ f &\longrightarrow x \in X_g : g = f \circ x \end{aligned}$$

is an isometric imbedding of H_g in X.

Proof: The mapping b is well defined since x is uniquely determined (see Corollary 3.9 and the following remark). Furthermore,

$$d^*_{H_g}(f_1, f_2) = d_X(b(f_1), b(f_2))$$

Hence, b is an isometric imbedding of H_g in X, or an isometry between H_g and X_g. Since b is a bijection between H_g and X_g, then d is a metric if and only if d^* is a metric. Furthermore, let $g = f_1 \circ x_1 = f_2 \circ x_2$, then $d^*(f_1, f_2) = d^*(g, g \circ x_2^{-1} \circ x_1)$ if and only if $d(x_1, x_2) = d(i, x_2^{-1} \circ x_1)$.
Isometry has important consequences since isometric spaces are identical in all respects except for the nature of their elements, which is inessential[26].

The question is: "Does there exist a valid distinction function?". Happily, the answer is "Yes!". Later in the discussion we will present several examples. For now, the reader can verify that the following is one such function

$$d^*(f, f \circ x) = \sup_{t \in I} |x(t) - t|$$

Before proceeding, it is time to define the third space of interest, W.

Definition 3.11 *Let X be the space of order-preserving homeomorphisms defined earlier. Define W to be the noise part of the domain deformation. In other words:*

$$W = \{w(t) : w(t) = x(t) - t, x \in X\}$$

The bijection between W and X is immediate, and the convexity of $(X, +)$ implies the convexity of $(W, +)$. However, $(W, \circ)$ is not a group any more. Also, $(W, +)$ is not a linear space, so we cannot define a norm over it. However, if we extend W by including all continuous functions w satisfying $w(0) = w(1) = 0$, then the extended space becomes a linear space. Any norm on the extended space may be inherited by W. Alternatively, any metric on the extended space can also be used to metrize W.

Theorem 3.12 *Let d' be a metric over W, induced by defining a norm over the extended W. Let d be a function over $X \times X$ defined as follows:*

$$d(x_1, x_2) = d'(x_1 - i, x_2 - i)$$

Let x_1 and x_2 be two points in X. Then, the following mapping

$$\begin{aligned} \Gamma : [0,1] &\longrightarrow X \\ \alpha &\longmapsto \alpha x_1 + (1-\alpha)x_2 \end{aligned}$$

is a shortest path joining x_1 to x_2 in (X, d). Furthermore, the mapping

$$\begin{aligned} b : X &\longrightarrow W \\ X &\longmapsto w = x - i \end{aligned}$$

is an isometry between (X, d) and (W, d').

Proof: b is a bijection that preserves distances, hence it is an isometry. Let d' be a norm in W. Consider the mapping

$$\begin{aligned} \Gamma' : [0,1] &\longrightarrow W \\ \alpha &\longmapsto \alpha w_1 + (1-\alpha)w_2 \end{aligned}$$

Then Γ' is a shortest path in W, since

$$\begin{aligned} & d'(w_1, \alpha w_1 + (1-\alpha)w_2) + d'(\alpha w_1 + (1-\alpha)w_2, w_2) \\ &= ||(1-\alpha)w_1 - (1-\alpha)w_2|| + ||\alpha w_1 - \alpha w_2|| \\ &= (1-\alpha)||w_1 - w_2|| + \alpha||w_1 - w_2|| \\ &= ||w_1 - w_2|| \\ &= d'(w_1, w_2) \end{aligned}$$

Since (X, d) and (W, d') are isometric, shortest paths in one space are also shortest paths in the other. Therefore, $\alpha x_1 + (1-\alpha)x_2$ is a shortest path in (X, d), joining x_1 to x_2. Note that the converse is not true. In other words, $\alpha x_1 + (1-\alpha)x_2$ can be a shortest path in a metric space (X, d) without d being a norm. An example of this is $X = I$ with the metric $d(x_1, x_2) = |\int_{x_1}^{x_2} \sqrt{1+4t^2}dt|$.
To summarize what has been done so far, consider the three spaces:

1. H_g, the space of observations, when only domain deformation is involved.

2. X_g, the space of domain deformations, constructed in a way to make a bijection with H_g.

3. W_g, the space of domain noise.

Then the following theorem illustrates the links between the various spaces.

Theorem 3.13 *Let d' be a norm metric over W, i.e., $d'(w_1, w_2) = ||w_1 - w_2||$. Let d be the corresponding metric over X, i.e., $d(x_1, x_2) = d'(x_1 - i, x_2 - i)$. Assume further that d is composition invariant, i.e., $d(x_1, x_2) = d(i, x_1 \circ x_2^{-1})$. Let d^* be the corresponding metric over H_g, i.e.,*

$$d^*(f_1, f_2) = d(x_1, x_2) \quad \textit{such that} \quad g = f_1 \circ x_1 = f_2 \circ x_2$$

Then, d^* is a distinction function over H_g. Furthermore, the following mapping

$$\begin{aligned} \Gamma : [a, b] &\longrightarrow H_g \\ \alpha &\longmapsto \Gamma(\alpha) = f : g = f \circ (\alpha x_1 + (1-\alpha)x_2) \end{aligned}$$

is a shortest path in (H_g, d^*) joining f_1 to f_2.

Let us list some examples of distinction functions over H_g:

Supremum $d^*(f_1, f_1 \circ x) =< x >= \sup_t |x(t) - t|$

Maxmax $d^*(f_1, f_1 \circ x) = \max_t(x(t) - t) + \max_t(t - x(t))$

Variation $d^*(f_1, f_1 \circ x) =$ *total variation* of $(x - t) = V(x - t)$. Note that if x is differentiable,

$$V(x - t) = \int_t |\dot{x} - 1| dt$$

3.3 Domain Deformation : Application

Given M functions $h_1, h_2, .., .h_M$ in H_g and a specific choice of the distinction function. The recognition rule is: Choose h_i that minimizes $< x_i >$, where $g = h_i \circ x_i$.

Example 1: Let $h_1(t) = \sin(\pi t)$ for $0 < t < 1$. Let $h_2(t) = 2t$ for $0 < t < 1/2$ and $h_2(t) = 2(1 - t)$ for $1/2 < t < 1$. Let $g(t) = 4t^2$ for $0 < t < 1/2$ and $g(t) = 4(1 - t)^2$ for $1/2 < t < 1$. Decide whether g is closer to h_1 or to h_2 in all of the above metrics.
Solution 1: Let $g = h_1 \circ x_1$. Solving, we get $x_1(t) = \sin^{-1}(4t^2)/\pi$ for $0 < t < 1/2$ with symmetry w.r.t. (1/2,1/2). To find the *supremum* distance we set $\dot{x_1} = 1$ to obtain (using Maple)

$$< x_1 >= 0.187$$

Let $g = h_2 \circ x_2$. Solving, we get $x_2(t) = 2t^2$ for $0 < t < 1/2$ with symmetry w.r.t.

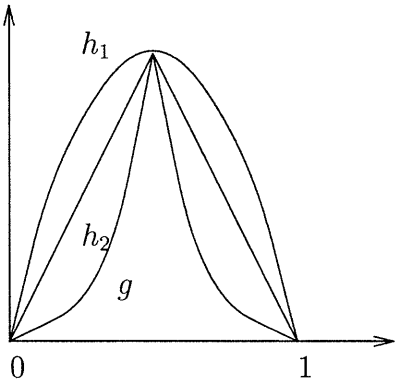


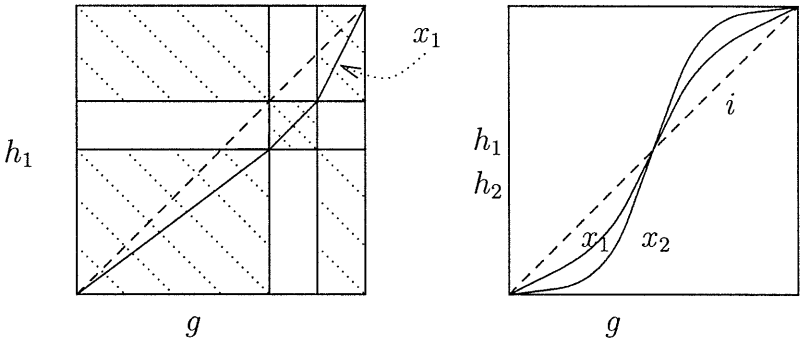
Figure 3-3: Binary hypothesis of Example 1

(1/2,1/2). Similarly, we set $\dot{x_2} = 1$ and we obtain the *supremum* distance

$$< x_2 >= 0.125$$

Since $< x_1 >>< x_2 >$, we say that g resembles h_2 more than h_1 in the *supremum* metric. The *maxmax* metric gives the same answer in this particular case since $\max(x(t) - t) = \max(t - x(t))$ (see Figure 3-4). Finally, the *total variation* metric also gives the same result since it is equal to twice the *maxmax* metric in this case.

In practice, one can view the various solutions of the equation $g(t) = h_i(x(t))$ by



Kernel for $g - h_1$ from Figure 3-1

Domain deformations from example 2

Figure 3-4: Calculation of Deformation

drawing the contour plot of the surface $|g(t) - h_i(x)|$ in the plane of Figure 3-2. The kernel of the surface (i.e., the regions where it is zero) contains all the solutions. This kernel can be viewed using a mathematical package like Matlab. Out of that kernel we choose the curve x that connects the origin (0,0) to the opposite corner (1,1) with minimum $< x >$. Note that unless g is constant for some interval $(a, b) \subset I$, the kernel will be a finite collection of curves. However, only one of them will connect the origin to the opposite corner. If g is constant for some interval, the corresponding part of the kernel will be a rectangle (see Figure3-4). In all of the defined metrics, the reader can verify that taking the diagonal of that rectangle only and dropping the rest results in the minimum $< x >$, which justifies the criteria made earlier in this chapter.

3.4 Domain and Amplitude Deformation : Theory

In the following theorem, which the reader is welcome to verify, we make a slight generalization:

Theorem 3.14 *Let H be the space of all functions of bounded variations whose domain is I. Let X be the space of homeomorphisms over I. Let d be a function over $H \times H$ defined as follows:*

$$\begin{aligned} d(f_1, f_2) &= < x > \text{ if there exists } x \text{ in } X \text{ such that } f_1 = f_2 \circ x \\ d(f_1, f_2) &= \infty \text{ otherwise} \end{aligned}$$

Then, d is a generalized metric over H. Furthermore, we call d a domain metric.

One can visualize H as a collection of disjoint slices where the distance d between any two slices is ∞. In fact, these slices are called *path components* in the topological sense, since any pair of points belonging to the same slice are path connected, and no two points belonging to different slices can be connected.

On the other hand, we can define over H an amplitude metric as follows:

Definition 3.15 *Let d be a metric over H such that for any pair of points f_1 and f_2 in H, $d(f_1, f_2)$ is a function of $f_1 - f_2$ only. Then, d is called an amplitude metric.*

Note that d is a pseudonorm but not necessarily a norm (hint: consider the discrete metric). The most widely used examples of amplitude metrics are the so-called L_p metrics, where for $p \geq 1$,

$$d_2(f_1, f_2) = \left(\int_t |f_1(t) - f_2(t)|^p dt \right)^{1/p}$$

Having defined suitable metrics for domain-only and amplitude-only deformations, how can we use these results to develop a suitable metric for mixed amplitude and domain deformations?

One way to do that is to note that the obtained metrics are also path metrics. In other words, they are defined over path connected spaces in such a way that $d(p, q)$ is equal to the length of the shortest path connecting p to q[31]. Now, envision the space H with two metrics: d_1, an amplitude metric, and d_2, a domain metric. We can view H with these two metrics as a city with two modes of transportation, e.g., *walking* and *train*. $d_1(A, B)$ can be thought of as the time needed to walk from A to B, while $d_2(A, B)$ is the time needed to travel by train. Note that unless A and B are train stations for the same train line, $d_2(A, B) = \infty$. When both means of transportation are possible, we can define the travel time between A and B as that time spent when the optimum use of both means is made. Along the same lines, we can define the cost of deforming a function f_1 to another function f_2 as the cost of optimum combination of amplitude and domain deformation. The recognition problem becomes that of finding the hypothesis that requires the least such cost. In what follows, we present the above analysis in a more precise way.

