Advanced correlation-based character recognition applied to the Archimedes Palimpsest

Derek J. Walvoord

Advanced Correlation-Based Character Recognition Applied to the Archimedes Palimpsest

by

Derek J. Walvoord

B. S. Rochester Institute of Technology, 2002

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the Chester F. Carlson Center for Imaging Science
Rochester Institute of Technology

May 2008

Signature of the Author ______________________________

Accepted by ______________________________
Coordinator, Ph.D. Degree Program Date

CHESTER F. CARLSON CENTER FOR IMAGING SCIENCE
ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

Ph.D. DEGREE DISSERTATION

The Ph.D. Degree Dissertation of Derek J. Walvoord
has been examined and approved by the
dissertation committee as satisfactory for the
dissertation required for the
Ph.D. degree in Imaging Science

Dr. Roger L. Easton, Jr., dissertation Advisor

Dr. Maria Helguera

Dr. Carl Salvaggio

Dr. Roxanne Canosa

Dr. Andrew Herbert

Date

Advanced Correlation-Based Character Recognition Applied to the Archimedes Palimpsest

by

Derek J. Walvoord

Submitted to the
Chester F. Carlson Center for Imaging Science
in partial fulfillment of the requirements
for the Doctor of Philosophy Degree
at the Rochester Institute of Technology

Abstract

The *Archimedes Palimpsest* is a manuscript containing the partial text of seven treatises by Archimedes that were copied onto parchment and bound in the tenth-century AD. This work is aimed at providing tools that allow scholars of ancient Greek mathematics to retrieve as much information as possible from images of the remaining degraded text. A *correlation pattern recognition* (CPR) system has been developed to recognize distorted versions of Greek characters in problematic regions of the palimpsest imagery, which have been obscured by damage from mold and fire, overtext, and natural aging. Feature vectors for each class of characters are constructed using a series of spatial correlation algorithms and corresponding performance metrics. *Principal components analysis* (PCA) is employed prior to classification to remove features corresponding to filtering schemes that performed poorly for the spatial characteristics of the selected region-of-interest. A probability is then assigned to each class, forming a character probability distribution based on relative distances from the class feature vectors to the ROI feature vector in principal component (PC) space. However, the current CPR system does not produce a single classification decision, as is common in most target detection problems, but instead has been designed to provide intermediate results that allow the user to apply his or her own decisions (or evidence) to arrive at a conclusion. To achieve this result, a probabilistic network has been incorporated into the recognition system. A probabilistic network represents a method for modeling the uncertainty in a system, and for this application, it allows information from the existing partial transcription and contextual knowledge from the user to be an integral part of the decision-making process.

The CPR system was designed to provide a framework for future research in the area of spatial pattern recognition by accommodating a broad range of applications and the development of new filtering methods. For example, during preliminary testing, the CPR system was used to confirm the publication date of a fifteenth-century Hebrew colophon, and demonstrated success in the detection of registration markers in three-dimensional MRI breast imaging.

In addition, a new correlation algorithm that exploits the benefits of *linear discriminant analysis* (LDA) and the inherent shift invariance of spatial correlation has been derived, implemented, and tested. Results show that this composite filtering method provides a high level of class discrimination while maintaining tolerance to within-class distortions. With the integration of this algorithm into the existing filter library, this work completes each stage of a cyclic workflow using the developed CPR system, and provides the necessary tools for continued experimentation.

Acknowledgements

First and foremost, I acknowledge my advisor, Roger Easton, for convincing me to extend my graduate studies to the doctoral level and for giving me the confidence to do so. I thank you for welcoming me into the fascinating world of the Archimedes Palimpsest, a manuscript that almost seems to laugh at pattern recognition. Your work ethic, dedication to teaching, and your ability to simplify are a constant inspiration to me. You are a great mentor and a true friend.

I acknowledge my graduate committee for providing a great deal of assistance throughout this endeavor. I owe many thanks to Maria Helguera, who devoted many hours toward this research and always kept an open door policy when I needed it most. I thank Carl Salvaggio for providing useful feedback and expressing sincere interest in my work. I thank Roxanne Canosa for introducing me to the probabilistic models that eventually found their way into this work. I thank Andrew Herbert for useful suggestions, and of course, many great discussions about the great sport of hockey.

I express my gratitude to the members of the Archimedes Palimpsest Project for their outstanding work, much of which has made the work presented in this dissertation possible. I thank Will Noel, Keith Knox, and Mike Toth for all that they have done for me over the past few years. I especially thank the project's most important member, The Owner of The Archimedes Palimpsest, for the generosity and vision that led to many valuable results during the course of this work.

I wish to acknowledge Alvin Spivey, with whom I have shared numerous, lengthy conversations on mathematical theory, philosophy, as well as other topics, which were probably less productive, but certainly as entertaining. I also thank Allison Bright for joining me in the painstaking process of extracting training and test sets of characters from the images of the Archimedes Palimpsest.

There are numerous others, both students and faculty who have shared their knowledge, experience, and most importantly, their time with me during my academic career. You have all played a part in making CIS my "home away from home."

To my wonderful parents, for their constant guidance and support. I am forever grateful for all that you have done for me.

To my brilliant, yet down-to-earth brother, Ryan, for providing additional motivation to conclude this work. He completed his undergraduate degree (in its entirety) and began his own graduate studies during the course of my doctoral research. I wish you the best of luck in all of your academic endeavors.

Lastly, and most importantly, to my loving wife, Kandace, for her encouragement, patience, and perseverance. This work would not have been possible without you in my life. You were my guide when I could not see the light at the end of the tunnel.

Contents

List of Figures

–Give me a place to stand, and I will move the Earth.

Archimedes

–If I have seen further than others, it is by standing on the shoulders of giants.

Isaac Newton

1 Introduction

Patterns exist universally. They are pervasive in nearly every aspect of our daily lives. We, as humans, are naturally inclined to seek out and recognize these existing patterns using our precisely tuned abilities for discriminating ordered arrangements or sequences. Over time, these abilities become subconscious routines used to recognize visual or aural patterns such as characters and words as we read, or friends and family members that we hear on the telephone. Even more impressive is the fact that humans can recognize patterns that deviate from what we consider to be the norm. For example, we can typically recognize the faces of individuals at a class reunion despite changes in their appearance due to aging.

Despite its many obvious strengths, pattern recognition in humans is not without limitation. The popular "Where's Waldo?" books by Martin Handford [18] provide children with the difficult task of finding the hidden Waldo character in a busy scene with numerous impostors. Similarly, the recognition of fingerprints is typically an intractable problem for the human pattern recognition system due to the complexity of the images. Our recognition capabilities are inhibited by clutter, speed, and intensity scaling.

To design a pattern recognition system that could compete with the inherent recognition ability of the average human observer is incomprehensible. Fortunately, technology has opened the door for the development of new machine-driven pattern recognition systems that can overcome many obstacles. The majority of this dissertation will focus on spatial pattern recognition algorithms, that is, methods by which information from the relative spacing between intensity values is used to appropriately classify an input signal (or image). In particular, correlation filter design is emphasized as a robust method for extracting spatial information to be used for image classification.

With this in mind, we now turn to an unsuspecting application for pattern recognition development, the *Archimedes Palimpsest*. This ancient manuscript is of great historical significance, as it contains the earliest known copies of text from Archimedes, the great mathematician. Ironically, many of the mathematical concepts derived therein serve as the foundation for the algorithms used in the retrieval of its text [39].

1.1 The Archimedes Palimpsest Project

The Archimedes Palimpsest is a Byzantine manuscript containing partial texts of seven treatises by Archimedes (287 BC–212 BC) that were copied onto parchment and bound into a codex in the tenth-century, AD. Among these seven treatises is the only extant copy of *On the Method of Mechanical Theorems*, which provides insight into the mathematical thought process of Archimedes [10]. The manuscript also includes the only copy of *On Floating Bodies*, Archimedes' most famous work, in the original Greek and of *Stomachion*, which recently has been identified as a very early study in combinatorics [40]. In addition, five folios of speeches by the Athenian orator Hyperides and six and a half folios of a commentary on a work by Aristotle were recently discovered in the palimpsest.

The most likely origin of the manuscript was the city of Constantinople, which was sacked in 1204 at the time of the Fourth Crusade. Without need of mathematical treatises, the book was disbound, the original text was erased, and the pages were cut in half, rotated ninety degrees, and rebound along with pages from other manuscripts to form a palimpsest. The erased pages were then overwritten with a prayer book, the

Euchologion, which was used in services for approximately 700 years.

In 1906, the palimpsest resurfaced in Istanbul when Johan L. Heiberg, a Danish philologist, had photographs made of several leaves to support his transcription of what remained of the Archimedes text. In 1998, the Archimedes Palimpsest was auctioned at Christies, Inc., in New York, and sold for two million dollars to an anonymous American collector. The manuscript has been lent to the Walters Art Museum in Baltimore for conservation and study.

Little is known about where the palimpsest resided during the twentieth century, but it was recently discovered that a page from the palimpsest was photographed in the USA in 1932. Significant damage suffered by the manuscript in the past one hundred years is evident from a comparison of the current appearance of the palimpsest to photographs taken at the beginning of the twentieth-century. The damage includes burn marks, mold growth, and even deliberate painting of forgeries over four of the leaves, in an apparent attempt to increase its sale value.

1.1.1 Early Transcription Efforts

A series of imaging sessions began in the summer of 2000 to capture and preserve the current state of the manuscript and to exploit image processing methods on the digitized data. Figure 1.1 shows a small section in the gutter region of the disbound Euchologion bifolio 98v-102r under strobe illumination. While the text of the Euchologion (running vertically) exhibits high contrast, the text of Archimedes (running horizontally) and a diagram of his treatise, *On Spiral Lines*, are barely discernible in the background.

The same section of the palimpsest under ultraviolet illumination is shown in Figure 1.2. Ultraviolet light causes the material in the parchment to *fluoresce*. Short wavelength photons $(\lambda = 365nm)$ are absorbed by the parchment and re-emitted at longer wavelengths, while the inks absorb some amount of both the incident and fluorescing light [10]. The result is that both texts appear dark, and the fluorescing parchment appears as a brighter blue color, effectively enhancing the contrast between the text and background regions.

Images of reflected light were collected at various wavelengths, using three different illuminations and five different tunable filters, to create a multispectral data set [10]. The initial approach used to provide scholars with readable Archimedes text was

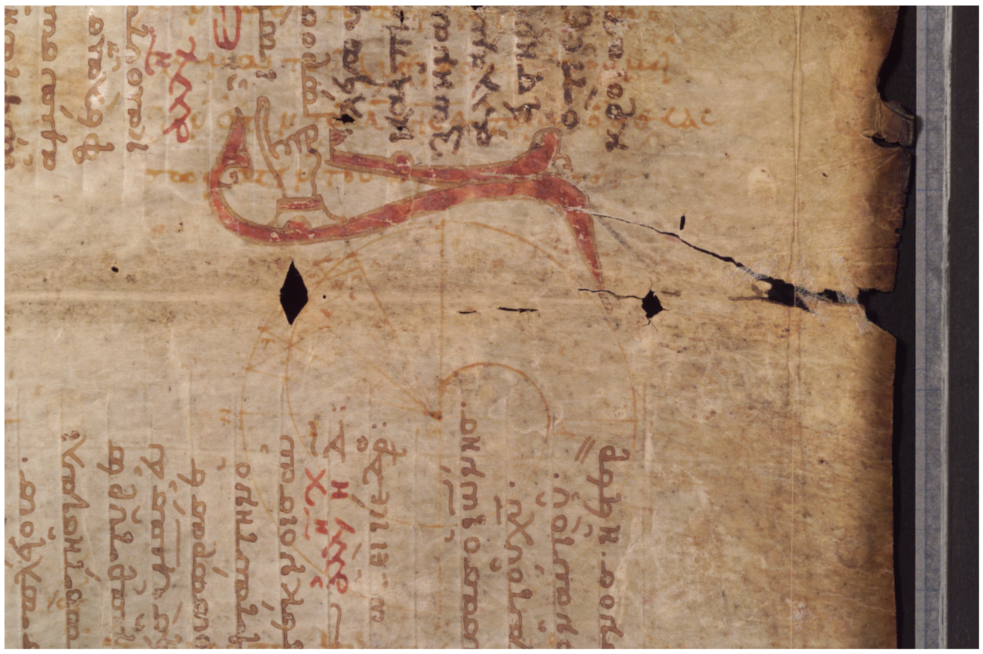


Figure 1.1: A section of leaves 98v-102r from the Archimedes Palimpsest under strobe illumination. The Archimedes text (underwriting) is shown running vertically. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

to apply a *supervised classification* algorithm of *least-squares spectral unmixing* to the imagery. Images were then registered, and four classes were selected for segmentation: parchment, overwriting, underwriting, and mold. Figure 1.3 shows images of leaf 28r of the palimpsest under normal white-light illumination and of the underwriting text class map, where the gray value is a measure of the membership in this class (underwriting appears as white and nonunderwriting appears as black).

Despite the imaging team's belief that the least-squares spectral unmixing results were useful, the scholars deemed them to be insufficient for their transcription efforts. Breaks in the characters led to a large amount of ambiguity as to whether an overwritten character had occupied a particular location prior to the processing. A much simpler result was desired by the scholars that would preserve the visibility of both writings, while distinguishing the texts in some manner. The imaging team developed a much simpler processing technique that produced pseudocolor images where

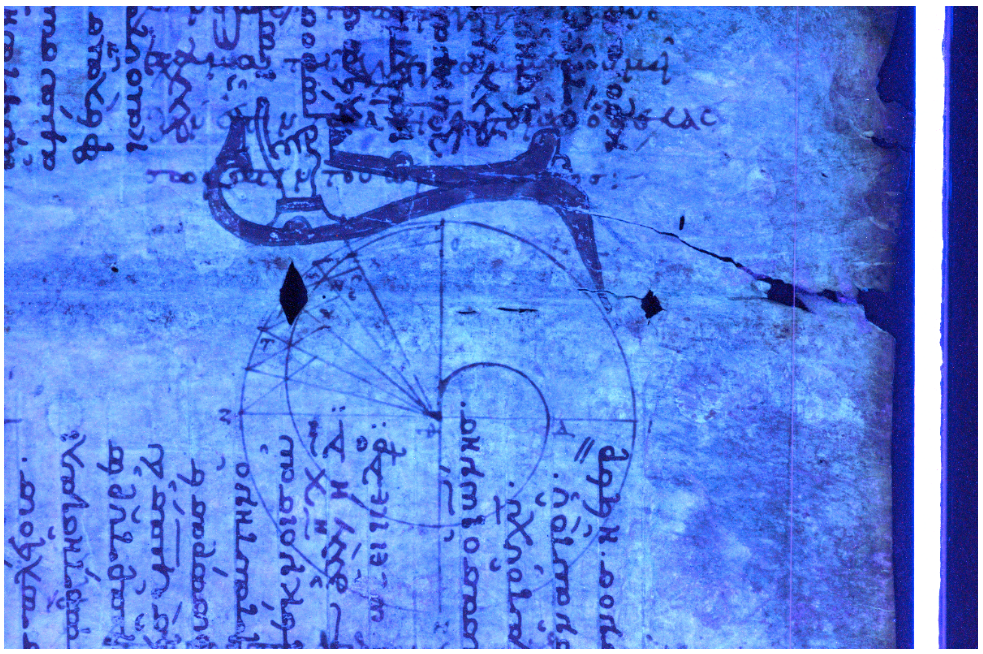


Figure 1.2: A section of leaves 98v-102r from the Archimedes Palimpsest under ultraviolet illumination. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

the underwriting, overwriting, and parchment classes appear as different colors. The process is based on the observation that the underwriting is barely detectable in the red channel of the tungsten images, but that both texts are visible with high contrast in the blue channel of the UV images. False-color images are generated by combining the tungsten-red image in the red channel and the UV-blue image in the green and blue channels. Figure 1.4 shows the result for a section of the Euchologion bifolio 98v-102r after processing.

1.1.2 Motivation for Additional Text Recovery

By early 2004, scholars had transcribed approximately eighty percent of the Archimedes text from images of the palimpsest under strobe and UV illumination, from pseudocolor-processed images, and from images of available Heiberg photographs. While multispectral imaging techniques were largely successful, some areas of text were still

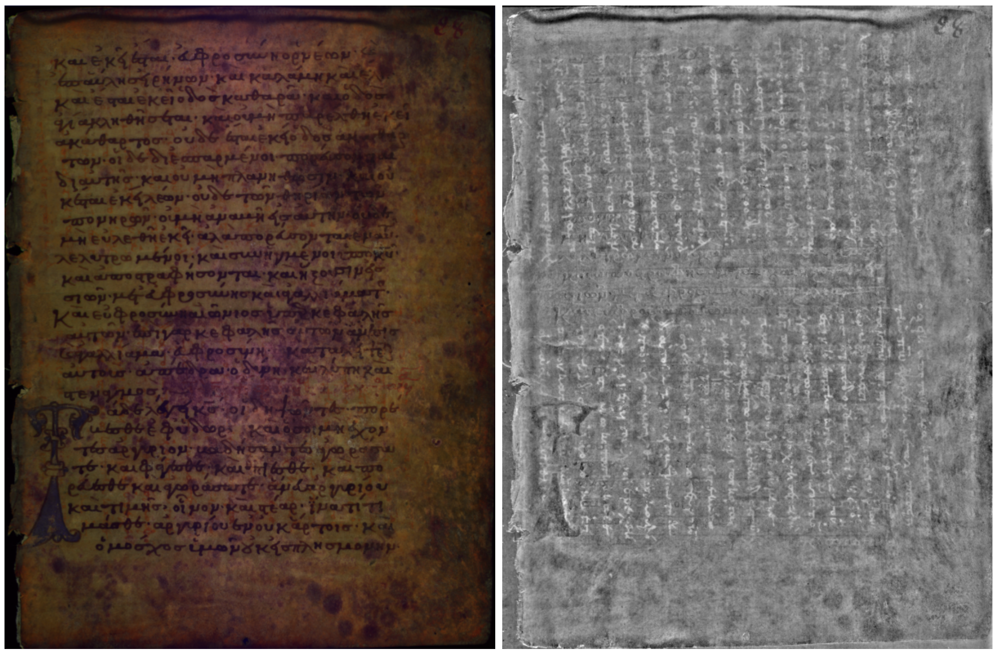


Figure 1.3: Leaf 28r from the Archimedes Palimpsest before (Left) and after (Right) least-squares spectral unmixing. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

undecipherable due to significant damage from mold, fire, and forged paintings. Examples of the degradations that the manuscript has endured over the past century are illustrated in the following set of figures. Figure 1.5 is a digitally stitched image of the original Heiberg photographs of the leaves 57v-64r captured in the early 1900's. Comparing this photograph to the images of the leaves in Figures 1.6–1.8 shows the extent of the recent damage which obscures the texts.

In April of 2004, a group of scientists gathered at a symposium to consider additional methods for recovering as much of the remaining difficult text as possible. The goal was not to replace multispectral imaging on the palimpsest, but to provide the scholars with additional imagery or tools to help in the transcription. Multiple avenues were considered, and two were eventually pursued.

X-ray fluorescence (XRF) imaging was the first imaging scheme to be employed as a result of the this symposium. Leaves containing the most obscured regions of

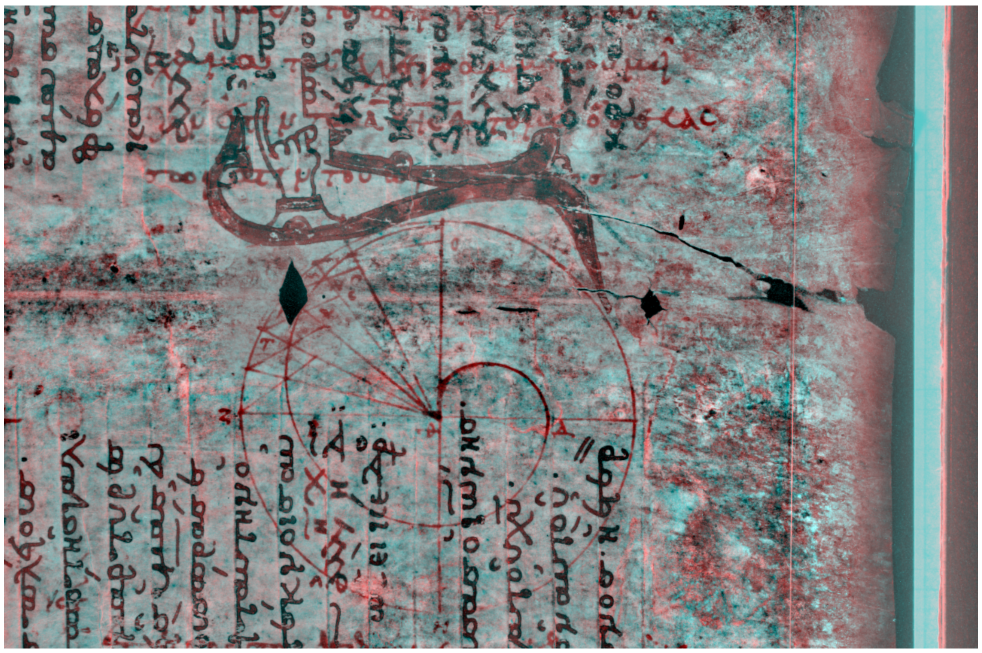


Figure 1.4: A section of leaves 98v-102r from the Archimedes Palimpsest after pseudocolor image processing. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

text (e.g., beneath forged paintings) were taken to the Stanford Linear Accelerator Center (SLAC) in California to be imaged using narrow beams of X-rays. The energy of the fluorescing X-rays characterize specific elements in the palimpsest, such as the iron in the two inks and the calcium within the parchment [40]. Only specific leaves have been imaged using the synchrotron radiation, as "beamtime" at SLAC is limited. The image-capture process is painstakingly slow (but improving), and only the most difficult leaves require this attention. Following the examples in Figures 1.5-1.8, "iron channel" and "calcium channel" XRF images of 057v are shown in Figure 1.9. Note that the overwriting, underwriting, and forged painting from 057r (the reverse side), are also apparent in these images.

The second selected area of research was the design of a character recognition system to assist scholars in transcribing the degraded Archimedes text. Several different methods for the task were considered due to the wide array of relevant recognition al-

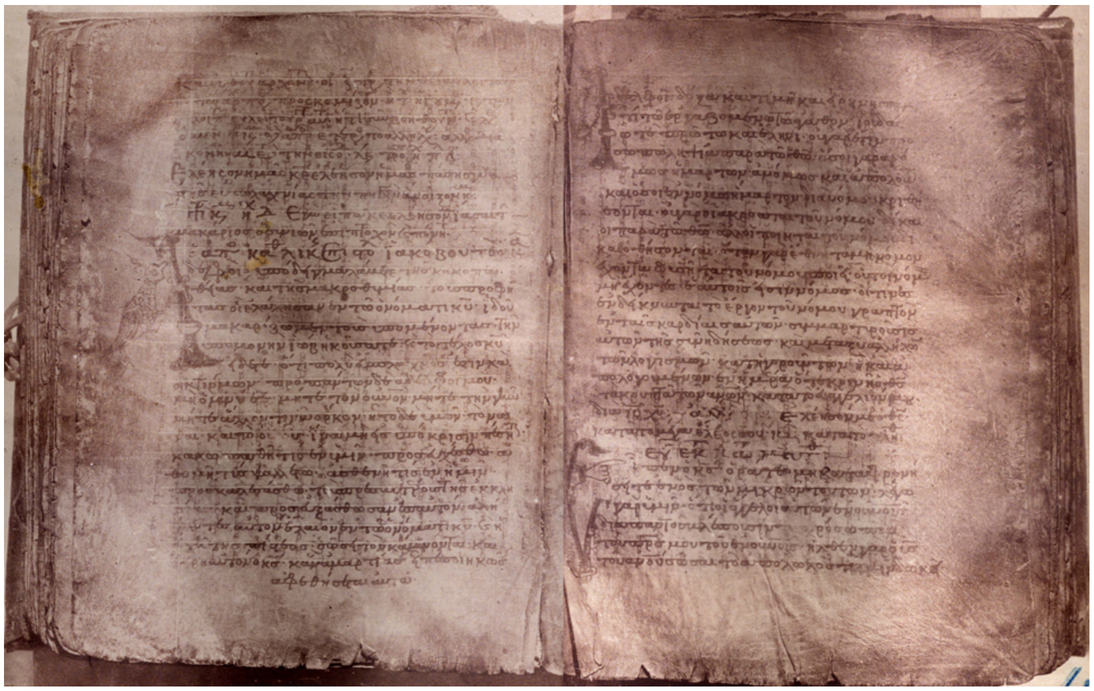


Figure 1.5: Original Heiberg photograph of leaves 57v-64r from the Archimedes Palimpsest. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

gorithms. The Archimedes Palimpsest Project held a competition among the scientists who advocated the design and use of such a tool, and submissions were evaluated in July of 2004. A correlation-based pattern recognition system designed by the author was awarded funding for further development. The preliminary system included a graphical-user-interface (GUI), developed in the IDL^{TM} programming language that provided classification results based solely upon the spatial characteristics of the images used in the correlation operations. Since its inception, the pattern recognition system has been updated over several iterations to accommodate new imagery, contextual knowledge from partial transcriptions, and is now being used in additional applications requiring improved flexibility. We follow the development of this system and examine each of its component algorithms throughout the remainder of this dissertation.

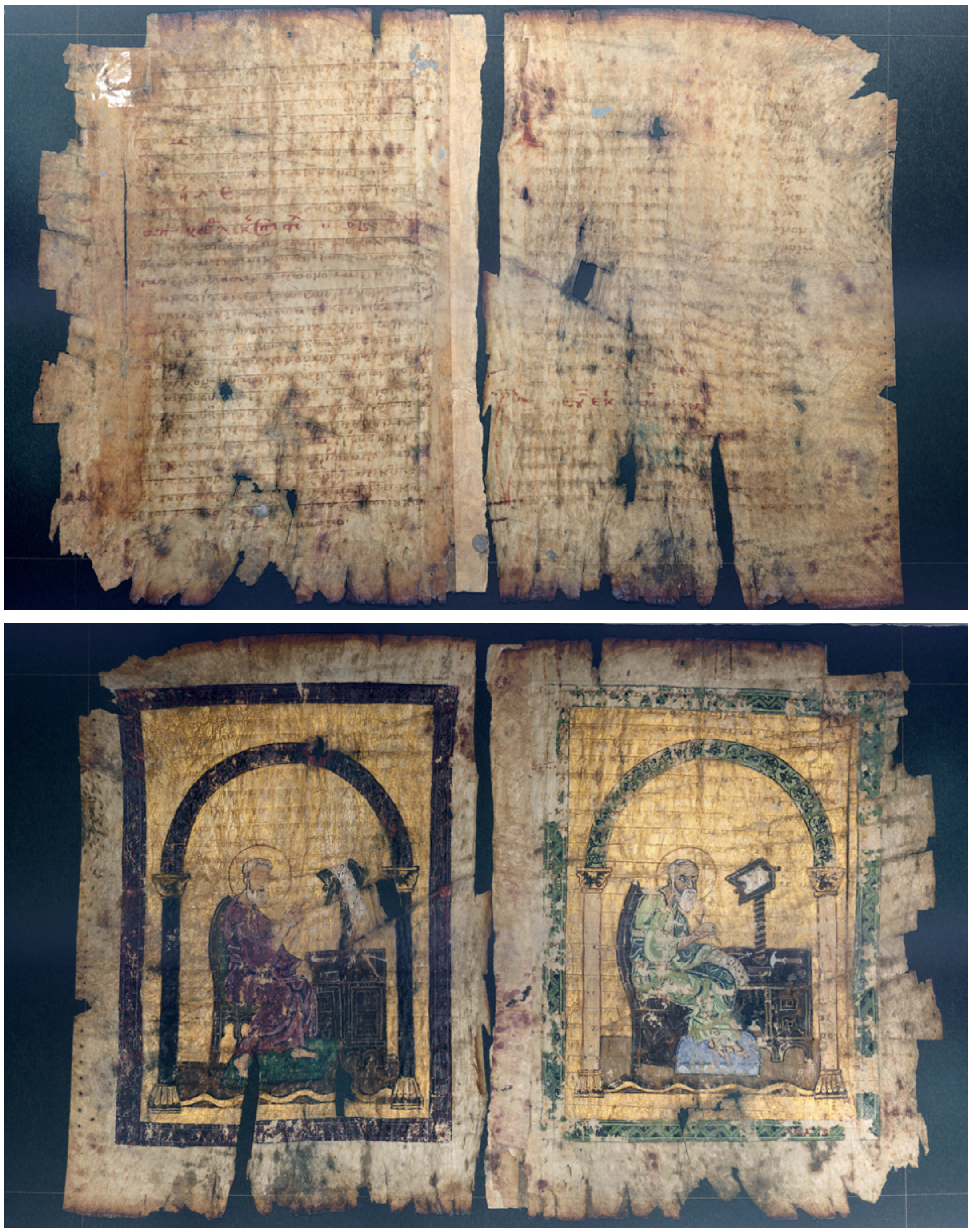


Figure 1.6: Leaves 57v-64r (Top) and reverse-side 64v-57r (Bottom) from the Archimedes Palimpsest images under strobe illumination. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