Let $\Gamma : [a, b] \rightarrow (H, d_1)$ be a path in (H, d_1) joining f_1 to f_2. For any partition of $[a, b]$ given by

$$P = \{t_0, t_1, \ldots, t_m\}$$

the points $\Gamma(t_0), \Gamma(t_1), \ldots, \Gamma(t_m)$ are the vertices of an inscribed polygon. The length of this polygon is denoted $\Delta_\Gamma(P)$ and is defined to be the sum

$$\Delta_\Gamma(P) = \sum_{k=1}^{m} \min(d_1(\Gamma(t_{k-1}), \Gamma(t_k)), d_2(\Gamma(t_{k-1}), \Gamma(t_k)))$$

Definition 3.16 *The distance between f_1 and f_2 is the least upper bound of $\Delta_\Gamma(P)$ over all partitions P of $[a, b]$, i.e.,*

$$d(f_1, f_2) = \sup\{\Delta_\Gamma(P) : P \in \mathcal{P}[a, b]\}$$

Theorem 3.17 *The distance function of Definition 3.16 is a metric.*

The fact that d is a metric comes from its very definition. The reader can refer to Shreider[45] for a discussion on this approach for defining distances. Nevertheless, we

will include the proof here for integrity.

Proof:

1. (Identity) If $f_1 = f_2$, then the constant function $\Gamma(\alpha) = f_1$ is continuous and defines a path of zero length. Therefore, $d(f_1, f_2) = 0$. To prove the converse, assume that $d(f_1, f_2) = 0$. Then, For any partition of the shortest path , we have to have either $d_1(\Gamma(t_{k-1}), \Gamma(t_k)) = 0$ or $d_2(\Gamma(t_{k-1}), \Gamma(t_k)) = 0)$. However, if one of the distances is zero the two points are coincident, hence the other distance should be zero as well. Therefore, all point of the partition are the same. Consequently, $f_1 = f_2$.

2. (Symmetry) follows from the symmetry of d_1 and d_2.

3. (Triangle Inequality) is the easiest to prove. For if there exists a point f such that $d(f_1, f) + d(f, f_2) < d(f_1, f_2)$, it means we can obtain a shorter path joining f_1 to f_2 by having the path pass through f, a contradiction.

The question becomes, given f_1 and f_2, how would we find the shortest path between them? This is not an easy question. Assume that the shortest path is an alternate sequence of domain and amplitude deformations such that, starting from f_1 we have:

$$f_1 \xrightarrow{n_1} f_1 + n_1 \xrightarrow{x_1} (f_1 + n_1) \circ x_1 \xrightarrow{n_2} (f_1 + n_1) \circ x_1 + n_2 \xrightarrow{x_2} \cdots \rightarrow f_2$$

where all n_i's, except possibly n_1, are different from 0. Similarly, all x_i's are different from the identity. Then, we can describe any path emanating from f_1 in the following recursive form:

$$\begin{aligned} p_1 &= f_1 + n_1 \\ p_{i+1} &= p_i \circ x_i + n_i \end{aligned}$$

The path length would be

$$\Delta_\Gamma = \sum_i < x_i > + ||n_i||$$

Our problem becomes: Find x_i and n_i, for $i = 1, 2, \ldots$ that minimizes

$$\sum_i < x_i > + ||n_i||$$

subject to

$$\lim_{i \to \infty} p_i = f_2$$

However, the problem can be simplified remarkably if we choose the amplitude norm $||\ ||$ also to be composition invariant. In that case, we can prove the following theorem:

Theorem 3.18 *Let f_1 and f_2 be two functions defined over the unit interval I. Assume that f_2 was obtained from f_1 by an alternate sequence of domain and amplitude deformation, $\{x_i\}$ and $\{n_i\}$. Assume that the path length corresponding to a domain deformation is $< x_i >$ and that corresponding to an amplitude deformation is $||n_i||$. Let l be the length of the shortest path joining f_1 to f_2. Assume that the amplitude norm $||\ ||$ is composition invariant. Then, l does not increase if we restrict the number of deformations of each kind to be at most one. In other words, we can write*

$$\begin{aligned} f_2 &= f_1 \circ x + n \\ l &= \min_x (< x > + ||f_2 - f_1 \circ x||) \end{aligned}$$

Finally, $d(f_1, f_2) = l$ defines a metric.

To prove it, however, we need one lemma:

Lemma 3.19 *Let H be the space of functions defined over I. Let f_1 and f_2 be 2 elements of H. Let d_1 be a composition invariant amplitude metric over H. Let d_2 be a domain metric over H. Let (x_1, n_1) be the shortest pair of domain-amplitude deformation taking f_1 to f_2, i.e., if we let*

$$D_1 = \{(x, n) : f_2 = f_1 \circ x + n\}$$

then

$$< x_1 > + ||n_1|| \leq < x > + ||n|| \quad \forall (x, n) \in D_1$$

Similarly, let (n_2, x_2) *be the shortest pair of amplitude-domain deformation taking* f_1 *to* f_2*, i.e., if we let*

$$D_2 = \{(n, x) : f_2 = (f_1 + n) \circ x\}$$

then

$$||n_2|| + < x_2 > \leq ||n|| + < x > \quad \forall (n, x) \in D_2$$

Then

$$||n_1|| + < x_1 > = ||n_2|| + < x_2 > \quad \forall (n, x) \in D_2$$

Proof: From the definition of x_1 and x_2 we can write

$$\begin{aligned} x_1 &= \arg\min_x < x > + ||f_2 - f_1 \circ x|| \\ x_2 &= \arg\min_x < x > + ||f_2 \circ x^{-1} - f_1|| \end{aligned}$$

However, the amplitude metric $||\ ||$ is composition invariant. Hence,

$$\begin{aligned} ||f_2 \circ x^{-1} - f_1|| &= ||(f_2 \circ x^{-1} - f_1) \circ x|| \\ &= ||f_2 - f_1 \circ x|| \end{aligned}$$

Therefore,

$$\min_x < x > + ||f_2 - f_1 \circ x|| = \min_x ||f_2 \circ x^{-1} - f_1||$$

i.e.,

$$< x_1 > + ||n_1|| = < x_2 > + ||n_2||$$

Note that n_1 need not be the same as n_2, only the norm is the same. Now, we are in a position to prove Theorem 3.18.

Proof: Let $\{x_i\}$ and $\{n_i\}$ be the sequence of deformations corresponding to the shortest path joining f_1 to f_2. Consider a particular domain-amplitude deformation pair (x_k, n_k), joining $\Gamma(t_{k-1})$ to $\Gamma(t_k)$. Then, using Lemma 3.19, we can replace (x_k, n_k) by another amplitude-domain deformation pair (n'_k, x'_k) without increasing the length. Finally, n'_k can be added to n_{k-1}, and x'_k can be composed with x_{k+1} thereby reducing

the number of points on the partition by 1. Repeating the process, we end up with one pair of amplitude-domain, or domain-amplitude, deformation without increasing the length. Obtaining one pair is equivalent to obtaining the other (see Lemma 3.19). Since every partition of the shortest path results in a polygon having the same length as one pair of domain-amplitude deformation, so is the supremum of all partition lengths. QED.

3.5 Domain and Amplitude Deformation : Application

To illustrate the consequences of Theorem 3.18, assume that we are using the supremum metric for both amplitude and domain deformations. Given two functions f_1 and f_2, Theorem 3.18 implies that the following is also a metric over H,

$$d(f_1, f_2) = \inf_x \left(\lambda_1 \sup_t |x(t) - t| + \lambda_2 \sup_t |f_2(t) - f_1(x(t))| \right) \tag{3.3}$$

where λ_1 and λ_2 are positive weights included for convenience.

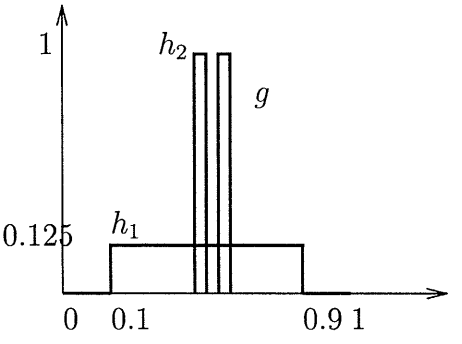


Figure 3-5: Counter example showing weakness of L_2 metric

In the literature on probability metrics, the above discovered metric is called the Skorokhod metric[39]. Using it to solve the recognition problem of Figure 3-1 (redrawn in Figure 3-5), we find after some thought that

$$d(g, h_1) = 0.875\lambda_2$$

$$d(g, h_2) = \min(0.02\lambda_1, \lambda_2)$$

which implies that g is closer to h_2 (in agreement with the human sense) unless $\lambda_1/\lambda_2 > 43.75$. If the last inequality is true, it means that we are assuming a huge cost for domain deformation, and in that case only it is reasonable to say that g is closer to h_1.

Note that we have used the *supremum* norm in Equation 3.3, since it is easier to deal with analytically. However, when running computer minimization algorithms that use gradient descent type techniques, it is more convenient to use the *total variation* norm instead.

3.5.1 Implementation on Matlab

When the functions under consideration cannot be expressed in analytic forms, we can discretize them as follows, and then use MATLAB to find the solution.

Let f and g be given waveforms. The distance between f and g is defined, in terms of the total variation metric, as :

$$d(f, g) = \inf_x \left(0.1 \int |\dot{x} - 1| dt + \int |\frac{d}{dt}(f(x) - g(t))| \right)$$

where the parameters λ_1 and λ_2 have been chosen heuristically. Discretizing t and x into column vectors of length N, the distance $d(f, g)$ is approximated with

$$d(f, g) = \inf_x \left(0.1 \sum_i |(x_i - x_{i-1}) - (t_i - t_{i-1})| + \sum_i |(g(t_i) - f(x_i)) - (g(t_{i-1}) - f(x_{i-1}))| \right)$$

where $f(x_i)$ is obtained using interpolation.
In MATLAB notation, the distance to be minimized over x becomes

```
d=0.1*norm(diff(x-t),1)+norm(diff(g(t)-f(x)),1);
```

To improve the convergence and speed, a multiresolution approach was adopted. First t and x are discretized into $N = 5$ samples. Then, a non-linear optimization algorithm is run. When the termination tolerances for x and d are met, the number of sampling

points is doubled (N=2N-1). The algorithm terminates when $N = 33$.

The following is a MATLAB program, used to generate Figures 3-6 thru 3-8.

```
N=5;
x0=[0 0 0.5 1 1]; % starting guess
options(13)=2; % 2 equality constraints x(0)=0, x(1)-1=0
while N<34,
        [x,options]=constr('fun',x0,options);
        t=linspace(0,1,N); th=linspace(0,1,2*N-1);
        x0=interp1(t,x,th)';
        keyboard; % return control to user to draw plots
        N=2*N-1;
end
----------------------------------------
function [d,c]=fun(x)
N=length(x);
t=linspace(0,1,N);
d=0.1*norm(diff(x-t),1)+norm(diff(g(t)-f(x)),1);
c=[x(1) (x(N)-1) -diff(x)]; % constraints: x is increasing
```

3.5.2 Parameter Selection

How do we choose λ_1 and λ_2 in general? In the following, we describe a trick for estimating λ_1 and λ_2.

Consider the following detection problem: Let $s_i, i = 0, 1, \ldots, M$ be the transmitted 2x1 vector. Let $n = (n_1, n_2)^T$ be an added Gaussian noise vector, with mean 0 and covariance matrix Λ. Let y be the received vector. Under hypothesis H_i, we have