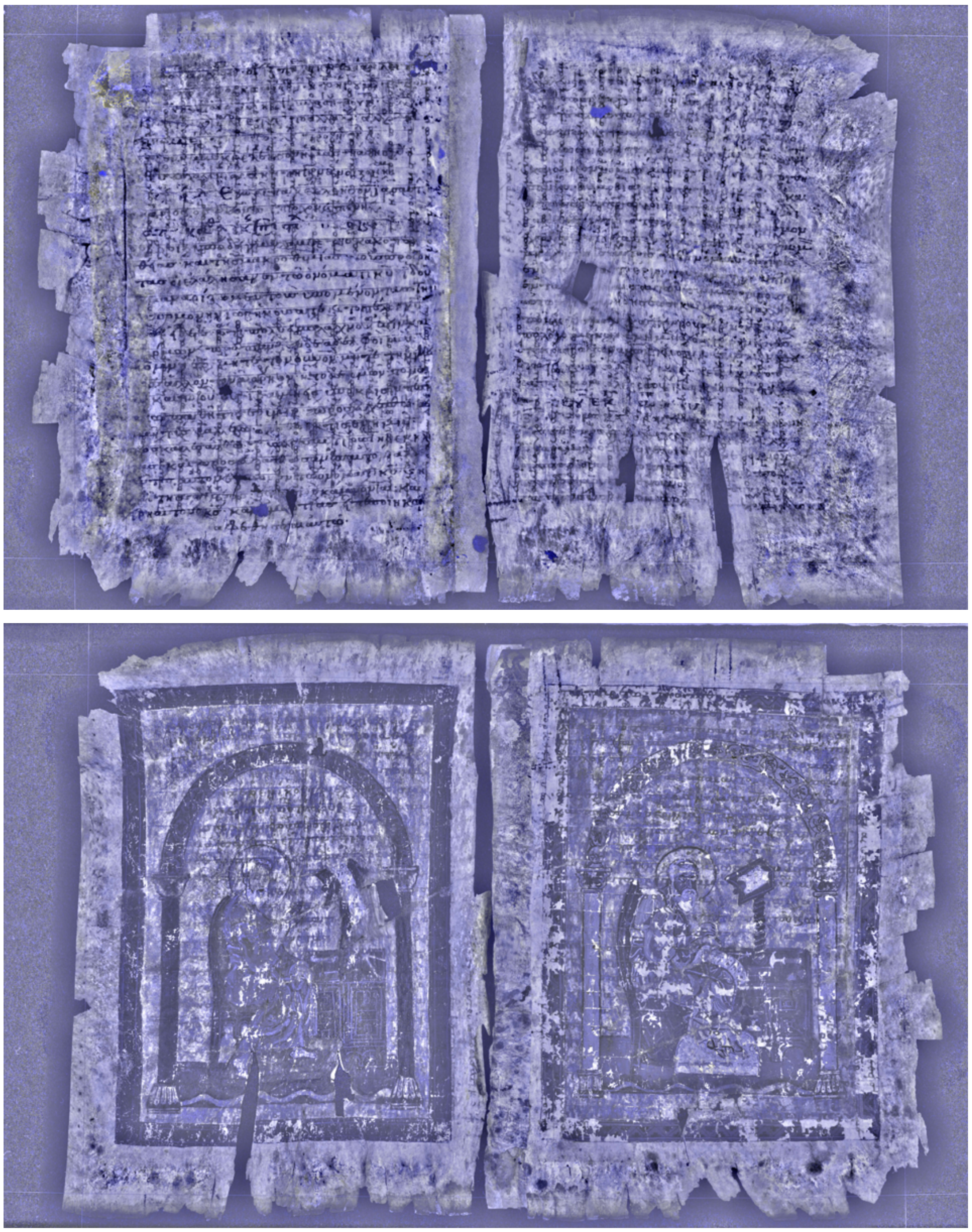


Figure 1.7: Leaves 57v-64r (Top) and reverse-side 64v-57r (Bottom) from the Archimedes Palimpsest images under ultraviolet illumination. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

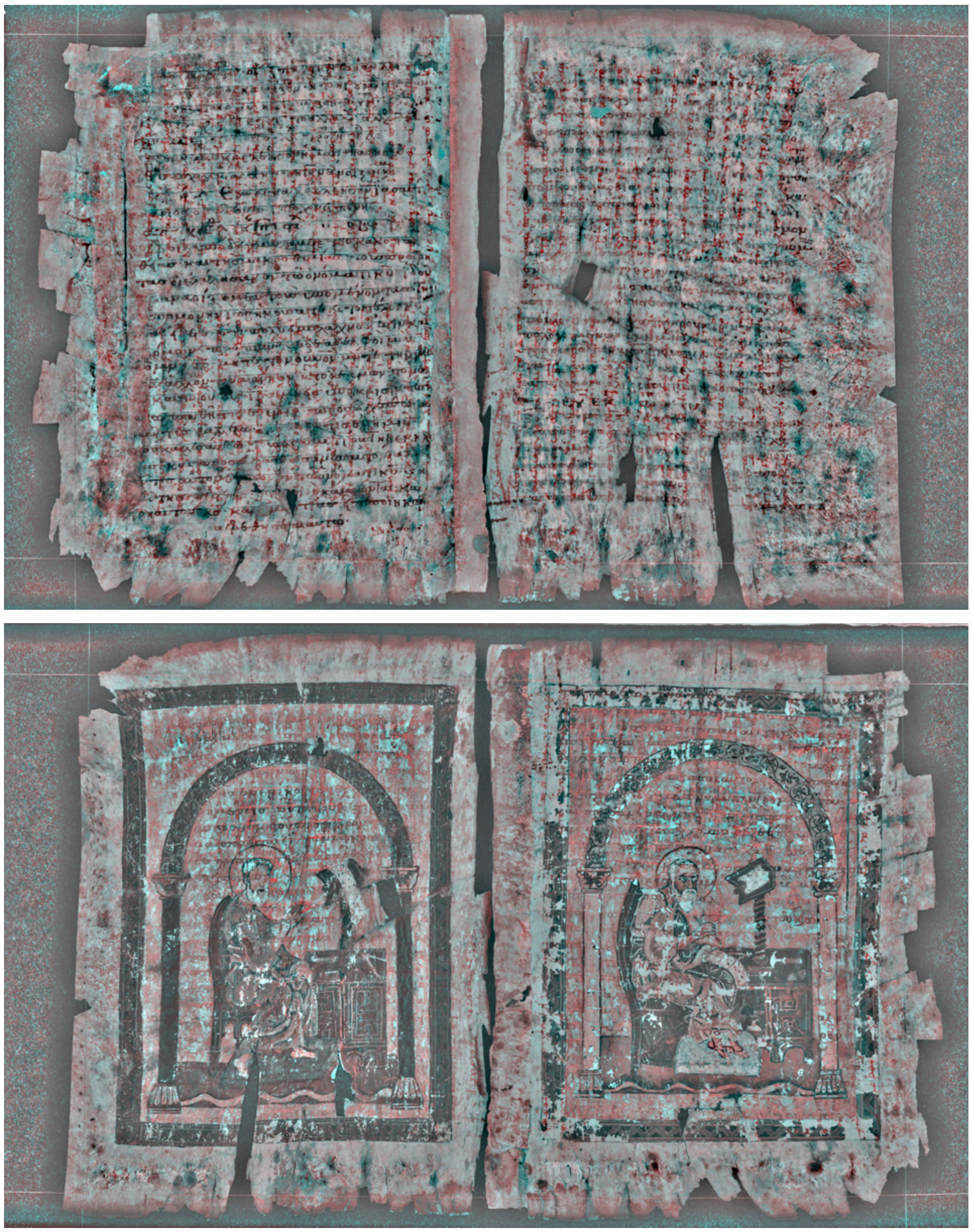


Figure 1.8: Leaves 57v-64r (Top) and reverse-side 64v-57r (Bottom) from the Archimedes Palimpsest images after pseudocolor image processing. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

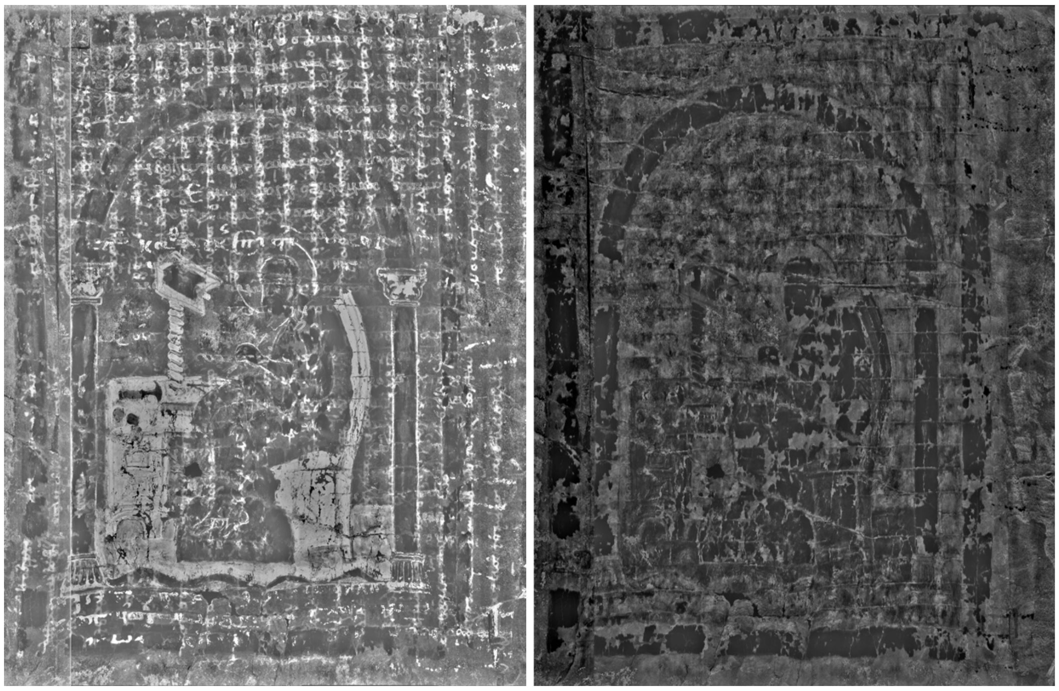


Figure 1.9: Leaf 57v from the Archimedes Palimpsest as the “iron channel” image (Left) and as the “calcium channel” image (Right) using XRF imaging. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

1.2 Dissertation Overview

This dissertation encompasses three segments of work (or is composed of three Acts), all of which interconnect with the Archimedes Palimpsest Project. The first section focuses on an extensive comparison of existing advanced correlation filter schemes in this application. A large portion of this dissertation is dedicated to providing the necessary background information used to develop these algorithms. The design of a correlation-based character recognition system for the scholars transcribing the Archimedes Palimpsest is the subject of the second section. One of the most appealing reasons for using correlation filtering as the primary spatial recognition method is its versatility over a wide range of applications; the ability of the system to accommodate additional application areas is demonstrated. The last portion of this work is an investigation aimed at improving the magnitude or phase representation in correlation filter designs using common geometric transforms. New filtering methods are incorporated into the working recognition system for analysis on real data. For purposes of both readability and clarity, each chapter presents the relevant content pertaining to each of these sections.

1.3 Organization

Chapter 2 provides a detailed list of objectives to be satisfied for successful completion of this dissertation. In Chapter 3, background theory is provided on the algorithms relevant to the design of the character recognition system for the Archimedes Palimpsest. Chapter 4 describes the approach used for the various studies performed on advanced correlation filtering and the methodology for developing the associated correlation pattern recognition system. Results pertaining to each of the objectives are then presented in Chapter 5, and Chapter 6 offers both concluding statements and suggestions for future work. An appendix is also included at the end of this dissertation to provide a brief review of the more common mathematical procedures used to derive several of the correlation algorithms.

–The more you know, the less sure you are.

Voltaire

2 Objectives

The underlying goal of this research has always been to implement algorithms and to develop tools to assist the scholars of ancient Greek mathematics in their transcription of the Archimedes Palimpsest. Thus, the majority of the objectives presented here were tailored to design a working pattern recognition system for handwritten character recognition on this specific manuscript. While certainly well-established, the field of correlation pattern recognition continues to evolve, largely due to increased attention from security and defense fields in recent years. One of the more attractive features of correlation filters is that they can be used in a variety of recognition tasks that offer adequate spatial structure for matching.

With the Archimedes Palimpsest Project providing the motivation, three categories of objectives were devised to determine the success of the work encompassed by this dissertation. The first is a study of both past and present correlation filtering methods. This includes investigation, implementation, and comparison of both classical and advanced correlation designs. The historical development of these algorithms is analyzed in Chapter 3 to reveal inherent limitations in correlation filter design and to facilitate improvement by unconventional methods (third objective category).

The second group of objectives is concerned with the design of the character recognition tool for the Archimedes Palimpsest. Evaluations of the correlation filters are used to create a pattern recognition system with a graphical-user-interface (GUI) for the scholars transcribing the degraded text of Archimedes. The preliminary system should be capable of implementing the desired filtering schemes to provide character classification results. Accuracy of these results can then be improved by integrating a probabilistic network into the system to take advantage of contextual information.

The final category focuses on the development of improved phase (or magnitude) representations in correlation filter design. An investigation of the "Corefaces" approach, by Savvides, et al., [43] is performed to determine the advantages of using a linear subspace for phase representation. The final goal of this dissertation is to use this method as a template for building more useful spatial correlation algorithms and then to incorporate any improvements into the pattern recognition system.

A detailed list of the main objectives for the work that constitutes this dissertation is presented below for reference:

1. Survey and evaluate the more common correlation filtering schemes
    (a) Implement the filtering algorithms in a suitable programming language
    (b) Survey and implement accepted correlation performance metrics
    (c) Compare the results of filtering for both synthetic and real imagery
2. Archimedes character recognition tool design
    (a) Develop a correlation-based character recognition system with a user-friendly GUI for use by scholars who are transcribing the Archimedes Palimpsest
    (b) Demonstrate that the system can be used in other applications
    (c) Improve recognition performance on the Archimedes Palimpsest by integrating a probabilistic network into the pattern recognition system to exploit the existing partial transcription
3. Improved phase (or magnitude) representation in correlation filtering
    (a) Implement and analyze an eigenspace approach to phase representation
    (b) Develop a method to improve the phase (or magnitude) representation in correlation filters using common geometric transformations
    (c) Incorporate improvements into the pattern recognition system designed for the Archimedes Palimpsest

The approach used to satisfy the objectives in each of these three categories is presented in Chapter 4, and the results of this work are reported in Chapter 5.

Before proceeding with further discussion of this work, it may be helpful to review some of the background theory provided in the following chapter. Advanced correlation filter design, geometric transforms for image processing, simple probabilistic networks, and other relevant topics are examined in detail.

–One of the most interesting aspects of the world is that it can be considered to be made up of patterns.
A pattern is essentially an arrangement. It is characterized by the order of elements of which it is made rather than by the intrinsic nature of these elements.

Norbert Wiener

–The deep study of nature is the most fruitful source of mathematical discoveries.

Jean-Baptist-Joseph Fourier

3 Theory

In its broadest context, pattern recognition refers to the task of assigning input data to one of multiple predefined classes. The input for the recognition problem can be N-dimensional. For example, the goal may be to appropriately classify a voice from a 1-dimensional audio signal, identify a fingerprint from a 2-dimensional grayscale image, or determine the constituents of a gaseous plume from a 100-band hyperspectral signature. Due in large part to the nature of the handwritten character recognition application described in Chapter 1, this dissertation focuses almost exclusively on extracting features from the spatial structure of target objects in 2-dimensional images. Pattern recognition of this flavor is commonly referred to as *spatial pattern recognition*.

Spatial pattern recognition can be subdivided into a series of steps; a simplified model of the process consists of just three fundamental steps: *preprocessing*, *feature extraction*, and *classification*. Figure 3.1 shows a block diagram of the serially-connected subprocesses that constitute a more complete recognition task. After the input has been acquired by image sensing, which also is a major step in the recognition chain, the image is preprocessed to maximize the efficiency of the feature extraction algo-

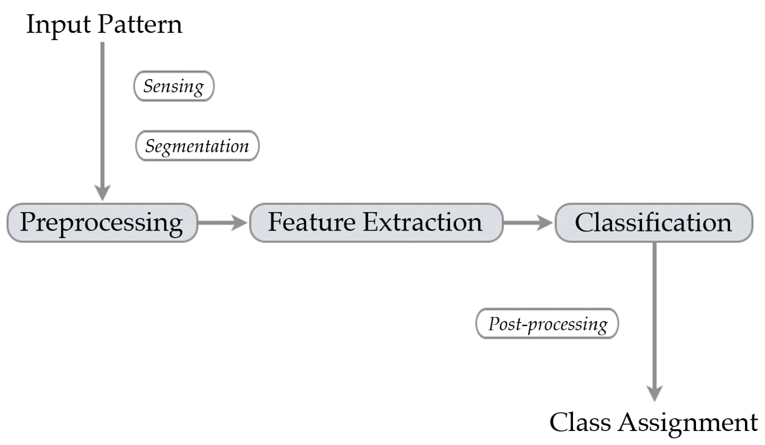


Figure 3.1: Block diagram of the major steps of the pattern recognition chain.

rithms for the desired application. A significant amount of research has been performed to improve segmentation techniques for spatial recognition tasks; this step is omitted from further discussion however, as we will assume segmentation is either incorporated into the feature extractor (via correlation filtering) or performed manually by the user (region-of-interest selection). Common application-specific preprocessing steps include (but are not limited to) reducing noise, spatial registration, edge detection or enhancement, and improving the dynamic range of the input [31]. Some of these preprocessing techniques will be revisited later in Chapter 4.

A main area of interest for this work is to develop accurate and unique descriptions for handwritten characters, the spatial patterns of interest, in digital image data. Thus, feature extraction methods will be an underlying focus of this work, while classification and post-processing techniques will receive less attention. Ideally, *features* (also called *pattern descriptors*) are needed that provide enough useful information to correctly classify the input pattern, or patterns. A *feature vector* $\mathbf{x}$ is composed of d

pattern descriptors x_i, and can be represented as a column vector:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \tag{3.1}$$

The feature vector $\mathbf{x}$ lies in a d-dimensional Euclidean space $\mathbf{R}^d$, referred to as the *feature space*. In this application, we will be concerned with using pixel gray values as the features x_i for correlation filtering (which will be discussed in detail throughout the remainder of this chapter). Moreover, feature vectors can be generated by subsequently extracting features from the correlation results. Once feature extraction is complete, it is the job of the classifier, or classifiers, to determine which class the feature vector is most associated.

Intuition suggests that increasing the number of descriptors in each feature vector improves the accuracy of classifying the pattern. However, the pattern classification task becomes increasingly complex and more computationally expensive as the number of features and number of classes are increased.

As is the case with many pattern recognition problems, class membership should not change as the pattern is distorted. In these situations, it is ideal to apply a feature extraction process, or to design an advanced classifier, that is invariant to the possible pattern transformations. Each feature vector $\mathbf{x}$ is a point in feature space $\mathbf{R}^d$, and to be correctly classified, each of these points must map to its corresponding class. Thus, an optimal system for recognizing handwritten characters would correctly classify the feature vectors regardless of background noise in the input image data, scaling and/or rotation of the pattern of interest, and any other "degradations" (Figure 3.2). This concept of distortion-tolerant feature extraction is essential to consider as we proceed; a quick glance at the images in Chapter 1 confirms this statement.

3.1 Correlation Pattern Recognition

A subcategory of the pattern recognition family exists in which the process of classifying an input pattern is based on a correlation-filtered output. The function of

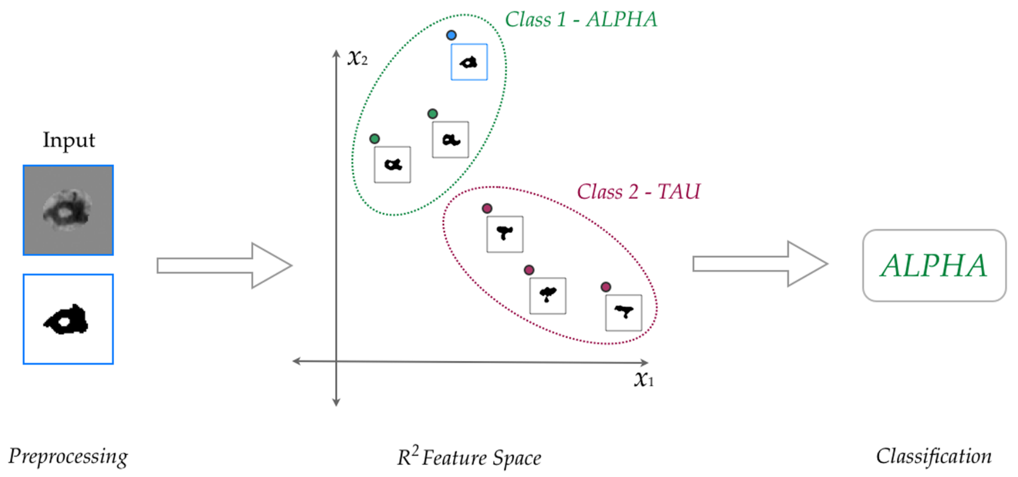


Figure 3.2: Example pattern recognition system that provides correct character classification in the presence of small handwriting distortions.

correlation filtering in pattern recognition is very intuitive; it should determine the degree of similarity between the input pattern and the *training* patterns assigned to known classes. While many fragments of this field have existed for decades, a useful resource by Kumar, et al., has recently been published that presents a large ensemble of various correlation filtering schemes, and designates the subject matter by the name "correlation pattern recognition" (CPR) [31].

Correlation filtering algorithms have been used as a means for feature extraction and classification in a variety of applications. Advanced correlation design is currently used in biometric verification to classify images of faces, eyes, and fingerprints [32], in defense and security problems to classify synthetic aperture radar (SAR) images of vehicles [45], and in document processing systems to detect and locate low-contrast character strings from coupons [33]. The flexibility of correlation as a robust pattern recognition tool is demonstrated for multiple applications in this body of work.

3.2 Correlation Filtering

Correlation filtering algorithms share a wide range of complexity in terms of development and synthesis. However, the ease of implementation is one of the most attractive features of spatial correlation; it is often as trivial as a multiplication of the input spectrum by the filter transfer function in the frequency domain. Perhaps the most useful property of correlation filtering is *shift invariance*. A shift-invariant operation ϑ that acts on an input function $f(x)$ and produces an output function $g(x)$, will produce a shift in the position of output function equal to the shift in position of the input. That is,

$$\vartheta\{f(x)\} = g(x) \tag{3.2a}$$

$$\vartheta\{f(x-x_0)\} = g(x-x_0) \tag{3.2b}$$

where x_0 is a real constant. Gaskill summarized the property well, noting "shift invariance implies that the *behavior of the system* is not a function of the independent variable" [14]. In the context of pattern recognition, shift invariance implies that use of correlation filters for feature extraction (or classification) alleviates the need for a segmentation algorithm. This section provides a mathematical foundation for several correlation filtering processes and introduces the notation to be used in a considerable portion of this dissertation.

3.2.1 Classical Spatial Matched Filters

We begin by introducing the "classical" spatial matched filters, i.e., correlation filters derived from a single training signal (the reference). The simplest and perhaps the oldest correlation filter, the "ideal" matched filter, is examined first, and we then follow the historical progression toward the development of more robust correlation schemes. While many classical matched filters may prove impractical in complex applications, the underlying concepts serve as a foundation for correlation filter development. Continuous Cartesian coordinates are used throughout this section for simplicity, though the extension to digital implementation is straightforward.

3.2.1.1 The Ideal Matched Filter

To develop a formulation for the *ideal matched filter*, let us first examine an imaging scenario under "ideal" conditions. Assume that we are given an *input* image $g(x,y)$ that contains only a spatially-translated version of a "known" *reference* image $f(x,y)$, which contains the object (or pattern) of interest (Eq. 3.3).

$$g(x,y) = f(x - x_0, y - y_0) \tag{3.3}$$

A matched filter actuates an "alarm" of some sort wherever the known reference $f(x,y)$ is detected in the input image $g(x,y)$. Applying a matched filter impulse response $h(x,y)$ to $g(x,y)$ should provide a result that specifies an estimate of the reference location (x_0, y_0) in the input. This spatial domain output is typically referred to as the *correlation plane*. The ideal correlation peak would be a Dirac delta function in the continuous case, or an approximate Dirac delta function for digital systems, at the location of the reference:

$$g(x,y) * h(x,y) = \delta(x - x_0, y - y_0) \tag{3.4}$$

where the asterisk in Eq. 3.4 denotes the 2-dimensional convolution operation.

In the spatial frequency domain, the input image $g(x,y)$ is expressed as

$$G(\xi,\eta) = F(\xi,\eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]} \tag{3.5}$$

where $i \equiv \sqrt{-1}$. In words, the input spectrum $G(\xi,\eta)$ is the product of the reference spectrum $F(\xi,\eta)$ and the linear phase introduced by the spatial translation. Multiplying $G(\xi,\eta)$ by the matched filter transfer function $H(\xi,\eta)$ then should yield a constant-magnitude linear phase-term that, after application of the 2-dimensional inverse Fourier transform, produces a Dirac delta function at the coordinates (x_0, y_0).

$$G(\xi,\eta) \cdot H(\xi,\eta) = \left[F(\xi,\eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]}\right] \cdot H(\xi,\eta) \tag{3.6a}$$

$$G(\xi,\eta) \cdot H(\xi,\eta) = e^{-2\pi i[\xi x_0 + \eta y_0]} \tag{3.6b}$$

Thus, from Eq. 3.6a and Eq. 3.6b, the frequency domain representation of the ideal matched filter is:

$$\boxed{H(\xi,\eta) = \frac{1}{F(\xi,\eta)} = \frac{1}{|F(\xi,\eta)| \cdot e^{i\Phi_F}} = \frac{e^{-i\Phi_F}}{|F(\xi,\eta)|}} \tag{3.7}$$

where Φ_F denotes the phase of $F(\xi,\eta)$. The ideal match filter in Eq. 3.7 is completely analogous to the simple *inverse filter* used in deconvolution tasks, which seeks compensation for degradations induced by the imaging system, typically characterized by the system point spread function (PSF). Unfortunately, both the ideal matched filter and the inverse filter neglect to account for an additive noise component.