$$y = s_i + n$$

The Bayesian detection rule would be to choose the hypothesis H_i that minimizes

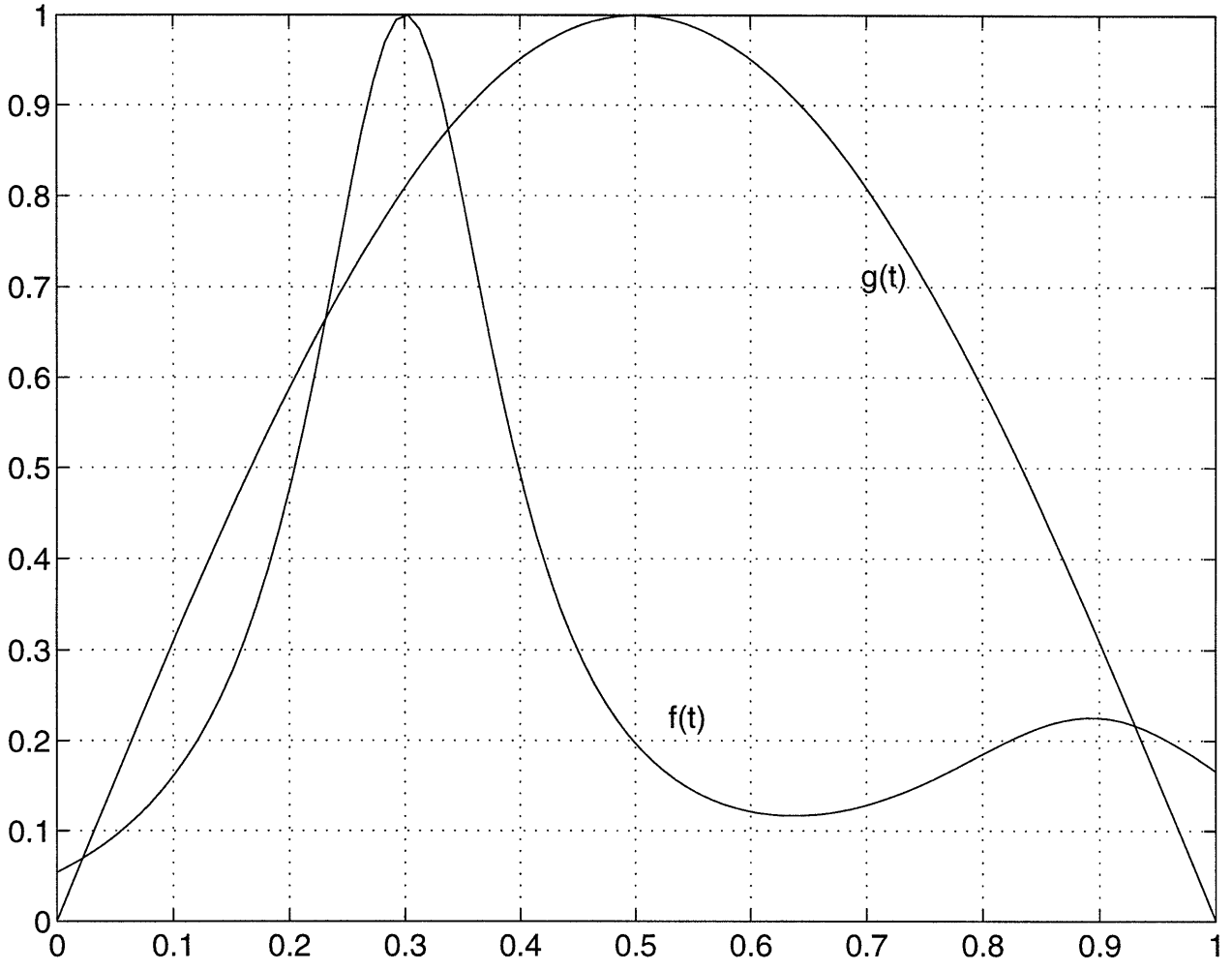


Figure 3-6: $f(t) = \frac{1}{(x-3)^2+0.01} + \frac{1}{(x-9)^2+0.04} - 6$, and $g(t) = \sin(\pi t)$

$(y - s_i)\Lambda^{-1}(y - s_i)$. When the noise components n_1 and n_2 are independent with variances σ_1^2 and σ_2^2, the detection rule simplifies to: Choose i that minimizes

$$\frac{(y_1 - s_{i,1})^2}{\sigma_1^2} + \frac{(y_2 - s_{i,2})^2}{\sigma_2^2}$$

The attentive reader would recognize the similarity between the above equation and Equation 3.3. Indeed, if we let $< x >= (y_1 - s_{i,1})^2$ and $||n|| = (y_2 - s_{i,2})^2$, then λ_1 and λ_2 become synonymous to the inverse of the variances and can be estimated using known techniques.

More precisely, let $h_i^1, h_i^2, \ldots, h_i^N$ be N observed samples of the function h_i. As-

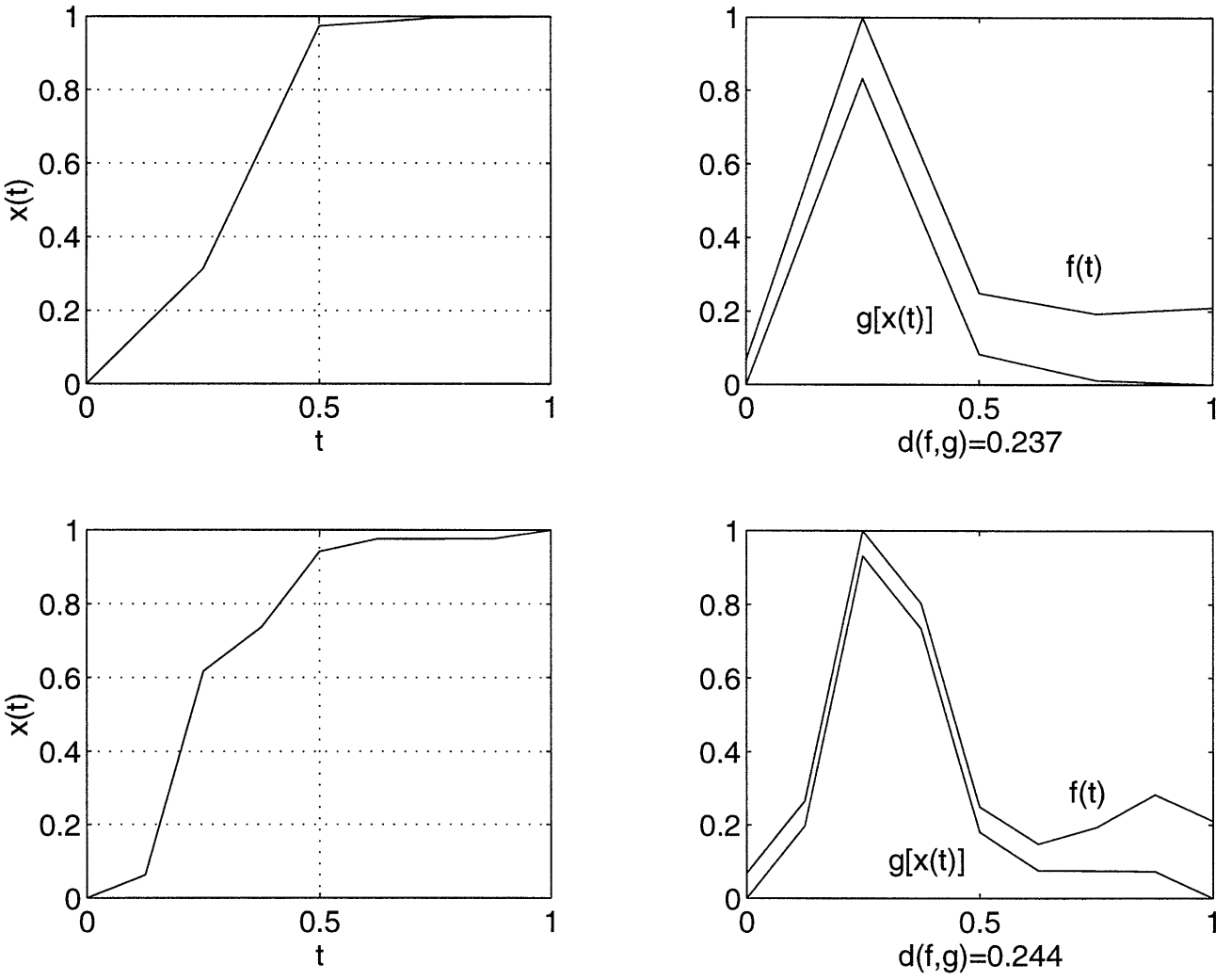


Figure 3-7: Multiresolution to find $\inf_x \left(0.1 \int |\dot{x} - 1| dt + \int |\frac{d}{dt}(f(x) - g(t))|\right)$

sume the domain noise on h_i to be square-Gaussian. Clearly, this assumption is not fully justified, since the distinction function is bounded in several interesting cases. However, we will assume that a Gaussian approximation is good enough for our analysis. Assume also that the amplitude noise on h_i is square-Gaussian, independent from domain noise. Let the domain deformations for the samples be $x^1, x^2, \ldots, x^N$ (determined empirically). Let the amplitude deformations be $n^1, n^2, \ldots, n^N$. The

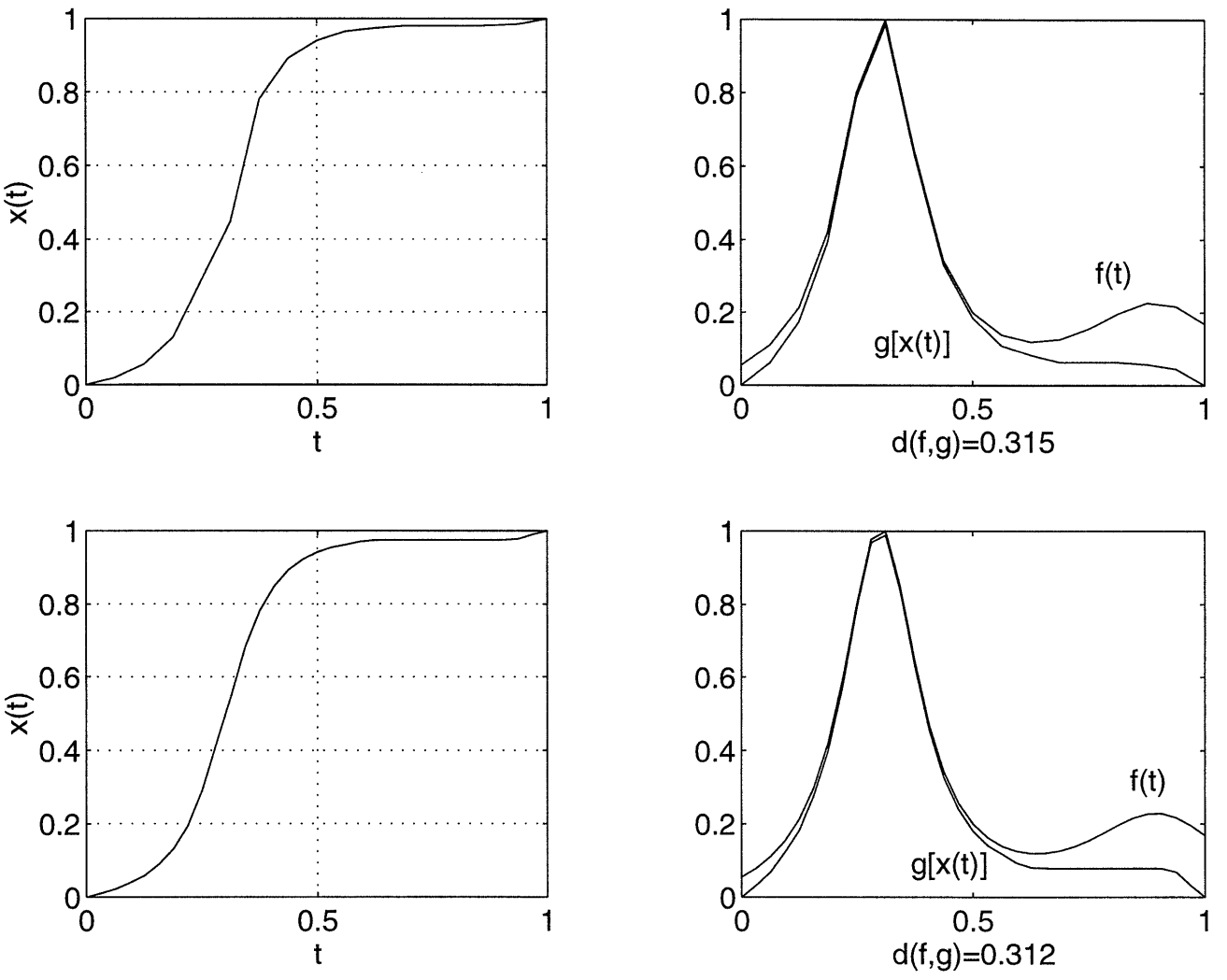


Figure 3-8: Amplitude noise appears as a difference between $g(x(t))$ and $f(t)$

non-biased estimators for λ_1 and λ_2 would be

$$\frac{1}{\lambda_1} = \sum_{i=1}^{N} \frac{< x^i >}{N-1} \qquad \frac{1}{\lambda_2} = \sum_{i=1}^{N} \frac{||n^i||}{N-1}$$

3.6 Observations and Conclusions

Clearly, a great deal had to be done in this chapter to handle the problem of combined amplitude and domain deformations. Part of the complication was that we were trying to find a metric that "commutes" between the two types of deformation. The

reason is that we wanted a cost function that was invariant with respect to the order of the two different types of noise occurrence, that is, invariant with respect to the incremental deformations: dx_i for domain, and dn_i for amplitude, at time t_i. Note that in the classical amplitude-only deformation case, the invariance was guaranteed by choosing a metric which is a function of the sum of differential amplitudes, or, alternatively, a function of the difference between the transmitted and the received signal. This choice guarantees invariance because, obviously, the sum is invariant with respect to the order of the elements being added. Similarly, in the domain-only deformation case, the invariance was guaranteed by choosing a metric that depends on the overall deformation $x(t)$, regardless of the individual domain deformations. Also, by forcing an isometric embedding with the space W, we were able to guarantee that the shortest path of differential deformations dx_i, is actually the line segment connecting the transmitted and received signals in the space W.