We now assume that the input $g(x,y)$ contains a translated replica of the reference in and an additive noise component $n(x,y)$:

$$g(x,y) = f(x - x_0, y - y_0) + n(x,y) \tag{3.8}$$

Its frequency domain representation now contains an additive noise spectrum $N(\xi,\eta)$:

$$G(\xi,\eta) = F(\xi,\eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]} + N(\xi,\eta) \tag{3.9}$$

To obtain the ideal correlation output, multiplying the Fourier transform of the input image by the matched filter transfer function should again result in only a constant-magnitude linear-phase term:

$$\begin{aligned} G(\xi,\eta) \cdot H(\xi,\eta) &= \Big[F(\xi,\eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]} + N(\xi,\eta)\Big] \cdot H(\xi,\eta) && \text{(3.10a)} \\ G(\xi,\eta) \cdot M(\xi,\eta) &= e^{-2\pi i[\xi x_0 + \eta y_0]} && \text{(3.10b)} \end{aligned}$$

Eq. 3.10a implies that two conditions must be satisfied at all spatial frequencies to produce the output in Eq. 3.10b:

$$F(\xi,\eta) \cdot H(\xi,\eta) = 1 \tag{3.11a}$$

$$N(\xi,\eta) \cdot H(\xi,\eta) = 0 \tag{3.11b}$$

Eq. 3.11a clearly illustrates that the Fourier transform of the reference $F(\xi,\eta)$ must be nonzero (at all spatial frequencies). The second criterion is evident by comparing Eq. 3.11a and Eq. 3.11b; a Dirac delta function correlation peak can only be achieved when the noise spectrum $N(\xi,\eta)$ is zero at all spatial frequencies. In this more realistic imaging scenario, the additive noise is often amplified by the ideal matched filter and spread throughout the spatial domain via the inverse Fourier transform. This is because $F(\xi,\eta)$ is typically dominated by low frequencies in "real" images, which in turn, cause its reciprocal to act as a high-frequency boosting filter. This amplification of noise prohibits the use of the ideal matched filter in practical imaging problems and instead limits it to a condition for constructing more useful matching techniques.

The following example illustrates the effects of small image distortions, such as noise and in-plane rotation, on the correlation peaks produced using the ideal matched filter. Figure 3.3 shows both the spatial and frequency domain representations of an original reference and three different targets. The 8-bit, 128×128 pixel images include a background with a gray level of 95 and a character with a gray level of 160. Each target contains a spatially translated replica of the reference: the first has no additional "distortions", the second contains additive white Gaussian noise (WGN), and the third introduces a 5° clockwise in-plane rotation of the reference object. These images will be used in future examples to illustrate the general behavior of other correlation filters, and for qualitative comparison and discussion.

Figure 3.4 shows the normalized correlation planes produced for each test target when using the reference to generate the ideal matched filter. As expected and in agreement with Eq. 3.4, the ideal matched filter produces an approximate Dirac delta function at the reference location of the first target. The addition of noise and/or small rotation of the reference causes the filter to amplify high frequencies, thus "burying" the correlation peak in noise.

3.2.1.2 Phase-Only Matched Filters

As indicated by its name, a *phase-only matched filter* (POMF) eliminates the magnitude term of the ideal matched filter (Eq. 3.7) to avoid the noise amplification that typically results at high spatial frequencies. The simplest form of the POMF was introduced by Horner and Gianino [20] in the early 1980's:

$$\boxed{M(\xi,\eta) = e^{-i\Phi_F}} \tag{3.12}$$

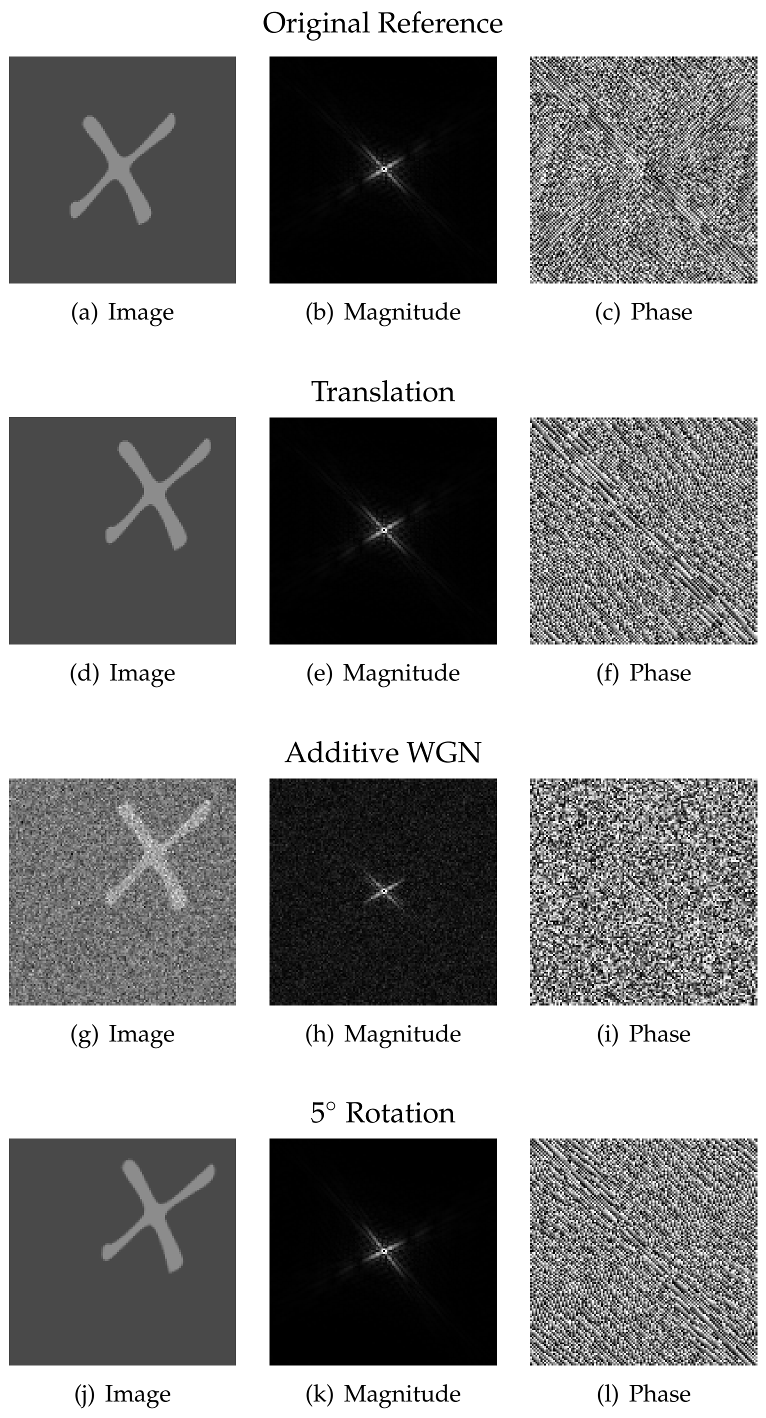
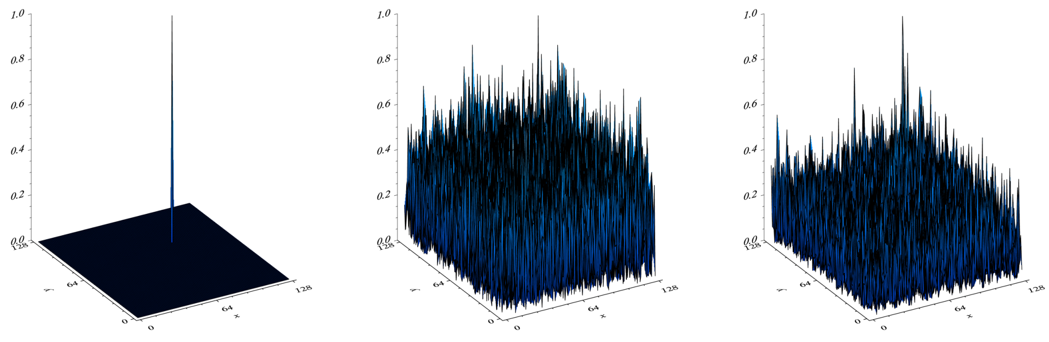


Figure 3.3: Example reference (a-c) and test targets (d-l) as images and as magnitude and phase representations.



(a) Translation (b) Additive WGN (c) 5° Rotation

Figure 3.4: Ideal matched filter example – normalized correlation planes for the ideal matched filter using the reference and test images in Figure 3.3.

This filter relies solely on "phase canceling" to form correlation peaks, i.e., the phase components introduced by any reference patterns in the input should be removed by the POMF, leaving only the linear phase term due to translation to specify target locations. This concept is more clearly illustrated by using the imaging scenario from the previous section. Assume that we are given an input image $g(x, y)$ that contains the known reference pattern $f(x, y)$ at its origin $(0, 0)$ and at the coordinates (x_0, y_0) as shown in Eq. 3.13.

$$g(x, y) = f(x, y) + f(x - x_0, y - y_0) \tag{3.13}$$

The Fourier transform of the input image $G(\xi, \eta)$ is:

$$G(\xi, \eta) = F(\xi, \eta) + F(\xi, \eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]} \tag{3.14}$$

Multiplying the POMF in Eq. 3.12 by $G(\xi, \eta)$ yields the following frequency-domain output:

$$G(\xi, \eta) \cdot M(\xi, \eta) = \left[F(\xi, \eta) + F(\xi, \eta) \cdot e^{-2\pi i[\xi x_0 + \eta y_0]}\right] \cdot e^{-i\Phi_F} \tag{3.15a}$$

$$= |F(\xi, \eta)| + |F(\xi, \eta)| \cdot e^{-2\pi i[\xi x_0 + \eta y_0]} \tag{3.15b}$$

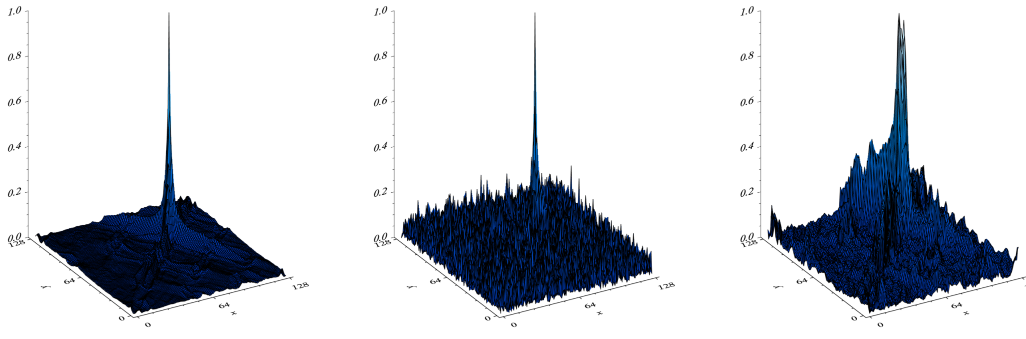
Applying the inverse Fourier transform, denoted by $\mathcal{F}^{-1}\{\cdot\}$, to the result in Eq. 3.15b produces a correlation plane $c(x,y)$ in the spatial domain.

$$\begin{aligned} c(x,y) &= \mathcal{F}^{-1}\{G(\xi,\eta)\cdot M(\xi,\eta)\} && (3.16a)\\ &= \mathcal{F}^{-1}\{|F(\xi,\eta)|\cdot 1(\xi,\eta)\} + \mathcal{F}^{-1}\{|F(\xi,\eta)|\cdot e^{-2\pi i[\xi x_0+\eta y_0]}\} && (3.16b)\\ &= \mathcal{F}^{-1}\{|F(\xi,\eta)|\} * [\delta(x,y)+\delta(x-x_0,y-y_0)] && (3.16c) \end{aligned}$$

Eq. 3.16c shows that the POMF indicates a match of the reference pattern $f(x,y)$ by replicating the inverse Fourier transform of the reference magnitude spectrum $\mathcal{F}^{-1}\{|F(\xi,\eta)|\}$ at the spatial coordinates $(0,0)$ and (x_0,y_0). It is important to recognize that this example demonstrates the usefulness of one of the most attractive properties of correlation filters: shift invariance. Translation of the reference pattern in the input results in the formation of the POMF's characteristic correlation peak, specified by $\mathcal{F}^{-1}\{|F(\xi,\eta)|\}$, translated by the same amount (x_0,y_0). Also, the effect of eliminating the magnitude term of the ideal filter is now obvious. The correlation peaks in $c(x,y)$ now have some finite support that is inversely proportional to the bandwidth of the reference magnitude spectrum. For example, if $|F(\xi,\eta)|$ has a wide support and is dominated by low frequencies, which is often true, the correlation peaks will be approximately shaped like Gaussian functions with small support. Lastly, it should be noted that any additive noise present in the input image $g(x,y)$ will produce an unwanted additive term $n(x,y) * \mathcal{F}^{-1}\{e^{-i\Phi_F}\}$ in the output that could potentially disrupt the ability to discern between correlation peaks and background noise.

Figure 3.5 shows examples of correlation peaks produced for each of the targets in Figure 3.3 using the reference to apply the POMF algorithm. Some important observations should be apparent upon comparison to the ideal matched filter outputs in Figure 3.4. The finite support of the POMF correlation peaks can be seen in the absence of distortion. More importantly, the noise floor (for the target with additive WGN) has not been amplified, as the POMF does not boost or attenuate specific frequencies.

Additional variations of the POMF have been developed to retain the useful spatial location information preserved in the phase, while reducing the effects of the peak-widening magnitude term $\mathcal{F}^{-1}\{|F(\xi,\eta)|\}$. The *symmetric phase-only matched filter* (SPOMF), which has seen certain success in image recognition and registration tasks [5], correlates only the phase information that is present in the input Φ_G and the refer-



(a) Translation (b) Additive WGN (c) 5° Rotation

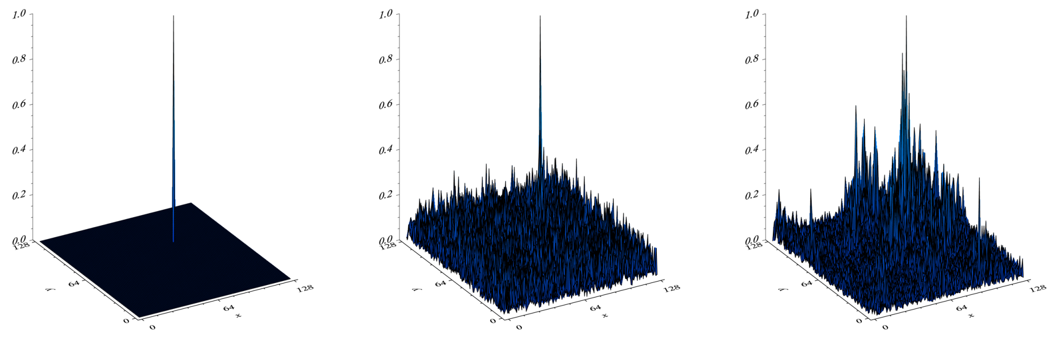
Figure 3.5: Phase-only matched filter example – normalized correlation planes for the POMF using the reference and test images in Figure 3.3.

ence Φ_F:

$$\boxed{C(\xi,\eta) = \frac{G(\xi,\eta)}{|G(\xi,\eta)|} \cdot \frac{F^*(\xi,\eta)}{|F^*(\xi,\eta)|} = e^{i[\Phi_G - \Phi_F]}} \tag{3.17}$$

Thus, $|C(\xi,\eta)| = 1$ at all spatial frequencies after the phases are extracted. In the absence of noise, the SPOMF will produce Dirac delta functions at all spatial locations containing copies of the reference pattern. Additive noise in the input will still produce an additive term $\mathcal{F}^{-1}\{e^{i\Phi_N}\} * \mathcal{F}^{-1}\{e^{-i\Phi_F}\}$.

Upon inspection, the SPOMF correlation peaks in Figure 3.6 exhibit more attractive characteristics than those produced by the ideal matched filter (Figure 3.4) and the POMF (Figure 3.5). An approximate Dirac delta function is produced at the location of the translated reference when the target contains no noise or other distortions. Amplification of additive noise is no longer an issue, as the SPOMF only affects the phase. One caveat is that the flat correlation spectrum inherent in the SPOMF algorithm allows noise at higher frequencies to pass into the spatial domain, potentially degrading the corresponding peak. The following section considers an algorithm that typically attenuates these high frequencies.



(a) Translation (b) Additive WGN (c) 5° Rotation

Figure 3.6: Symmetric phase-only matched filter example – normalized correlation planes for the SPOMF using the reference and test images in Figure 3.3.

3.2.1.3 The Matched Spatial Filter

Sharp correlation peaks with large amplitudes are desired and often necessary to discriminate multiple reference patterns that are small spatial distances apart in the input. We have shown that the POMF in Eq. 3.12 described in the previous section offers additional tolerance to noise relative to the ideal matched filter in Eq. 3.7 at the cost of reducing the sharpness of the correlation peak. In some imaging applications, it may be imperative to push this tradeoff further to provide as much noise tolerance as possible while still providing adequate peaks. Perhaps it is known that only one instance of the reference pattern is present in the input, or that useful correlation peaks will only be obtained by providing some control over input noise.

To develop a filter that meets this criterion, we again refer to the "problematic" reciprocal-magnitude term of the ideal matched filter. By observing the behavior of this filter as a spatial domain impulse response, the contribution to noise sensitivity can be additionally reduced.

$$\begin{aligned} h(x,y) &= \mathcal{F}^{-1}\left\{\frac{e^{-i\Phi_F}}{|F(\xi,\eta)|}\right\} = \mathcal{F}^{-1}\left\{\frac{F^*(\xi,\eta)}{|F(\xi,\eta)|^2}\right\} && (3.18a) \\ &= \mathcal{F}^{-1}\left\{F^*(\xi,\eta)\right\} * \mathcal{F}^{-1}\left\{\frac{1}{|F(\xi,\eta)|^2}\right\} && (3.18b) \\ &= f^*(-x,-y) * \mathcal{F}^{-1}\left\{\frac{1}{|F(\xi,\eta)|^2}\right\} && (3.18c) \end{aligned}$$

Eliminating the second term in Eq. 3.18c reduces the amount of wideband noise amplification after applying the inverse Fourier transform. The resulting filter is:

$$h(x,y) = f^*(-x,-y) \tag{3.19}$$

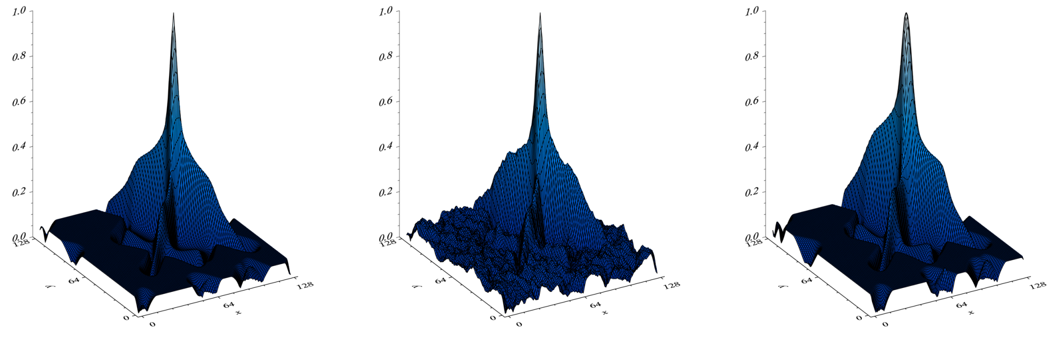
which is the complex conjugate of a reversed (or rotated) replica of the reference. This well-known filter is commonly referred to as the *matched spatial filter* (MSF), and its application produces the *crosscorrelation* of $g(x,y)$ and $f(x,y)$. The corresponding frequency-domain representation of the MSF is:

$$\boxed{H(\xi,\eta) = F^*(\xi,\eta) = |F(\xi,\eta)| \cdot e^{-i\Phi_F}} \tag{3.20}$$

If we again assume that the magnitude spectrum is dominated by components with low frequencies, the magnitude term $|F(\xi,\eta)|$ acts as a lowpass filter.

The major advantage of the MSF is its inherent maximization of the signal-to-noise power ratio at the reference image location(s) (x_0,y_0) in the correlation plane [4]. However, as expected, the peak sharpness has been reduced dramatically. Recall that the shape of the peak in the correlation output produced by the POMF was determined by $\mathcal{F}^{-1}\{|F(\xi,\eta)|\}$. By comparison of Eq. 3.12 and Eq. 3.20, it is evident that the correlation peak produced by the MSF is inversely proportional to the bandwidth of $\mathcal{F}^{-1}\{|F(\xi,\eta)|^2\}$.

In addition, the central ordinate theorem verifies that in the absence of noise, the maximum amplitude of the MSF peak is the area of the power spectrum of the reference. Changes in the amplitude of $f(x,y)$ and $g(x,y)$ directly influence the height of the floor in the correlation plane $c(x,y)$. Thus, a correlation peak produced by the MSF at a reference pattern location (x_0,y_0) may have a smaller amplitude than other loca-



(a) Translation (b) Additive WGN (c) 5° Rotation

Figure 3.7: Matched spatial filter example – normalized correlation planes for the MSF using the reference and test images in Figure 3.3.

tions in the correlation plane where $g(x, y)$ contained large amplitudes (bright spots).

The peaks in Figure 3.7 exhibit the wide support that is characteristic of the result after applying the MSF to the input. Both additive WGN and a small rotation of the reference pattern were distortions capable of degrading the correlation planes produced by the ideal matched filter (Figure 3.4) and POMF (Figure 3.5). Qualitatively, the three results obtained using the MSF share similar a structure. The reference pattern in Figure 3.3 is low-frequency dominated, which in turn causes the MSF to act as a lowpass filter during the matching process. Noise and edge effects at high frequencies are attenuated, and the support of the input spectrum is decreased, which produces broad correlation peaks.

A normalized version of the crosscorrelation exists, *correlation coefficient*, that accommodates changes in image amplitude [16]. The crosscorrelation between an input image $g(x, y)$ and a reference image $f(x, y)$ is:

$$g(x,y) * f^*(-x,-y) = \int\int_{-\infty}^{+\infty} g(\alpha,\beta) f^*(\alpha - x, \beta - y)\, d\alpha d\beta \tag{3.21}$$

Normalizing each image yields an expression for the spatial-domain implementation of the correlation coefficient:

$$\boxed{c(x,y) = \frac{\int\int \left[g(\alpha,\beta) - \bar{g}_{x,y}\right]\left[f(\alpha - x, \beta - y) - \bar{f}\right] d\alpha\, d\beta}{\sqrt{\int\int \left[g(\alpha,\beta) - \bar{g}_{x,y}\right]^2 d\alpha\, d\beta \int\int \left[f(\alpha - x, \beta - y) - \bar{f}\right]^2 d\alpha\, d\beta}}} \tag{3.22}$$

It is apparent from Eq. 3.22 that the correlation coefficient is bounded between negative one and positive one, achieving a maximum at locations in the input that are identical to the reference pattern. This method of correlation is very useful for applications containing exact replicas of the known reference pattern located far enough apart in the input to be detected despite partially overlapping peaks.

3.2.1.4 The Complement Matched Filter

The simple correlation filters discussed thus far have been presented in the chosen order to emphasize the following relationship: *the ideal matched filter, POMF, and MSF contain identical phase terms and differ only in the exponent of the magnitude term.*

$$H_{ideal}(\xi,\eta) = |F[\xi,\eta]|^{-1} \cdot e^{-i\Phi_F} \tag{3.23a}$$

$$H_{pomf}(\xi,\eta) = |F[\xi,\eta]|^{0} \cdot e^{-i\Phi_F} \tag{3.23b}$$

$$H_{msf}(\xi,\eta) = |F[\xi,\eta]|^{+1} \cdot e^{-i\Phi_F} \tag{3.23c}$$

This observation is essential for the development of future correlation filters, as it indicates that the tradeoff between peak sharpness and noise tolerance is controlled entirely by the magnitude component of the filter transfer function.

The so-called *complement matched filter* [26][50] is derived from a truncated Taylor-series expansion to approximate the magnitude term of the ideal matched filter and to provide a tunable response along this tradeoff continuum between the filters in Eq. 3.23a–3.23c.

The magnitude of the Fourier transform of the reference can be written as a sum of terms:

$$|F(\xi,\eta)| = |F(\xi,\eta)| - |F|_{max} + |F|_{max} \tag{3.24}$$

where $|F|_{max}$ is the maximum value of $|F(\xi,\eta)|$. It follows that $F(\xi,\eta)$ can be written as

$$\begin{aligned} F(\xi,\eta) &= |F(\xi,\eta)| \cdot e^{i\Phi_F} && (3.25a) \\ &= [|F(\xi,\eta)| - |F|_{max} + |F|_{max}] \cdot e^{i\Phi_F} && (3.25b) \end{aligned}$$

Eq. 3.25b can be easily manipulated into a more useful representation:

$$F(\xi,\eta) = |F|_{max} \cdot \left[1 - \left[\frac{|F|_{max} - |F(\xi,\eta)|}{|F|_{max}}\right]\right] \cdot e^{i\Phi_F} \quad (3.26)$$

By direct substitution of Eq. 3.26 into Eq. 3.23a, the ideal matched filter transfer function can be written as

$$H_{ideal}(\xi,\eta) = \frac{e^{-i\Phi_F}}{|F|_{max}} \cdot \left[1 - \left[\frac{|F|_{max} - |F(\xi,\eta)|}{|F|_{max}}\right]\right]^{-1} \quad (3.27)$$

The last term of Eq. 3.27 can now be expressed using the Taylor series expansion,

$$f(t) = \frac{1}{1-t} = \sum_{n=0}^{\infty} t^n \quad (3.28)$$

when $|t| < 1$. This result yields an expression for the transfer function of the complement matched filter.

$$\boxed{H_N(\xi,\eta) = \frac{e^{-i\Phi_F}}{|F|_{max}} \cdot \sum_{n=0}^{N} \left[\frac{|F|_{max} - |F(\xi,\eta)|}{|F|_{max}}\right]^N} \quad (3.29)$$

Thus, we arrive at a solution that may be truncated at any order N to approximate the ideal matched filter. For example, when $|F(\xi,\eta)|$ is normalized, the 0^{th}–order complement matched filter is just a POMF. In the limit $N \to \infty$, it acts as the ideal matched filter. Justification for naming the filter the "complement" matched filter can be seen by adding the first-term such that

$$\begin{aligned} H_1(\xi,\eta) &= [2 - |F(\xi,\eta)|] \cdot e^{-i\Phi_F} && (3.30a) \\ &= [1 + [1 - |F(\xi,\eta)|]] \cdot e^{-i\Phi_F} && (3.30b) \end{aligned}$$

where $1 - F(\xi,\eta)$ is the complement of the normalized magnitude.

Instead of exhibiting characteristics of the ideal matched filter as $N \to \infty$, the complement matched filter can be extended:

$$H_N(\xi,\eta) = \frac{e^{-i\Phi_F}}{|F|_{max}} \cdot \left[\sum_{n=0}^{N} \left[\frac{|F|_{max} - |F(\xi,\eta)|}{|F|_{max}} \right]^N \right]^{-1} \tag{3.31}$$

where the transfer function now acts more like the MSF as the order of approximation increases. This form of the complement matched filter [52] may be useful for applications with a large amount of noise and clutter.

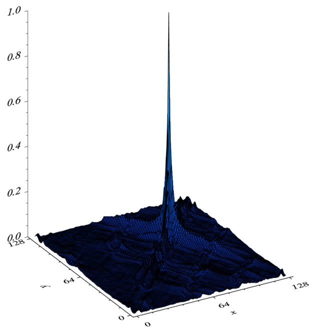
Figure 3.8 shows correlation peaks produced using the images in Figure 3.3 and the complement matched filter in Eq. 3.29 at three different orders of approximation ($N = 1$, $N = 10$, and $N = 10000$). As expected, the results from the 1^{st}-order complement matched filter are similar to those produced by the POMF in Figure 3.5. When the order is increased to $N = 10$, the magnitude term sharpens the peak slightly, and the noise floor is more noticeable for the case of the rotated reference pattern. The $10,000^{th}$-order complement matched filter behaves like the ideal matched filter and produces output correlation planes almost identical to those shown in Figure 3.4. Thus, the order of approximation in the magnitude term selects a point on the trade-off continuum for noise tolerance and peak sharpness.