When both types of deformations were considered, special care was needed to make sure that, when the different types of deformation intermix, the cost function is not affected. The main result of the last chapter is that this can be guaranteed if the amplitude metric was a composition invariant function of $f - g$, and the domain metric was a composition invariant metric of $x(t) - t$. Examples of such metrics were also given.

Chapter 4

The Principles of Recognition

4.1 Discussion

In the last chapter, we addressed the problem of finding the proper metric for combined amplitude and domain deformation in the single dimensional case. In this chapter, several changes are made:

1. First, the objects of interest are not one dimensional signals but two dimensional signals. These signals are subject to domain deformation in both the x and y direction, as well as amplitude deformation turning pixels on and off in a random manner.

2. Second, as we will shortly argue, we will not be looking for a metric to measure similarity.

3. Third, we need not struggle with the coupling between the two types of deformations. This is because in the on-line recognition case, there is no amplitude deformation that is worth analyzing. On the other hand, in the off-line recognition case, the deformations come from different types of channels that do not interfere with each other. In particular, the domain deformation is due to the hand or printer, and the amplitude deformation is due to ink fading, paper aging, scanner dirty lenses, video camera blurring, or the facsimile machine that was used to transmit the document. In all these interesting cases, amplitude

deformation occurs *after* domain deformation, and the channels can be viewed as consecutive.

Before proceeding, some discussion is needed regarding the representation issue. Should we represent the characters as surfaces over the plane, where the height of the surface corresponds to the grey level of the pixel underneath it? or, should we represent them as regions in the plane, i.e., binary images? or, should we represent them as a collection of (x, y) coordinates in the plane?

Here, we go back to the information-theoretic framework to find that the best representation is the one on which the noise operates directly. When we discuss amplitude noise, the best representation is a surface over the complex plane. In discrete terms (for computer purposes), the character is a matrix whose (i, j) entry contains the grey level at the (i, j) pixel.

When we discuss domain noise, the best representation is the one that describes the domain. For this purpose, a character is a region in the plane. For programming purposes, a character is a finite set of (x, y) coordinates (or complex numbers z).

4.2 Domain Deformation

Clearly, the character templates are binary images, having a value of 1 (foreground) along the trace of the character and 0 (background) outside. For convenience, we define the character domain to be the foreground region only. The consequence is that character templates are constant functions having the value 1 but over different domains.

If we assume for the time being that there is no amplitude noise, the deformed samples will also be constant over their domain. This situation arises in on-line character recognition, where the pen position is sampled whenever it touches the digitizer tablet. All patterns of interest will hence be represented using their domains, i.e., the template T_i is a set of complex numbers denoted $\{z_{T_i}\}$, and the sample S is a set of complex numbers denoted $\{z_S\}$.

If we were to use a metric to measure the similarity between S and T_i, then the

Hausdorff metric[1] would be the natural choice. First, it is a supnorm type metric and we have seen in the last chapter that a supnorm metric has desirable properties in the case of domain deformation. Second, it is one of the mathematically tractable metrics for sets, and can be easily computed using tools from computational geometry.

However, the Hausdorff metric expresses in the clearest fashion the problems that arise from using a metric. A small blob close to the sample can ruin the distance, even if there is – otherwise – a complete matching (see Figure 4-1(d)).

Researchers have attempted to fix the problem by looking for various alternatives. One variation is to define the distance between two sets as

$$H_1(A,B) = \max\left(\sum_{a\in A}\inf_{b\in B} d(a,b), \sum_{b\in B}\inf_{a\in A} d(a,b)\right)$$

effectively converting Hausdorff, which is an L_∞ metric, to an L_1 metric. While this distance also has its own problems, such as confusing 5 and S, or 2 and Z, the problem is more profound. The difficulty lies in the insistence on using a metric, thereby assigning equal weights to the "clean" knowledge (template) and the "noisy" observation (data).

In the following discussion, we present several principles that – we believe – are in close affinity with the human process of recognition. We conclude that a much better measure to use than the Hausdorff metric is the single sided Hausdorff demi-metric (i.e., satisfies triangle inequality) defined from a set A to a set B as:

$$h(A,B) = \sup_{a\in A}\inf_{b\in B} d(a,b)$$

[1]The Hausdorff distance between two bounded sets A and B in a metric space (X,d) is defined as

$$H(A,B) = \max(\sup_{a\in A}\inf_{b\in B} d(a,b), \sup_{b\in B}\inf_{a\in A} d(a,b))$$

4.3 Recognition = Search for patterns

We start by citing an established principle in the text recognition field: the *layering* principle. We follow it by a fundamental principle: the *searching* principle.

Principle 1 (Layering Principle) *The recognition process is a layered hierarchical process representing different levels of abstraction. Sets of points are recognized as characters, and sets of characters as words, and sets of words as concepts.*

This principle is actually a natural byproduct of the information theoretic framework. What we would like to emphasize, however, is that all the recognition principles that follow apply to each level of the hierarchy exhibited by the layering principle.

Principle 2 (Searching Principle) *The recognition process is not a complete matching between what we know and what we observe. Rather, it is a search for what we know in what we observe.*

In other words, when we recognize we do not try to make sense out of everything. Rather, we look for things that make sense to us. Consider the picture of a scratched A in Figure 4-1a. We were able to find an A *in* the picture. We were not able to identify the scratch as a character, so we simply rejected the scratch and decided that this is an A. Similarly, consider in Figure 4-1b the picture of H with serifs, presented to someone who has only seen H without serif. Since an H was found *in* the pattern having serifs, the serifs – which did not make sense to the new observer – were ignored.

Figure 4-1c shows a standard problem in optical character recognition, that of touching. Indeed, the right idea to handle that problem is not to segment the picture before recognition. On the contrary, we look in the whole shape for something we know, we find an F, we also find an E, so we say it is a FE. The final example in Figure 4-1d illustrates how one should deal with yet another challenging problem in OCR, namely that of blobs. Again, we look for the letter M and we find it. We cannot make sense of the blob, and therefore we ignore it.

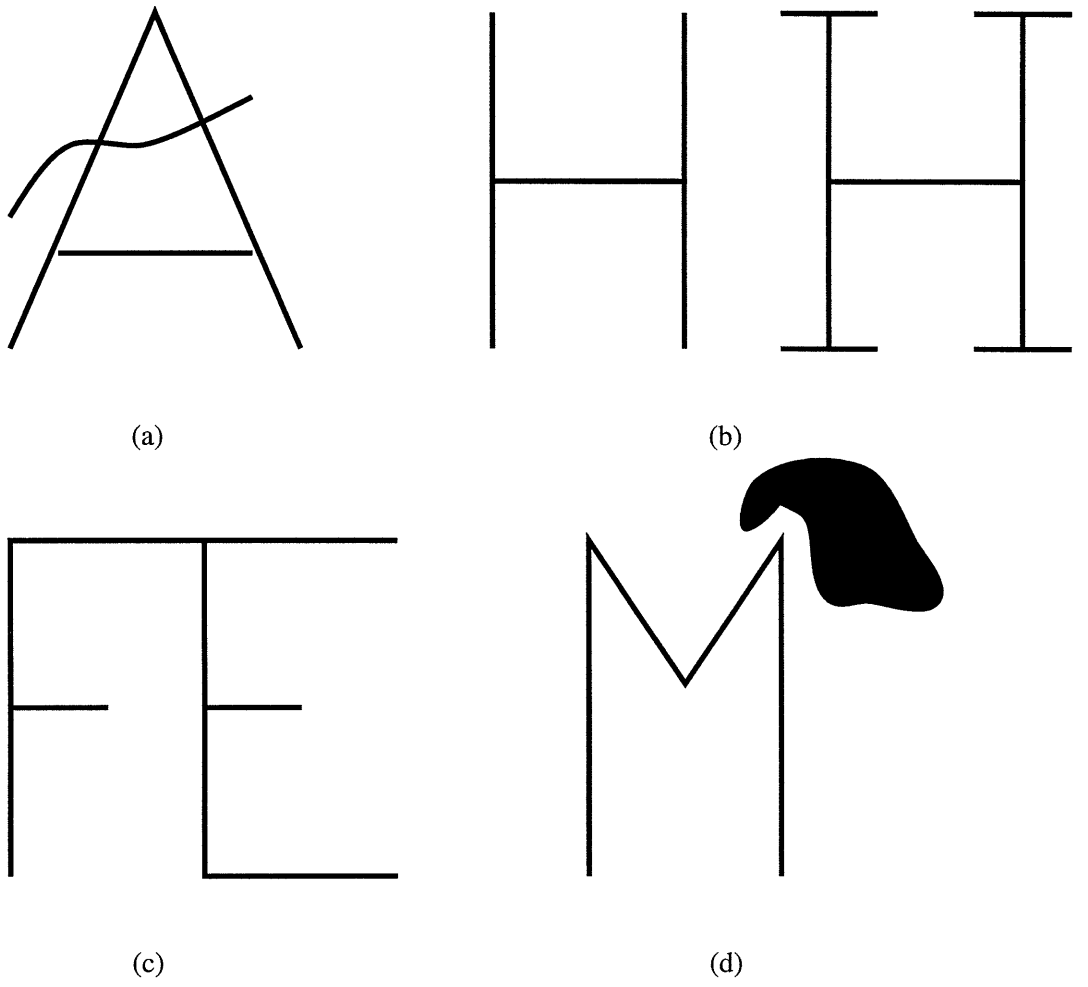


Figure 4-1: Searching over the Space of Defined Templates

4.3.1 Recognition is Asymmetric: Internalist View

All the above examples demonstrate also another concept, that of *asymmetry.* While we may find our known object (template) in the environment (sample/observed data), the opposite is almost never true. When we look at a tableau, we rarely find that same tableau in our memoirs. On the contrary we search that tableau for objects we have learned, and then we try to make sense out of these found objects.

An important consequence of asymmetry is that templates are given more emphasis than the observed data. Furthermore, and contrary to the prevalent view in the OCR field, the design of templates does not require a huge training set. Starting with a particular set of templates such as the definition of letters, recognition is readily possible. By looking at more and more samples of characters, the parameters of al-

lowable deformations can be updated. Note that the deformations themselves do not have to be determined since they are already built in the system. It is only needed to determine the extent to which these deformations are applied. This view is very much in line with the internalist view of linguistics proposed by Chomsky. The internalist view suggests that humans already have an established universal grammar. Through interaction with the environment, the grammar parameters (and not the grammar rules) are adjusted. The consequence is that the size of the required training set is tremendously reduced. After all, why should we need 5 million samples to train an OCR algorithm when probably no human was ever exposed to this large a set in his lifetime?!

If the recognition process is so asymmetrical, it means that a metric is indeed not the right way to measure similarities. Despite its mathematical tractability and the nice properties that are possessed by a metric space, it is – unfortunately – very restrictive.

4.3.2 Supporting Experiments

This anti-metric view is also supported by a set of experiments performed by Mumford and described in[34]. In these experiments, a set of new planar patterns were taught to pigeons and humans, and were later presented for recognition. The confusion matrix[2] turned out to be highly asymmetrical, i.e., object A was confused for B much more than B was confused for A. Mumford also reports additional experiments made by Amos Trevsky that only drives the point home: *a similarity measure is not necessarily obtained from a metric*!

However, the fact that $d(f, g) = 0$ does not imply that $f = g$ raises another question. What if several objects were found in the picture at the same location, which object do we choose? In Figure 4-1c, for instance, we see an F and E, but there is also a 1 (the vertical stroke in either E or F) and another F (the one in E!). This question is related to the third principle of recognition:

[2]The confusion matrix for M patterns is an $M \times M$ matrix, where entry (i, j) contains the probability of deciding j when i is observed.

Principle 3 (Maximum Understanding) *The recognition process is a process of maximum understanding obtainable from knowledge about the space of templates.*

In more fancy terms, ambiguities in recognition are resolved by minimizing the description length of the observed data. To see this, let us consider Figure 4-2. We find in this figure the words LAB, LABOR, ORATORY, and LABORATORY. Why did we choose the last word? simply because it is the *maximum* we can understand from the figure.

LABORATORY

Figure 4-2: Maximize Understanding

To show the equivalence between maximum understanding and minimum description length, imagine you want to encode Figure 4-3, which contains the shape of the letter E using 10 points. Encoding it as such will cost $20K$ bits[3], where K bits are used to represent a real number. Let N be the size of the alphabet, and hence any character be represented in $\log N$ bits. To say that there is a 1 in the picture allows us to encode 5 points (forming the 1) into $\log N$ bits (plus K bits for the position) and the other 5 points using 10K bits, i.e., a total of $\log N + 11K$ bits. To say that there is an F in the picture allows us to encode it in $\log N + 5K$ bits. Finally, to say there is an E in the picture allows us to encode it in $\log N + K$ bits. Hence, maximum understanding of the observed data has resulted in the shortest description.