The *locally nonlinear matched filter* (LNMF), developed by Gualdron and Arsenault [17], is very similar to the complement matched filter, but the properties of the ideal matched filter, POMF, and MSF are essentially "blended" rather than "tuning" the filter. The position on the continuum of peak sharpness and noise tolerance trade-off moves as a function of spatial frequency. The expression for the LNFM transfer function is:

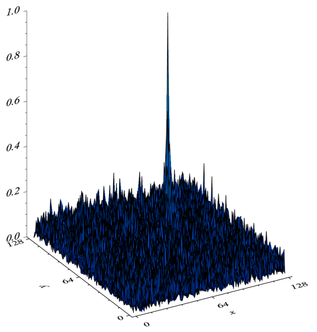
$$\boxed{H(\xi,\eta) = \frac{F(\xi,\eta)}{|F(\xi,\eta)|^m}} \tag{3.32}$$

where m is some function of the radial spatial frequency ρ. Gualdron and Arsenault suggest using a Gaussian function to vary these filter attributes across the spectral plane. Using Eq. 3.33 for m, the LNMF behaves more like the ideal matched filter at low spatial frequencies and more like the MSF at high spatial frequencies, which are generally more susceptible to noise amplification.

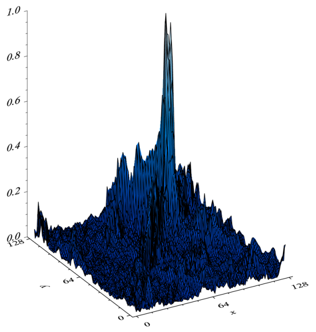
$$m = f(\rho) = 2 \cdot e^{-k\rho^2} \tag{3.33}$$



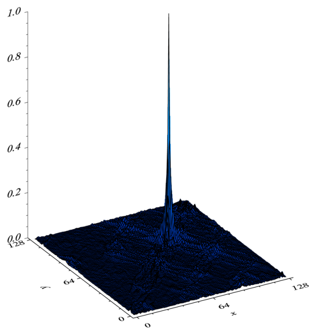
(a) Translation (Order = 1)



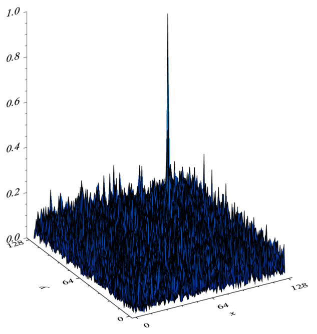
(b) Additive WGN (Order = 1)



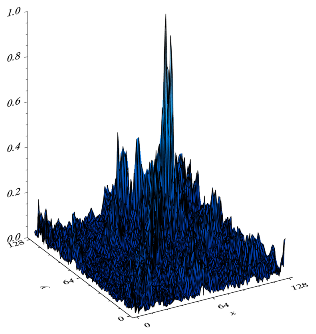
(c) 5° Rotation (Order = 1)



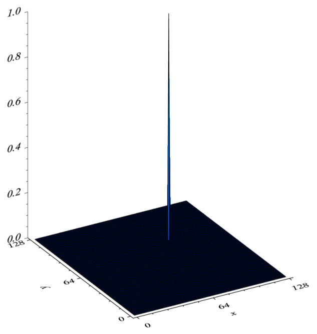
(d) Translation (Order = 10)



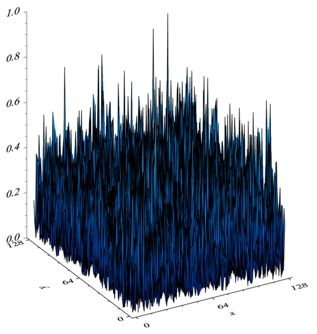
(e) Additive WGN (Order = 10)



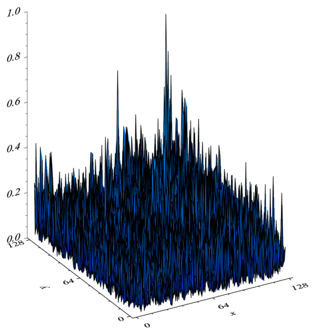
(f) 5° Rotation (Order = 10)



(g) Translation (Order = 10000)

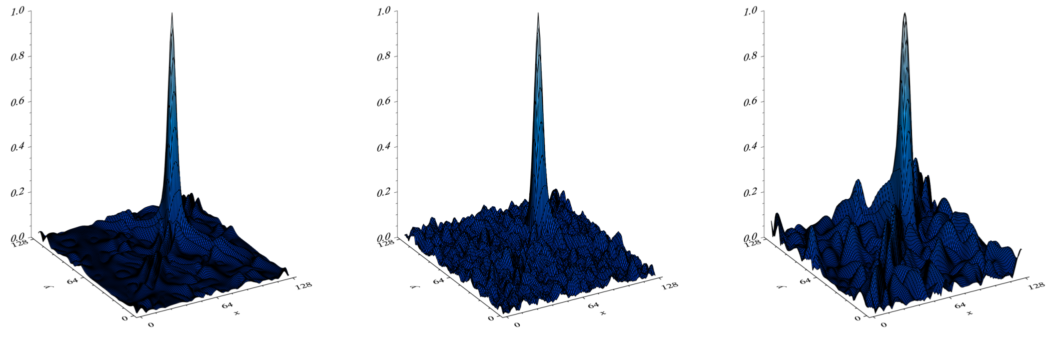


(h) Additive WGN (Order = 10000)



(i) 5° Rotation (Order = 10000)

Figure 3.8: Complement matched filter example – normalized correlation planes for the 1^{st}-order (a-c), 10^{th}-order (d-f), $10,000^{th}$-order (g-l) complement matched filter using the reference and test images in Figure 3.3.



(a) Translation (b) Additive WGN (c) 5° Rotation

Figure 3.9: Locally nonlinear matched filter example – normalized correlation planes for the LNMF using the reference and training images in Figure 3.3.

The LNMF yields higher peak sharpness, lower noise sensitivity, and better discrimination ability than the POMF in Eq. 3.12 [17].

Some interesting points should be addressed regarding the peaks produced by the LNMF in Figure 3.9. These peaks are sharper than those resulting from the application of the MSF (Figure 3.7), and the noise floor is suppressed more than that in the correlation planes produced using the POMF (Figure 3.5). However, in the case when the input contains no distortion, the POMF produces a sharper peak than the LNMF due to the LNMF acting as a lowpass filter at high spatial frequencies, which effectively reduces the support of the input spectrum.

3.2.1.5 Limitations in Practical Applications

We have been primarily concerned with controlling the tradeoff between noise sensitivity and correlation peak sharpness to develop expressions for the correlation filters introduced thus far. In doing so, we have neglected a major drawback that exists in nearly all matching problems: *the demand to recognize distorted versions of the reference pattern*. It is important to address not only noise tolerance and peak shape, but the *within-class distortion tolerance* such that correlation filters can be designed for more realistic recognition problems. The filters presented in the following section expand on the inherent properties of the ideal matched filter, phase-only matched filter, and crosscorrelation to achieve similar control over the correlation peak shape in the presence of spatially-distorted targets.

3.2.2 Composite Correlation Filters

As with all pattern recognition schemes, we desire correlation filters that are capable of accurately classifying distorted versions of the reference pattern, in the presence of noise and clutter, while maintaining a low false alarm rate. *Composite correlation filters* address many of the limitations inherent in classical correlation design by using a set of training images to provide several representative views of the reference pattern containing the expected distortions. Obviously, one objective in the design of composite correlation filters is to recognize distorted versions of the reference pattern that are absent from the training set. This *within-class distortion tolerance* provided by the correlation filter, as well as many other desirable characteristics such as high noise tolerance and sharp peaks, are criteria commonly specified during algorithm development. Integration of these criteria into the design of composite correlation filters is examined in this section. In addition, this section introduces matrix-vector notation to better represent the ensemble of images in the training set.

3.2.2.1 Synthetic Discriminant Functions

One of the first correlation filters to incorporate the spatial characteristics from a library of training images into its design was the *synthetic discriminant function* (SDF) filter. It was originally conceived by Hester and Casasent in the early 1980's to recognize an input target belonging to the set of reference training images and otherwise rejecting it [19]. A rather novel idea at the time, the filter was derived from an expression for the shape of a *decision boundary* in a multidimensional vector space.

In conventional pattern classification, the projection of an input feature vector onto the discriminant feature vector (that is orthogonal to a linear decision boundary) yields a scalar quantity used to make a decision (Figure 3.10). The design of SDF filters uses a similar approach: training images are represented as a set of vectors in a derived *hyperspace*, and a decision boundary can be constructed to differentiate between distorted versions of the reference pattern and false class inputs. However, the *basis functions* that define this hyperspace are 2-dimensional spatial functions (images) ob-

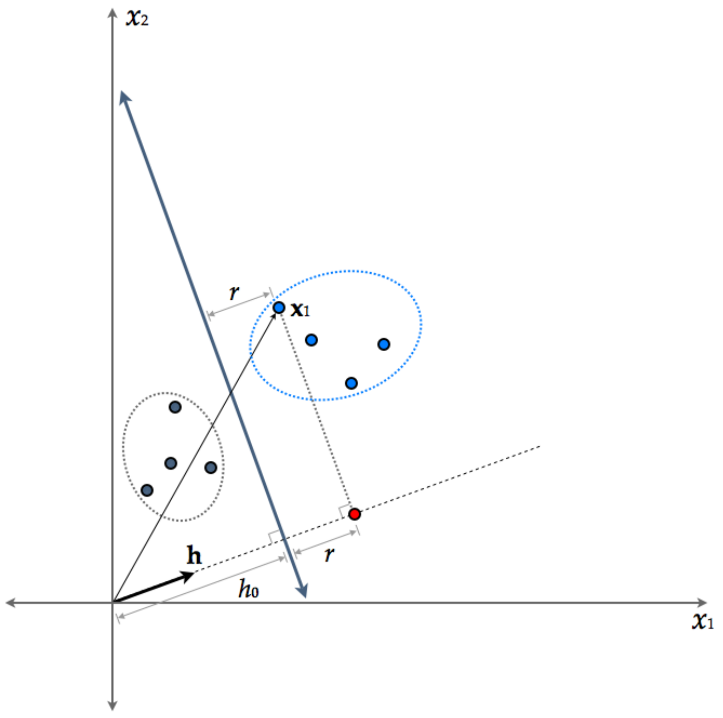


Figure 3.10: A linear discriminant function in 2-dimensions – an input feature vector $\mathbf{x}_1$ is projected onto the discriminant function $\mathbf{h}$ that points normal to the linear decision boundary. The distance from the origin to the decision boundary (in the direction of $\mathbf{h}$) is denoted by h_0. By subtracting this bias from the projection $\mathbf{h}^T\mathbf{x}_1$, the sign of the distance r is used to classify $\mathbf{x}_1$. As shown, $r > 0$ implies that the feature vector belongs in the class to the right of the decision boundary.

tained by an orthogonal basis function expansion of the training set. Consequently, the *discriminant function* that describes the orientation of the *hyperplane* decision boundary can also be interpreted as a 2-dimensional spatial function [2]. Thus, the projection of an input image vector onto the discriminant vector can be viewed either as (1) *the sum of the orthogonal distances from both the input vector and the origin to the decision surface* or (2) *the value at the origin of the correlation plane resulting from correlating the input image with the 2-dimensional spatial representation of the discriminant function*. The duality of this problem is illustrated in Figure 3.11. An important fact to note is that the essential property of shift invariance can be ensured by correlating the spatial image representations of both the input and discriminant vector rather than simply projecting one onto the other.

With the aforementioned motivation, we can now develop an expression for the SDF filter [3]. Assume that we are given a set of N training images that have been lexicographically ordered into d-element column vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ (where d is the number of pixels). Also, note that these vectors are assumed to be *linearly independent* of one another. We first represent each training image vector $\mathbf{x}_i$ as a linear combination of orthonormal basis functions $\boldsymbol{\phi}_j$:

$$\mathbf{x}_i = \sum_j a_{ij} \boldsymbol{\phi}_j \tag{3.34}$$

It follows that the $d \times 1$ filter vector $\mathbf{h}$ (the discriminant function vector) can be expressed in terms of the same orthonormal basis:

$$\mathbf{h} = \sum_j b_j \boldsymbol{\phi}_j \tag{3.35}$$

The original SDF filter was designed such that the projection of the filter vector onto the i^{th} training image $\mathbf{x}_i$ would produce some specified value u_i.

$$\mathbf{h}^T \mathbf{x}_i = \mathbf{x}_i^T \mathbf{h} = u_i \tag{3.36}$$

Thus, the goal is to find the filter vector $\mathbf{h}$ pointing normal to a hyperplane of training image vectors that is offset from the origin by the bias constraint u_i. As previously mentioned, this imples that u_i is a hard constraint placed on the origin of the