Consequently, if several templates were found close to the sample, we simply pick the template that has the maximum length of strokes[4]. The sum of the strokes length is in this case taken as a *complexity measure* of the character. If a word is being recognized, we choose the dictionary word that has the largest complexity sum of its characters.

[3]or slightly less if efficient encoding is being used
[4]or the maximum area in the case of off-line

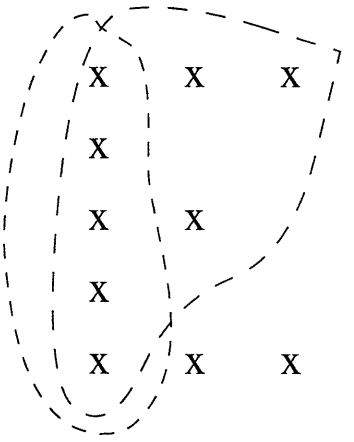


Figure 4-3: Minimum Description Length

We will now break for a moment to see how does amplitude deformation fit in this whole paradigm.

4.4 Amplitude Deformation

As discussed earlier, in almost all the interesting practical cases, the amplitude deformation occurs *after* the domain deformation.

Repeating the definition of a geometric planar pattern from Chapter 2.

Definition 4.1 *Let D be a bounded subset of the plane. Let S be a surface defined over D such that the height of S over D is strictly positive. Then, S is a geometric planar pattern whose domain is D. Let RD be the smallest rectangle (with sides parallel to the axis) containing D. Then, RD is the rectangular domain of S.*

If the amplitude noise is not restricted to operate in the character domain[5], we need to extend the domain of the character to include – possibly – the whole page, by assigning a surface height of zero over the extended region.

In that case, we have

$$P(z) = S(z) + N(z) \quad z \in \mathcal{C}$$

[5]an exception of that case is ink fading

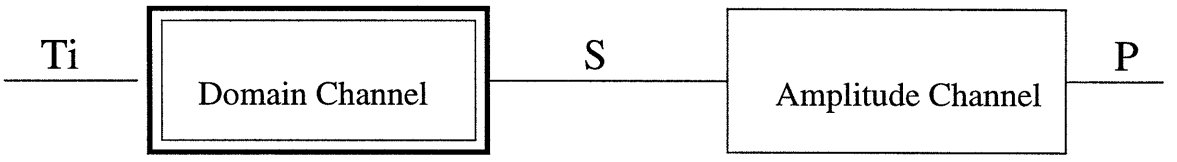


Figure 4-4: Amplitude noise following domain noise

where $N(z)$ corresponds to amplitude noise.

There are several techniques to deal with amplitude noise. Which technique is best depends on the noise distribution. Linear lo-pass filtering does reduce noise, but – unfortunately – at the expense of softening character edges, making (5 and S) or (Z and 2) closer to each other.

To reduce the high frequency noise while preserving edges, two non-linear techniques – among others – are known in the literature[28]: (a) separable median filtering, and (b) out of range smoothing.

The techniques are presented in [28] for restoration from grey level images to grey level images. In what follows, we present a modification of the *out of range smoothing* to handle restoration from grey level images to binary images, since the templates (and their domain deformed version) are binary images.

4.4.1 Out of Range Smoothing

Let $P(z)$ be the amplitude deformed image that we want to restore. We determine $\tilde{S}$, an estimate of S, as follows:

1. Let RD be the rectangular domain of P.

2. Slide a $k \times k$ window over RD, calculating the average of the pixel values, excluding the middle pixel.

3. If the difference between the average and the value of the pixel in the middle of the window is above a certain threshold τ_1, replace the middle pixel value by the obtained average value. Otherwise, nothing is affected.

4. Every pixel value higher than some other threshold τ_2 is converted to one. Otherwise, it is converted to zero.

One may try to combine *single sided Hausdorff* and *Amplitude Smoothing* to do detection for the combined channels. However, we will not discuss that issue in our thesis.

4.5 Emergence of Shape

Finally, we cite the last principle of recognition:

Principle 4 (Allowable Deformations) *We recognize a shape if we can reach it by deforming a template using some chain of allowable rules of deformation, that could be - possibly - template dependent.*

More precisely, a shape from a higher level of abstraction emerges from a collection of shapes from a lower level, if the span of allowable deformations at the higher level intersects the collection of shapes from the lower level.

To grasp the consequence of the last principle, we need to describe the representation at various levels of abstraction.

1. At the pixels level: A point is a complex number.

2. Character: A character is a triplet (label, size, position) associated with a collection of points. Predefined subsets of the character points are called primitives.

3. Word: A word is a triplet (label, size, position) associated with a collection of characters.

4. Concept: A concept is a label associated with a collection of words.

Hence, what the allowable deformations principle says is that a character shape S emerges from a collection of points if there are certain allowable deformations (e.g., slanting by 30 degrees, disturbance of points by % 1 of the character size, etc.) of the character S that makes it match a subset of the collection of points. Similarly, a word shape W emerges from a collection of points (characters) if there are certain allowable deformations of W that makes it match a subset of the collection of points (characters).

4.6 Measure of Similarity

From the previous discussion, we conclude that we need a measure with the properties:

1. It checks if T_i is in S, not if T_i matches S.

2. It is mathematically tractable to minimize over the allowable deformations, such as scaling, slanting, etc.

One candidate for this measure is the single sided Hausdorff distance defined earlier. In that measure, if $d(T_i, S) = 0$, we conclude that T_i is in S. Furthermore, there are algorithms to minimize the single sided Hausdorff distance over translation in $O(N)$ operations, where N is the maximum number of points in either domain. The distance itself can be calculated in $O(N \log N)$ operations. On the other hand, we will discuss in Chapter 5 how to minimize the single sided Hausdorff distance, over translation and allowable linear deformations.

4.7 Putting it All Together

In this chapter we discussed four principles that – we believe – governs any recognition process. We pointed out the usefulness of the single sided Hausdorff distance in implementing these principles. Figure 4-5 summarizes the information theory framework as we view it at this stage for the text recognition problem. The channel is a sequence of five blocks:

1. The first block operates on the word as a whole, with slanting, translation, scaling, etc.

2. The second block operates on individual characters in a word. In this block, touching is manifested through relative displacement of characters. Also, any character slanting different from the word slanting is accounted for in this block.

3. The third block operates on primitives. The horizontal stroke of an A, or the vertical stroke of a 4 is displaced or scaled (or in general deformed) with respect to the character in this block.

4. The fourth block operates on points. The deformation in this block is assumed to be constrained to a small translation in the point position.

5. The fifth block introduces amplitude noise, especially in the form of blobs, scratches or pepper and salt.

The first four blocks combined make the domain channel, which when combined with the amplitude channel constitutes "the channel".

To decode the output of the channel, we start by eliminating the "pepper and salt" amplitude noise. Other types of amplitude noise are handled by the inclusive property of the distance.

If primitive segmentation information is available, the direct decoding strategy would be to go over every possible primitive, checking if block 4 allows such a primitive to look like a subset of the observed data. We continue by inverting the domain channel blocks one by one, effectively estimating characters followed by words.

If primitive segmentation information is not available but character segmentation information is, one decoding strategy would be to go over every character of the alphabet checking if blocks 3 and 4 allows such a character to look like a subset of the observed data. Based on the collection of estimated characters we estimate the word (effectively inverting block 1).

If only word segmentation information is available, the brute force strategy would be to go over every possible dictionary word, checking if the channel allows such a word to look like a subset of the observed data. The trouble is that the dictionary size could be enormous, ruling out the practicality of this approach. Besides, it does not seem that we humans actually go over every dictionary word we know whenever we attempt to read a sequence of characters.

A way around this is to do a combination of top-down and bottom up decoding. In other words, we start at the interface between blocks 1 and 2. We try to detect as many characters in a word as we can. We use this information to select the set of candidate words (effectively sending information from the inner decoder to the outer decoder). From that set of candidate words, we try to look for certain characters in

certain positions (effectively sending information from the outer decoder to the inner decoder) until a decision is made.

The next chapter is devoted to the description of tools required for the implementation of the above algorithm.

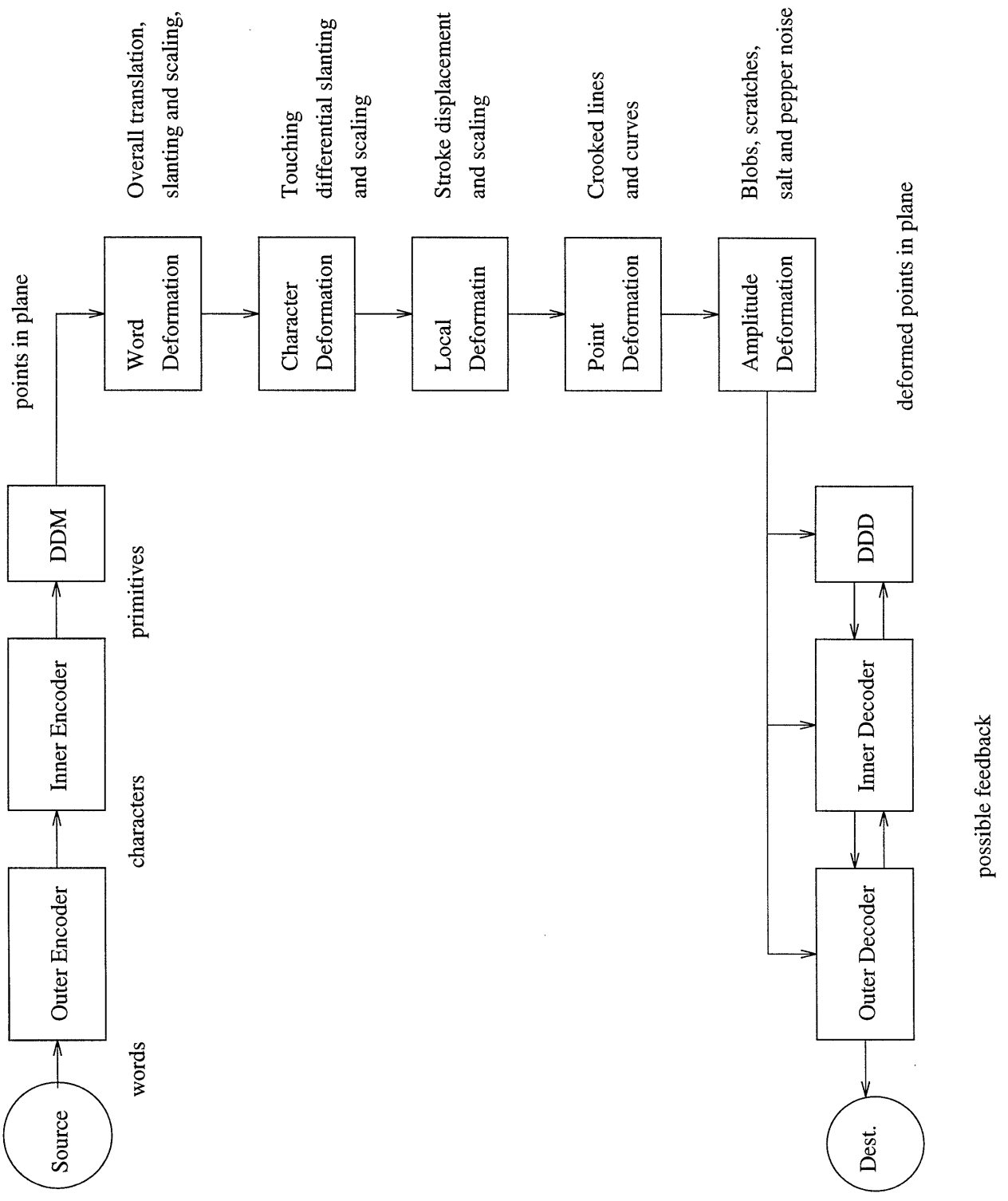


Figure 4-5: Information theory framework revisited

Chapter 5

Implementation and Results

5.1 Required Tools

In this chapter, we will start by presenting few tools. The first tool is the computation of the single sided Hausdorff distance for two sets. The second tool is the minimization of the distance over translation. The third tool is the minimization of the distance, over the space of allowable deformations in several interesting cases. Finally, we present the results of the implementation.

5.1.1 Single Sided Hausdorff

Definition 5.1 *Let $T = \{\alpha_1, \alpha_2, \ldots \alpha_M\}$ and $S = \{\beta_1, \beta_2, \ldots \beta_N\}$ be two finite sets of points in R^2. Let d be the euclidean metric in R^2. Let $h(T, S)$ be calculated as follows:*

$$h(T, S) = \max_i \min_j d(\alpha_i, \beta_j)$$

Then, $h(T, S)$ is the single sided Hausdorff distance from T to S.

Hence, we match to each point $\alpha_i \in T$, the closest point $\beta_j \in S$. $h(T, S)$ becomes the largest distance between matched pairs[1].

[1]Using the minmax theorem, we can also get h by minimizing the maximum distance.

Computing the closest point is a classical problem in computational geometry. The direct approach is to go over every pair of points in T and S, an $O(MN)$ algorithm. However, a better approach is to find the Voronoi diagram of the set S, which takes $O(N \log N)$ operations. Then, for every point in T, we can find the closest point in S in $O(\log N)$ operations. While this might look like an $O(MN \log^2 N)$ algorithm, it is not. The reason is that the Voronoi diagram is calculated for S only once. All the deformations are performed later on T and any calculation of the single sided Hausdorff distance becomes actually an $O(M \log N)$ operation (compared to $O(MN)$ in the direct approach).

Details on computing the Voronoi diagram and the list of closest points can be found in references such as Preparata[38] and Edelsbrunner[7].

5.1.2 Minimize over Translation

Definition 5.2 *Let* $T = \{\alpha_1, \alpha_2, \ldots, \alpha_M\} \subset R^2$. *A* <u>*translation*</u> *of* T *by* $t \in R^2$ *is denoted* $T \oplus t$ *and is defined as the set:*

$$\{\alpha_1 + t, \alpha_2 + t, \ldots, \alpha_M + t\}$$

The problem becomes: Given T and S, find t that minimizes $h(T \oplus t, S)$.

This problem can be solved elegantly using the following observation, which will be referred to again and again. First, without loss of generality, sort the points of S in such a way that the i-th point in S is the closest to the i-th point in T, listing certain points in S more than once if needed. Second, consider only the first M points of S, and call them S_M. Third, view the sets T and S_M as complex row vectors (or as $2 \times M$ matrices). Let $Z = S_M - T$, then $h(T, S)$ becomes the magnitude of the largest entry of Z (or the largest column norm). Note that we have taken the liberty in moving between the complex vector representation of Z and the real matrix representation. These two representations are obviously equivalent, and which one we are using should be clear from the context.

Definition 5.3 *Z, as described above, is called the matching vector (or matrix) from T to S.*

Let us now draw the entries of the complex vector Z in the plane. The distance $h(T, S)$ becomes the radius of the minimum enclosing circle of the points of Z, centered at the origin. When we translate T by t, this results in translating the points of Z by $-t$. Alternatively, this is equivalent to translating the axis of the plane of Z by t. Since we commented that $h(T, S)$ is actually the radius of the smallest enclosing circle of Z centered at the origin, we get the the following theorem:

Theorem 5.4 *Let T and S be two sets in R^2. Let Z be the matching vector from T to S. Let c be the center of the smallest enclosing circle of the entries in Z (regarded as points in R^2). Then, $h(T, S)$ decreases if we translate T by c.*

The next step is to recalculate Z, since the matching between T and S might have been disturbed. If the new calculated Z has a center at the origin we stop, a minima is found. Otherwise, we recalculate the smallest enclosing circle.

Note that the radius of the smallest enclosing circle is guaranteed not to increase at each iteration. The reason is because, for every point in T we can only find a closer neighbor from S than before. Hence, the algorithm converges to a minimum in a finite number of steps.

The question is: is this a global minimum? The answer is: No, it is a local minimum! While this might seem at first a disadvantage of the technique, it is not. Consider, for instance, a line of connected characters containing several versions of the character A, where some are more neatly written than others. Globally minimizing techniques, will find for us only the best written A. Other A's will not be detected. With this technique, however, we can search for local minima, and any minimum that is less than a threshold ϵ is accepted. This way, we guarantee the detection of all the A's that are close enough to the template.

5.1.3 Smallest Enclosing Circle

We have not yet settled the problem of efficiently finding the smallest enclosing circle of M points. This is also a very old problem in Geometry dating back 200 years ago. Making the observation that the smallest enclosing circle is determined by either two or three points out of M, the brute force approach takes $O(M^4)$ operations in the worst case.

However, refinements to the algorithm were made by researchers. $O(M^2)$ algorithm was proposed in 1972 by Elzinga and Hearn[8]. In 1978, Shamos[38] presented an $O(M \log M)$ algorithm using Voronoi diagrams. Finally, in 1982, Megiddo[30] discovered an exciting approach to find the smallest enclosing circle using $O(M)$ operations! We refer the reader to the cited references for more detail on the various algorithms.

5.2 Allowable Deformations

The human eye can easily translate, scale, rotate, scale in one direction or the other, slant, or even do strange non-linear deformations. How can we minimize the single sided Hausdorff distance, over all these deformations?

5.2.1 Scaling

Assume for the moment that the space of allowable deformations contain only the scaling operation, defined as follows:

Definition 5.5 *Let $T = \{\alpha_1, \alpha_2, \ldots, \alpha_M\} \subset R^2$. Let α be a positive real number. Let $P = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix}$. A scaling of T by P, denoted PT, is the set*

$$\{P\alpha_1, P\alpha_2, \ldots, P\alpha_M\}$$

The problem is to find $\min_{P,t} h(PT \oplus t, S)$. The direct approach for finding the nearest local minimum already involves searching in a three dimensional space (two

dimensions for t and one for P). However, a more elegant strategy emerges if we go back to our observation made earlier regarding the matching vector Z, which is now equal to $S_M - PT$ (see Section 5.1.2).

As we vary α, the location of points of Z moves in the plane on straight lines of different slopes. Starting from $\alpha = 1$, we can either increase α or decrease it, and this process is equivalent to moving on each line in one direction or the other. What we care about is not the locations of the Z points, but rather the radius of the smallest enclosing circle. Hence, we check in which direction the radius is increasing, and we move along that direction.

This looks like a tedious computation, recalculating the radius every time. However, a small trick saves a lot. Look only at the point on the boundary of the circle. Most of the times, there are 2 or three points there only, and the radius of the circle formed by these points can be calculated in constant time!

Obviously, when α changes considerably, Z has to be recalculated. However, there is a vast computational savings between this optimized approach and the brute force approach that was used by other researchers[22].

An important remark is that there is an interval that we allow the scaling parameter α to move in. Without this restriction, h can be made zero by arbitrarily decreasing α, for example. The same remark applies to other deformations also. Note however, that the allowable intervals of the parameter deformations need not be the same for every template.

5.2.2 Slanting and Rotation

Assume that the space of allowable deformations $\mathcal{X}$ constitutes of slanting instead of scaling. Then, the same procedure that we just followed applies in this case. The only exception is that the deformation matrix is now $P = \begin{pmatrix} 1 & \tan\alpha \\ 0 & 1 \end{pmatrix}$. $Z = S_M - PT$, and the points of Z move also along a line. We can easily determine the best direction for α, and the rate of change of $\min_t h(PT \oplus t, S)$ along that direction.

In the case of rotation, $P = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$. However, by varying α, the points of Z will move on circles instead of straight lines. One might wonder: what determines the

shape of the curves along which the Z points move? This question will be resolved later. For the time being, our priority is to minimize h, over a richer space of allowable deformations. In the next section, we will consider constrained affine transformations, and we will describe how to compute

$$min_A h(AT, S)$$

where A belongs to a subset of the space of affine deformations $\mathcal{A}$. However, since $\mathcal{A}$ is not a vector space, we will decouple the translation from this space and handle it separately.

5.2.3 Linear Deformations

We begin by making the following definition:

Definition 5.6 *Let T be a template. A deformation $\mathcal{P}$ of T is called linear if it can be represented as premultiplication of T by a 2×2 matrix P. P is called the representative matrix of the deformation $\mathcal{P}$.*

Examples of linear deformations include: rotation, slanting, scaling, etc. Note that translation is not a linear deformation since it cannot be represented by a matrix premultiplication.

If a deformation $\mathcal{P}$ is linear, then there is a bijection between the deformation and its representative matrix P, and the two can be discussed interchangeably.

Let $\mathcal{Y}$ be the space of all linear deformations, then a sequence of linear deformations can be viewed as a path in $\mathcal{Y}$. Corresponding to a template T, let $\mathcal{X}_T$ be the space of allowable linear deformations, then $\mathcal{X}_T \subset \mathcal{Y}$ is a path connected subspace. The problem of recognition boils down to finding a path in $\mathcal{X}_T$ that, with appropriate translation, matches T to a subset of the observed data S.

Note that $\mathcal{Y}$ is a four dimensional vector space, whose basis is the matrices $\left(\begin{smallmatrix}1 & 0\\ 0 & 0\end{smallmatrix}\right)$, $\left(\begin{smallmatrix}0 & 1\\ 0 & 0\end{smallmatrix}\right)$, $\left(\begin{smallmatrix}0 & 0\\ 1 & 0\end{smallmatrix}\right)$ and $\left(\begin{smallmatrix}0 & 0\\ 0 & 1\end{smallmatrix}\right)$. $\mathcal{X}$ is a path-connected subset of $\mathcal{Y}$. For every point P in $\mathcal{Y}$ (and consequently for every point in $\mathcal{X}_T$), there corresponds a cost equal to $\min_t h(PT \oplus$

$t, S)$. We want to find if the cost function attains a minimum less than ϵ anywhere in $\mathcal{X}_T$.

Definition 5.7 *A deformation path is a sequence of deformations $\mathcal{P}_\alpha$, where the index α varies of a subset of the reals. If the deformations are linear, the corresponding sequence of representative matrices is denoted P_α.*

Definition 5.8 *A linear deformation path $\mathcal{P}_\alpha$ has degree n, if all the 4 entries of the matrix P_α are polynomials – of some function of α – of degree at most n.*

For example, slanting paths and scaling paths have degree one, while rotation paths do not have an integer degree.

Lemma 5.9 *When the index α of a deformation path $\mathcal{P}_\alpha$ varies, the Z points move on straight lines if and only if $\mathcal{P}_\alpha$ is linear of degree 1.*

The preceding lemma can be proved by using tools from differential geometry. The reader can refer to Do Carmo[3] for details.

5.2.4 Shape Emergence

Based on the previous discussion, a template T emerges from an observed data S if

$$\min_{\mathcal{P} \in \mathcal{X}_T} \min_t h(PT \oplus t, S) < \epsilon$$

Clearly, the space $\mathcal{X}$ might have several local minima of h. The problem is: how to determine if there is at least one minima which is less than ϵ? The direct approach is to do a gradient-like algorithm, searching for a local minimum, and repeating the process using several starting points in $\mathcal{X}$ (which corresponds to deformations learned from the training set).

Starting from a certain point in $\mathcal{X}$ (e.g., the identity), which direction should we move in to reduce the single sided Hausdorff distance? Since we are dealing with a four dimensional space, we can essentially consider only 8 directions of motion along

the axis, and check which directions reduce the distance. With each good direction, we associate a vector whose norm is equal to the rate of change of the distance in that direction. Finally, we move in the direction of the sum of these vectors. We iterate the process until no movement is possible in any direction. Since the matching vector Z may change after each deformation, it is necessary to update it on a regular basis.

But then, how do we determine if a certain direction increases the distance or decreases it? We go back to the observation made about the matching vector Z in Section 5.1.2. We look at the points of Z lying on the boundary of the smallest enclosing circle (usually two or three). We deform these points in the direction of interset, i.e., by evaluating $(S_M - (P + \delta P)T)$ for δP corresponding to a certain direction. Finally, we compute $\delta h/\delta P$ in that direction.

Whenever we get to the boundary of $\mathcal{X}$, the number of allowable directions of deformations decreases, so as not to leave $\mathcal{X}$.

The above ideas can be generalized to general spaces of operators that include nonlinear deformations. However, this is beyond the scope of this thesis.

5.3 Experimental Results

Instead of testing the previous ideas on a handwriting database such as NIST, we tested them on specific examples to illustrate the cases we can deal with. The following is a sequence of images that were recognized successfully, followed by the sequence of template images. Note that the following samples were recognized successfully despite their peculiar properties, and without any preprocessing of the observed data: a dotted T, a W with serif, a highly distorted A, a w with curved strokes, a 5 where the horizontal stroke is disconnected from the rest of the character. All these characters can also be recognized in the presence of blobs or scratches, etc.

Note how small the training set is. Some characters have only one template. For some characters, such as W and A, we could have also used one template only, but we have used more templates to increase reliability.

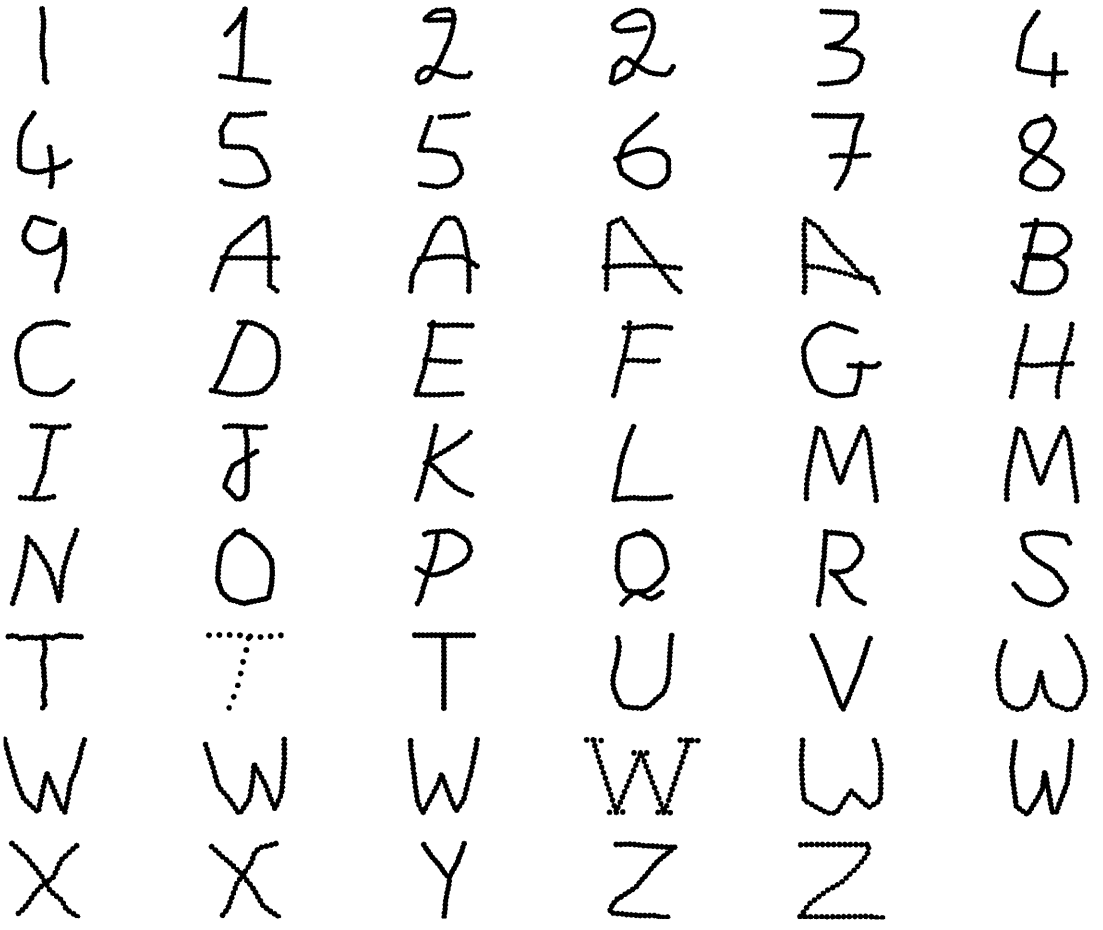


Figure 5-1: Samples of handwritten characters that were recognized correctly.

5.4 When Does it Fail?

The question is: are there cases where the above approach for recognition fail? The answer is: A failure will actually happen if the space $\mathcal{X}$ is does not fully imitate $\mathcal{X}^*$, the "human" space of allowable deformations. In other words, if a certain deformation such as slanting was not included in $\mathcal{X}$, the algorithm may fail to recognize italics. If local deformations are not included, the algorithm may fail to recognize multifont. The various types of deformations that one can include in $\mathcal{X}$ is constrained by the time limit imposed on real life applications. Currently, we can handle linear deformations, including slanting, rotation, scaling in x and y direction, total scaling, together with local deformation in a reasonable time. The combination of top-down and bottom-up recognition makes the program considerably faster. Finally, the algorithm itself is

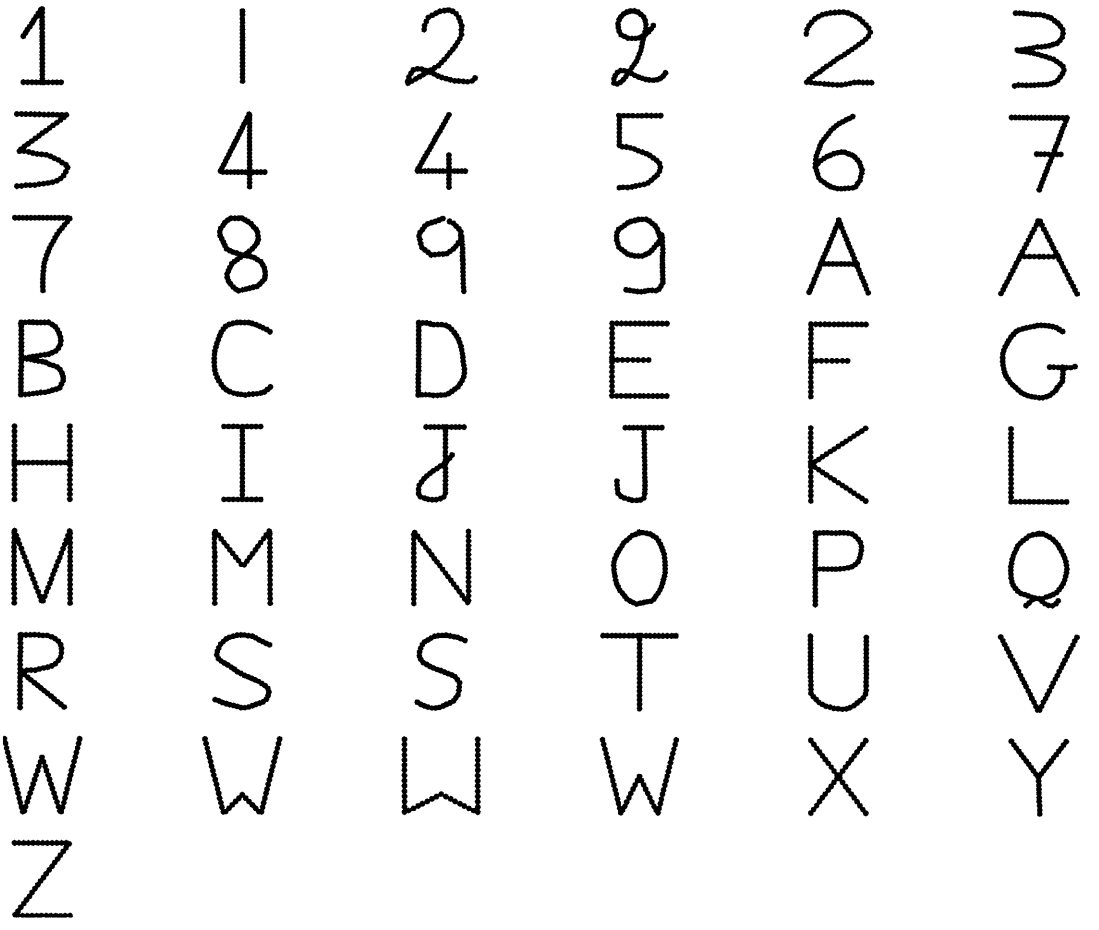


Figure 5-2: Templates used for recognition.

intrinsically parallel, with the consequence that the time needed to recognize a page is almost the same as the time needed to recognize one word.

5.5 Related Research Directions

Before concluding, we would like to select and comment on two research directions that we feel are related to our work in one way or another.

5.5.1 Statistical Geometry

Statistical geometry deals with geometric patterns and probabilistic measures on the space of geometric transformations. It provides tools for image restoration and detection reminiscent of the tools provided by statistics for the field of signal estimation

and detection.

Motivated by this paradigm, Grenander at Brown started a theoretical line of research in pattern analysis since the late sixties[15, 16, 17, 5, 18, 1]. He started by laying down a general framework or language for recognition, which he required to be general, precise, descriptive, structured, and deformation based[15] (see also Mumford[33]). Continuous changes and improvements in the framework were made and updates were published in 1970[16] and 1989[18]. One successful application of his framework was in the recognition of biological hands[5, 1].

However, Grenander emphasized in[16] that pattern theory "will not solve the problem in special cases, but only help in expressing them clearly and concisely". In particular, while the assumptions made and the tools presented and used in[5, 1] may work well for hand recognition, we have several reasons to believe that they are not suitable for text recognition.

1. For instance, when restoring hand images, the domain deformation was assumed continuous. This restriction seems valid in the case of hands (we rarely view a finger at a distance from a hand!), but is not justified in the case of characters. For example, the loop in a handwritten O often becomes disconnected, which is a discontinuous deformation.

2. In addition, the described scheme was designed to handle restoration and recognition of one template only (in that case, the hand). It is not clear how to extend the scheme when more than one template is involved. For instance, allowing arbitrary continuous domain deformations may cause a certain observed data to be described as both K and X, since each character is a continuous deformation of the other. Interaction between templates is a serious problem when large alphabets are considered, e.g., the alphanumeric alphabet.

3. The hand template representation was a connected polygon, describing the boundary of the hand. Characters cannot be represented conveniently as connected polygons.

Regarding the relation between the information theory framework and Grenander

framework for pattern recognition, they have several points in common. Both emphasize the importance of studying deformations. Both allow for integration of probability in the analysis. Both are mathematically sound. However, the information theory framework has two main advantages over Grenander's:

1. It has a well established and standard language.

2. It provides a unified framework for viewing evaluating pattern recognition as developed in this thesis as well as isolating the weaknesses of previous field in this field.

5.5.2 Choice of Hausdorff Metric for Measuring Similarity

A group at Cornell has worked on object recognition and tracking using the Hausdorff distance as a measure of similarity[22, 23]. Since the Hausdorff metric is sensitive to outliers, they modified the metric into a K-th order Hausdorff metric (rather than just abandoning symmetry).The new metric $H_K(A, B)$ is defined as the distance between the K-th furthest matching pairs of A and B as opposed to the furthest matching pairs. A problem arises when computing the *reverse* distance, i.e., the single sided Hausdorff distance *from* the data to the template. It was not clear what part of the observed data should be counted in the distance computation. A heuristic fix was to choose the observed points that lie *under* the rectangular domain of the template. However, another problem arises when attempting to recognize circles in drawings. Since these circles may have lines passing through them, the reverse distance may be large even when there is – otherwise – a complete matching. An additional uncomfortable consequence of using $H_K(A, B)$ instead of the usual $H(A, B)$ is doubling the computational time (or quadrupling the computational time over single sided Hausdorff distance $h(A, B)$). All these problems and others could have been avoided by just dismissing the metric principle from the beginning, instead of viewing it at one point as a kludge to handle scratches. Furthermore, the group attempted to do a complete search of the space of deformations. Even when using multiresolution, this brute force search is time consuming and cannot be generalized to handle more than

a four dimensional space of deformations (translation and scaling in two directions). Searching for one template (a computer system) in an observed image (a room) took 90-250 seconds on a SPARC2 machine.

Appendix A

Modified Fano Algorithm

Let $\boldsymbol{a} = \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_N$ be the transmitted address. Similarly, let $\boldsymbol{r} = \boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_M$ be the received string. Due to the nature of the channel, M, the number of non-empty characters received, can be larger or smaller than N.

Given $\boldsymbol{r}$, the problem is to determine the closest $\boldsymbol{a}$, in a distance measure to be defined later, bearing in mind that this measure should tend to reduce the number of insertions, deletions and substitutions.

Due to the special structure of mail addresses, we can organize them in a tree format with each character represented by a branch. Specifically, the states (assumed 2 letters) all stem from the root, followed by cities, ZIP Codes, street names and numbers. Note that the number of different addresses (i.e. delivery points) in U.S. is 118 million, and hence the tree of addresses just described can easily fit on a few CD-ROM disks.

On the other hand, consider the received string $\boldsymbol{r}$. Construct a graph representation of $\boldsymbol{r}$. It is a linked list having $M+1$ nodes and M branches corresponding to $d_1, \ldots, d_M$. In order to handle incorrect *deletions*, add to every node starting from the root, a loop labeled $\emptyset$. To handle incorrect insertions, join the node i to node $i+2$ with an arc labeled $d_i d_{i+1}$, and do this for $i = 1, \ldots, M-1$. Finally, expand the graph into a tree structure, rooted at node 1, by recursively copying the nodes. The resulting tree is infinite, but can be limited to length 2M for all practical purposes. What we ended up with are two trees, one corresponding to the set of addresses, another

for the received address, modified to allow for incorrect *deletions* and *insertions*. The problem becomes - formally - that of finding the best match between the paths of the two trees, where matching is being performed over paths of equal lengths.

The algorithm we will be using is a modification of the well-known Fano decoding algorithm for convolutional codes. The idea is to start with 2 pointers at the root of both trees. These pointers will move -together- forward or backward[1], until both pointers reach a leaf.

To explain in more detail. Assume that we are at level l in both trees. Let $\boldsymbol{a}_l$ and $\boldsymbol{r}_l$ be the sequences corresponding to the paths traversed so far in both trees. Define the distance measure between $\boldsymbol{a}_l$ and $\boldsymbol{r}_l$ to be

$$\Gamma(\boldsymbol{a}_l; \boldsymbol{r}_l) = \sum_{i=1}^{l} \left(\ln \left(\frac{P(a_l^{(i)}/r_l^{(i)})}{w(a_{l+1}^{(i)})} \right) - B \right) \tag{A.1}$$

In this expression B is an arbitrary bias term to be selected later and $w(j)$ is the nominal probability of the jth letter of the channel output alphabet.

$$w(j) = \sum_{i=1}^{N+1} Q(i)P(j/i)$$

where $Q(i)$ is the relative frequency of the character c_i.

Note that, to make sure that only equal-level paths are considered, infinite cost can be assigned for bypassing a leaf.

At this point, the algorithm proceeds in a fashion similar to Fano. We will adopt the algorithm description from Gallager[13]. We shall allow only three kinds of moves: forward, lateral, and backward. On a *forward move*, both pointers go one branch to the right in both trees from the previously hypothesized node. Since the new sequences $\boldsymbol{a}_{l+1}$ and $\boldsymbol{r}_{l+1}$ differ from the old only in the last position, the new value, $\Gamma(\boldsymbol{a}_{l+1}; \boldsymbol{r}_{l+1})$ can be found from $\Gamma(\boldsymbol{a}_l; \boldsymbol{r}_l)$ by adding one more term to the summation

[1]Lateral move may be performed by only one of them

in Equation A.1.

$$\Gamma(\boldsymbol{a}_{l+1};\boldsymbol{r}_{l+1}) = \Gamma(\boldsymbol{a}_l;\boldsymbol{r}_l) + \left(\ln\left(\frac{P(a_{l+1}^{(i)}/r_{l+1}^{(i)})}{w(a_{l+1}^{(i)})}\right) - B\right)$$

A *lateral move* is a move of only one pointer from one node to another node differing only in the last branch of the tree. A *backward move* is a move one branch to the left of both tree pointers. The new value is calculated from the old value by subtracting the last term in the summation over i in A.1. The rules of motion involve the value Γ_l of the current node being hypothesized, the value Γ_{l-1} of the node one branch to the left of the current node, and a *threshold* T. The value of the threshold T is constrained to change in increments of some fixed number Δ, the changes being determined by the algorithm.

The initial conditions at the beginning are : both pointers at root, $\Gamma_0 = 0$, $\Gamma_{-1} = -\infty$. The algorithm follows rule 1 with the final threshold set at 0.

A forward move could take place by the pointers to any pair of nodes. We assume a predetermined ordering among the nodes. A forward move always occurring at the first node in both trees, and a lateral move occurring by the pointer in the $\boldsymbol{a}$ tree. When the current node in the $\boldsymbol{a}$ tree is the last one, the pointer is reset to the first node, and the $\boldsymbol{r}$ pointer is moved to next node. When both pointers reach the last node, they are moved backward. In other words, we use dictionary order on the nodes of the two trees.

Clearly, rule 1 is the rule that normally applies when the noise is not severe, reaching a final decision in $O(M)$ steps. When the noise is more severe, however, the behavior will be considerably more complicated. It is interesting to observe however, that if M is long enough, this algorithm is guaranteed to provide the correct decoding, i.e., the closest match.

While the constructed tree for the received address relatively simplifies the analysis, their storage requirements are huge. Practically, we can use the graph representation which is more compact. Then, the algorithm can keep 2 stacks, each of width $2L$ to keep track of the path traversed so far. This stack will store the index of the

Rule	Conditions on Node		Action to be Taken	
	Previous move	Comparison	Final Threshold	Move
1	F or L	$\Gamma_{l-1} < T + \Delta$; $\Gamma_l \geq \Delta$	Raise	F
2	F or L	$\Gamma_{l-1} \geq T + \Delta$; $\Gamma_l \geq \Delta$	No change	F
3	F or L	any Γ_{l-1}; $\Gamma_l < T$	No change	L or B
4	B	$\Gamma_{l-1} < T$; any Γ_l	Lower by Δ	F
5	B	$\Gamma_{l-1} \geq T$;any Γ_l	No change	L or B

Table A.1: Rules of Motion for the recognition algorithm

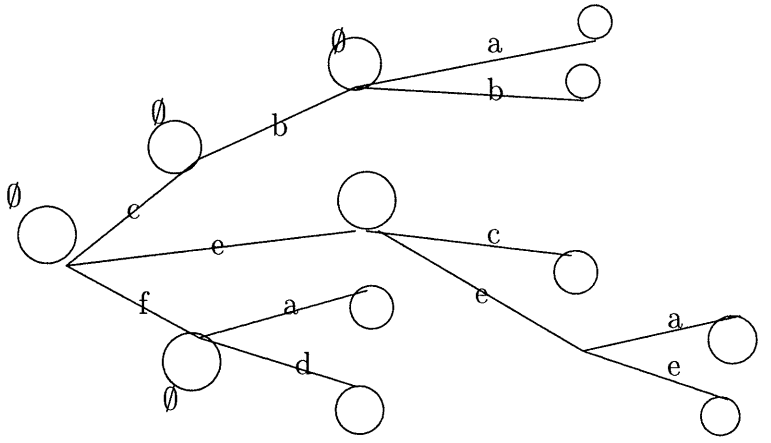


Figure A-1: Modified Tree

branch chosen at every node so that, when the algorithm backtracks, it can simply increment this index and move to next branch, thereby avoiding the possibility of the algorithm falling in an infinite loop. One final word regarding B. For reasons outside the scope of this article, B is to be chosen such that $B = E_0(1, \mathrm{Q})$(see Gallager[13]).

Note that this decoding scheme tends to best decode the beginning of the sequence based on the redundancy available in the tail. Hence, the resulting decoded address will have the correct "state name" with very high probability. This is precisely what is needed in an application such as mail sorting. For the less the errors are in the state and the city name the less are the costs, in time and money, of redirecting the mail back to the proper destination.

Another advantage of this approach is that it decodes correct addresses in no time. This results in a real-time overall average processing, compared to the query-based techniques or n-gram statistics. Abbreviations can be handled in two ways. Either include all possible abbreviations in the tree, e.g. street, st., road, rd, etc... Or, assign

a low cost for dropping subsequent characters from the original address.

Bibliography

[1] Y. Amit, U. Grenander, and M. Piccioni. Structural image restoration through deformable templates. Technical Report CICS-P-205, Center for Intelligent Control Systems, M.I.T., Cambridge, Massachusetts, April 1990.

[2] R. Blahut. *Fast Algorithms for Digital Signal Processing*, section 12.4, pages 399–404. Addison-Wesley, Reading, Massachusetts, 1985.

[3] M.P. Do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

[4] R.G. Casey and G. Nagy. Decision tree design using a probabilistic model. *IEEE Transactions on Information Theory*, 30(1), January 1984.

[5] Y. Chow, U. Grenander, and D. Keenan. Hands: A pattern theoretic study of biological shapes. Technical report, Brown University, Providence, Rhode Island, 1988.

[6] P.A. Chu. *Applications of Information Theory to Pattern Recognition and the Design of Decision Trees and Trellises.* PhD dissertation, Stanford University, Department of Electrical Engineering, 1988.

[7] H. Edelsbrunner. *Algorithms in Combinatorial Geometry.* Springer-Verlag, Berlin, Germany, 1987.

[8] J. Elzinga and D.W. Hearn. Geometrical solutions for some minimax problems. *Transportation Science*, 6:379–394, 1972.

[9] G.D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3), March 1973.

[10] G.D. Forney, Jr. Concatenated codes. The MIT Press, Cambridge, Massachusetts, 1966.

[11] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, second edition, 1990.

[12] R. Gallager. Sequential decoding for binary channels with noise and synchronization errors. Lincoln Lab Group Report 25-G-2, October 1967.

[13] R. Gallager. *Information Theory and Reliable Communication.* Wiley, New York, 1968.

[14] Postmaster General. Annual report of the Postmaster General. Technical report, U.S. Postal Service, Fiscal Year 1991.

[15] U. Grenander. Foundations of pattern analysis. Technical Report 2, Brown University, Providence, Rhode Island, 1967.

[16] U. Grenander. A unified approach to pattern analysis. *Advances in Computers*, 10:175–216, 1970.

[17] U. Grenander. *Pattern Analysis*, volume 2 of *Lectures in Pattern Theory.* Springer-Verlag, New York, New York, 1978.

[18] U. Grenander. Advances in pattern theory. Technical Report CICS-P-121, Center for Intelligent Control Systems, M.I.T., Cambridge, Massachusetts, March 1989.

[19] L. Grunin. OCR software. *PC Magazine*, 9(18):299–354, October 1990.

[20] J. Hertz, A. Krogh, and R. Palmer. *Intorduction to the Theory of Neural Computation.* Addison Wesley, Reading, Massachusetts, 1991.

[21] J.J. Hull and S.N. Srihari. Experiments in text recognition with binary n-gram and Viterbi algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(5), September 1982.

[22] D.P. Huttenlocher and W.J. Rucklidge. A multiresolution technique for comparing images using the Hausdorff distance. Technical Report 1321, Cornell University, Ithaca, New York, December 1992.

[23] D.P. Huttenlocher and W.J. Rucklidge. Tracking non-rigid objects in complex scenes. Technical Report 1320, Cornell University, Ithaca, New York, December 1992.

[24] S. Impedovo, L. Ottaviano, and S. Occhinegro. *Character @ Handwriting Recognition : Expanding Frontiers*, chapter Optical Character Recognition - A Survey, pages 1–24. World Scientific, 1991.

[25] R.L. Kashyap and B.J. Oommen. Spelling correction using probabilistic methods. *Pattern Recognition Letters*, 2(3), March 1984.

[26] A. Kolmogorov and S. Fomin. *Elements of the Theory of Functions and Functional Analysis.* Graylock Press, Rochester, New York, 1957.

[27] J. Kulikowski. *Algebraic Methods in Pattern Recognition : Course Held at the Department of Automation and Information, July 1971, Udine 1971.* Springer-Verlag, Wien, New York, 1972.

[28] J.S. Lim. *Two-Dimensional Signal and Image Processing*, section 8.2, pages 468–476. Prentice Hall, Englewood Cliffs, New Jersey, 1990.

[29] L. Lindgren. Machine recognition of human language. *IEEE Spectrum*, May 1965.

[30] N. Megiddo. Linear-time algorithms for linear programming in R^3 and related problems. *SIAM Journal of Computing*, 12(4), November 1983.

[31] J. Milnor. *Morse Theory.* Princeton University Press, Princeton, New Jersey, 1973.

[32] MIT. *Technology Review*, May/June 1992. publihsed by MIT alumni association.

[33] D. Mumford. Pattern theory : A unifying perspective. Preprint.

[34] D. Mumford. Mathematical theories of shape: Do they model perception? *Geometric Methods in Computer Vision*, 1570, 1991.

[35] D.L. Neuhoff. The Viterbi algorithm as an aid in text recognition. *IEEE Transactions on Information Theory*, March 1975.

[36] T. Okuda, E. Tanaka, and T. Kasai. A method for the correction of garbled words based on the Levenstein metric. *IEEE Transactions on Computers*, 25(2), February 1976.

[37] R. Plamondon, C.Y. Suen, and M.L. Simner, editors. *Computer Recognition and Human Production of Handwriting.* World Scientific, Singapore, New Jersey, 1989. Proceedings of the Third International Symposium on Handwriting and Computer Applications held at Montreal on July 20-23, 1987. Call Number : TA1650.C66 1987.

[38] F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, 1988.

[39] S.T. Rachev. *Probability metrics and the stability of stochastic models.* Wiley, Chichester, New York, 1991.

[40] E. Riseman and A. Hanson. A contextual postprocessing system for error correction using binary n-grams. *IEEE Transactions on Computers*, 23(5), May 1974.

[41] H.F. Schantz. *The History of OCR, Optical Character Recognition.* Recognition Technologies Users Association, Manchester Center, Vermont, 1982.

[42] Internal Revenue Service. Annual report, Fiscal Year 1990.

[43] C.E. Shannon and Weaver W. *A Mathematical Theory of Communication.* University of Illinois Press, Champaign, Illinois, 1964 edition, 1949.

[44] R. Shinghal and G.T. Toussaint. A bottom-up and top-down approach to using context in character recognition. *International Journal on Man-Machine Studies*, 11:201–212, 1979.

[45] I. Shreider. *What is Distance?* University of Chicago Press, Chicago, 1974.

[46] M.J. Usher. *Information Theory for Information Technologists.* McMillan Publishers Ltd, Great Neck, New York, 1984.

[47] C.M. Vossler and N.M. Branston. The use of context for correcting garbled english text. In *Proceedings of the 19th National Conference*, pages D2.4.1–D2.4.13. Association for Computing Machinery, 1964.