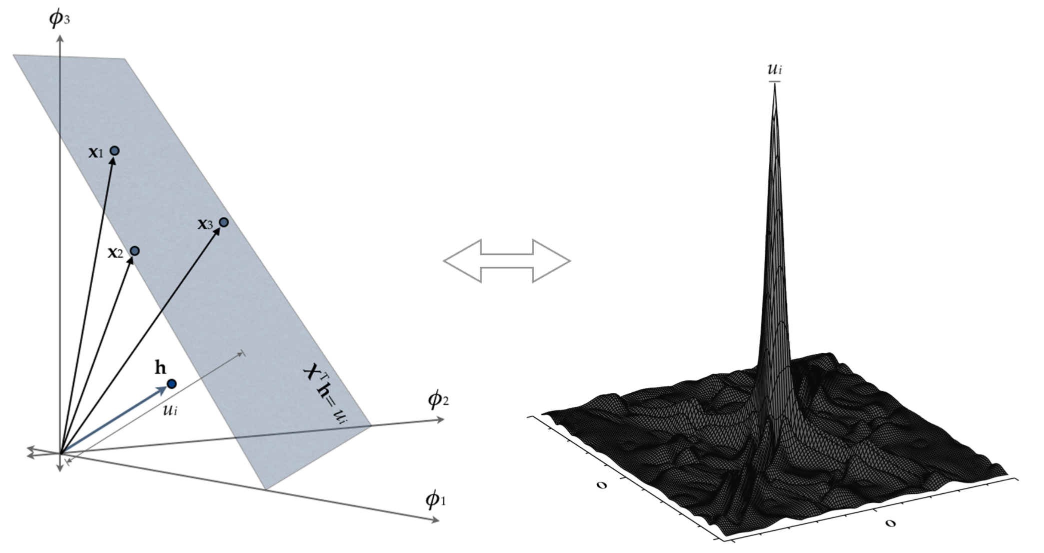


Figure 3.11: Conceptual duality for using a linear discriminant function for image classification – (Left) three training image vectors $\mathbf{x}_i$ lie on a hyperplane surface defined by an orthonormal vector, the discriminant function $\mathbf{h}$. The projection of all vectors that lie on this surface onto the discriminant vector $\mathbf{h}$ will yield the scalar value u_i, which denotes the orthonormal distance (or bias) from the origin to the hyperplane. (Right) Equivalently, the spatial correlation of the training vectors with the discriminant vector (as 2-dimensional spatial image representations) is constrained to u_i at the origin of the correlation plane.

correlation plane $g_i(m,n)$ between the training image $x_i(m,n)$ and the filter $h(m,n)$:

$$g_i(0,0) = \sum_m \sum_n x_i(m,n)h(m,n) = u_i \tag{3.37}$$

The projection in Eq. 3.36 can be rewritten by substituting the orthonormal basis set expressions for the training and filter vectors:

$$\mathbf{x}_i^T \mathbf{h} = \sum_j a_{ij} b_j = u_i \tag{3.38}$$

We now recognize that since both $\mathbf{x}_i$ and $\mathbf{h}$ are specified in the same basis, the particular solution for the filter $\mathbf{h}$ can be expressed as a linear combination of the training $\mathbf{x}_i$. Rearranging Eq. 3.34, the set of basis functions can be written as

$$\boldsymbol{\phi}_j = \sum_i d_{ij} \mathbf{x}_i \tag{3.39}$$

Substituting Eq. 3.39 into the expression for the filter vector in Eq. 3.35 yields:

$$\begin{aligned} \mathbf{h} &= b_1 \sum_i d_{i1} \mathbf{x}_i + b_2 \sum_i d_{i2} \mathbf{x}_i + \dots && \text{(3.40a)} \\ &= e_1 \mathbf{x}_1 + e_2 \mathbf{x}_2 + \dots && \text{(3.40b)} \\ &= \sum_i e_i \mathbf{x}_i && \text{(3.40c)} \end{aligned}$$

where e_i denotes the weight of vector $\mathbf{x}_i$ in the linear combination of $\mathbf{h}$. We can now substitute the expression for the filter into the projection equation (3.36) to obtain

$$\mathbf{x}_i^T \mathbf{h} = \mathbf{x}_i^T \left[\sum_i e_i \mathbf{x}_i \right] = \sum_i e_i (\mathbf{x}_i^T \mathbf{x}_i) = u_i \tag{3.41}$$

which can be rewritten in matrix form as

$$(\mathbf{X}^T \mathbf{X})\mathbf{e} = \mathbf{u} \tag{3.42}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$ is a $d \times N$ matrix of training images, and $\mathbf{e} = [e_1, e_2, \dots, e_N]^T$ and $\mathbf{u} = [u_1, u_2, \dots, u_N]^T$ are $N \times 1$ column vectors of corresponding weights and desired peak values, respectively. Thus, we arrive at an expression for the coefficients

e_j of the linear combination of training images:

$$\mathbf{e} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{u} \tag{3.43}$$

Casasent notes that for SDF filters designed to produce equal correlation outputs for all targets in the true class, the vector space described by the correlation matrix $\mathbf{X}^T\mathbf{X}$ is ideal to compute the coefficients for the linear combination of training vectors [2]. The resulting projection SDF filter is:

$$\boxed{\mathbf{h} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{u}} \tag{3.44}$$

It is now worth pausing to examine the reason why this filter design is of limited use as a pattern recognition tool. First, the SDF filter is derived such that its discriminant function representation defines a hyperplane on which each training vector lies (Figure 3.11). Input vectors that are in-class but nontraining may not lie directly on this surface, and will thus produce values u_i other than those specified. Moreover, hard constraints in the SDF filter design control only the origin of the correlation plane. Characteristic large sidelobes can exceed the constraint placed on the origin, essentially burying the correlation peak [31]. The composite correlation filters discussed hereafter address these issues by attempting to control the entire correlation plane in similar fashion to the classical correlation filters reviewed in Section 3.2.1.

3.2.2.2 Minimum Average Correlation Energy Filters

The *minimum average correlation energy* (MACE) filter was developed shortly after the SDF filter to address the issue of suppressing large sidelobes in the correlation plane. Mahalanobis, et al., [34] note that to provide adequate detection, the correlation filter should reduce the intensity of the correlation plane at all nonpeak locations (corresponding to locations in which the target is absent). Thus, to develop a formulation for the MACE filter, we begin by defining the criterion that it minimize the *average correlation energy* (ACE) of the correlation outputs for each of the N training images

$$ACE = \frac{1}{N}\sum_{i=1}^{N}\sum_{m}\sum_{n}|g_i(m,n)|^2 \tag{3.45}$$

where $g_i(m,n)$ is the correlation plane produced for the i^{th} training image. Using the discrete form of Parseval's theorem, the ACE can be expressed in the frequency domain as

$$ACE = \frac{1}{N \cdot d} \sum_{i=1}^{N} \sum_{k} \sum_{l} |G_i(k,l)|^2 \tag{3.46}$$

where $G_i(k,l)$ is the Fourier transform of $g_i(m,n)$. This expression can now be expanded in terms of the filter transfer function $H(k,l)$ and training spectra $X_i(k,l)$:

$$ACE = \frac{1}{N \cdot d} \sum_{i=1}^{N} \sum_{k} \sum_{l} |H(k,l)|^2 |X_i(k,l)|^2 \tag{3.47}$$

In matrix-vector notation, the filter $H(k,l)$ can be represented as a $d \times 1$ vector $\mathbf{h}$ and the training spectra $X_i(k,l)$ as $d \times d$ diagonal matrices $\mathbf{X}_i$, where the diagonal elements contain the i^{th} training spectrum. The ACE can now be written:

$$\begin{aligned} ACE &= \frac{1}{N \cdot d} \sum_{i=1}^{N} (\mathbf{h}^{+}\mathbf{X}_i)(\mathbf{X}_i^{*}\mathbf{h}) && (3.48a) \\ &= \mathbf{h}^{+} \left[\frac{1}{N \cdot d} \sum_{i=1}^{N} \mathbf{X}_i \mathbf{X}_i^{*} \right] \mathbf{h} && (3.48b) \\ &= \mathbf{h}^{+}\mathbf{D}\mathbf{h} && (3.48c) \end{aligned}$$

where $\mathbf{D}$ is a $d \times d$ diagonal matrix containing the average training power spectra.

Analogous to the SDF filter, the MACE filter is designed to ensure that the inner product of the filter vector and the training spectrum produces a specified output peak amplitude u_i at the origin of the correlation plane; this is multiplied by the scale factor d in the frequency domain. This condition can be expressed for all training images and their corresponding constant values as

$$\mathbf{X}^{+}\mathbf{h} = d \cdot \mathbf{u} \tag{3.49}$$

where $\mathbf{X}$ now represents a $d \times N$ matrix with columns consisting of lexicographically-ordered training spectra.

We now have a quadratic function, specified by the ACE, and a linear condition, specified by the hard constraints. Minimizing Eq. 3.48c subject to Eq. 3.49 is a *con-*

strained minimization problem, which can be solved by the method of *Lagrange multipliers* (Appendix A.2). The Lagrangian function [34] can be expressed as:

$$L(\mathbf{h};\lambda) = \mathbf{h}^{+}\mathbf{D}\mathbf{h} - 2\lambda_1(\mathbf{x}_1^{+}\mathbf{h} - u_1) - \ldots - 2\lambda_N(\mathbf{x}_N^{+}\mathbf{h} - u_N) \tag{3.50}$$

where the λ_i's are the Lagrange multipliers. We can now solve for the filter **h** by setting the derivative of Eq. 3.50 to zero:

$$\mathbf{D}\mathbf{h} = \lambda_1\mathbf{x}_1 + \ldots + \lambda_N\mathbf{x}_N \tag{3.51a}$$

$$\mathbf{h} = \mathbf{D}^{-1}\left[\sum_{i=1}^{N}\lambda_i\mathbf{X}_i\right] = \sum_{i=1}^{N}\lambda_i\mathbf{D}^{-1}\mathbf{X}_i = \mathbf{D}^{-1}\mathbf{X}\mathbf{l} \tag{3.51b}$$

where $\mathbf{l} = [\lambda_1, \lambda_2, \ldots, \lambda_N]^{+}$. Substitution of Eq. 3.51b into Eq. 3.49 yields

$$\mathbf{X}^{+}\mathbf{D}^{-1}\mathbf{X}\mathbf{l} = d \times \mathbf{u} \tag{3.52}$$

The vector of Lagrange multipliers **l** can be evaluated via:

$$\mathbf{l} = (\mathbf{X}^{+}\mathbf{D}^{-1}\mathbf{X})^{-1}u \tag{3.53}$$

and substituted into Eq. 3.51b to produce the solution for the MACE filter:

$$\boxed{\mathbf{h} = \mathbf{D}^{-1}\mathbf{X}(\mathbf{X}^{+}\mathbf{D}^{-1}\mathbf{X})^{-1}\mathbf{u}} \tag{3.54}$$

Numerous variations of matched filtering algorithms have been derived using the simple ideal matched filter in Eq. 3.7 as the benchmark for perfect noise-free correlation. The MACE filter is no exception. In the case of a single training image, the MACE filter is in fact the ideal matched filter. Because the MACE filter is similar to an inverse filter design, they share many of the same advantages and disadvantages. Sharp correlation peaks are typically attained for regions of the input corresponding to exact replicas of images in the training set, while the remainder of the correlation plane is controlled by minimizing the ACE. However, as expected, the MACE filter suffers from high noise sensitivity and low within-class distortion tolerance.

Kumar, et al., [30] motivated the removal of hard constraints in the SDF (Eq. 3.44) and MACE (Eq. 3.54) filter algorithms. They note that nontraining inputs will typically produce different peak values than those specified for training inputs, and no relationship between the hard constraints and tolerance to within-class distortions has been established. In addition, it is believed that removing the hard constraints may increase the filter solution domain.

We begin to develop a formulation for an unconstrained version of the MACE filter by referring back to the ACE in Eq. 3.48c. Without the linear condition specified in Eq. 3.49, the optimization problem must be slightly reworked. Using the central ordinate theorem, the origin of the correlation plane produced for the i^{th} training image $g_i(0,0)$ (the peak location during filter design) can be expressed in the frequency domain as the inner product of the i^{th} training spectrum $\mathbf{x}_i$ and the filter vector $\mathbf{h}$:

$$g_i(0,0) = \mathbf{h}^+\mathbf{x}_i \tag{3.55}$$

While the constraints on the origin are now unspecified, it makes sense to maximize the value of the correlation peak in the new objective function [43]. By maximizing the square of the average magnitude of the peak $|\mathbf{h}^+\mathbf{m}|$, while minimizing the ACE $\mathbf{h}^+\mathbf{D}\mathbf{h}$, the criterion function $J(\mathbf{h})$ is obtained:

$$J(\mathbf{h}) = \frac{\mathbf{h}^+\mathbf{m}\mathbf{m}^+\mathbf{h}}{\mathbf{h}^+\mathbf{D}\mathbf{h}} \tag{3.56}$$

which is in the form of a *Rayleigh quotient* (Appendix A.1.2). By maximizing $J(\mathbf{h})$ with respect to $\mathbf{h}$, we arrive at an expression in the form of a *generalized eigenvalue problem* (Appendix A.1):

$$\mathbf{m}\mathbf{m}^+\mathbf{h} = J(\mathbf{h})\mathbf{D}\mathbf{h} \tag{3.57}$$

Assuming the diagonal matrix $\mathbf{D}$ is invertible, Eq. 3.57 can be rearranged:

$$\mathbf{D}^{-1}\mathbf{m}\mathbf{m}^+\mathbf{h} = J(\mathbf{h})\mathbf{h} \tag{3.58}$$

The dominant eigenvector of $\mathbf{D}^{-1}\mathbf{m}\mathbf{m}^+$ is the optimum filter:

$$\boxed{\mathbf{h} = \mathbf{D}^{-1}\mathbf{m}} \tag{3.59}$$

Eq. 3.59 is the expression for the *unconstrained minimum average correlation energy* (UMACE) filter. By removing the constraints placed on the peak value, a simple "average" ideal matched filter is obtained. The result is a filter with improved within-class distortion tolerance that retains the sharp peaks and suppressed correlation floor characteristic of the MACE filter [35].

3.2.2.3 Maximum Average Correlation Height Filters

The *maximum average correlation height* (MACH) filter [35] is an unconstrained composite correlation filter, similar to the UMACE filter in the previous section, that satisfies three different conditions. Most notably, the MACH filter introduced a condition to address the issue of maximizing within-class distortion tolerance. The crosscorrelation between each image in the training set and the input function produces a series of output correlation planes. Depending on the similarity between the individual exemplars and the input, some correlation planes may signal better detection characteristics than others. The *average similarity measure* (ASM) [35] is used to quantify the deviation between each of these N individual correlation outputs $g_i(m,n)$ to the mean $\bar{g}(m,n)$. In the spatial domain, the ASM can be expressed:

$$ASM = \frac{1}{N}\sum_{i=1}^{N}\sum_{m}\sum_{n}|g_i(m,n) - \bar{g}(m,n)|^2 \tag{3.60}$$

The corresponding frequency domain representation is

$$ASM = \frac{1}{N \cdot d}\sum_{i=1}^{N}\sum_{k}\sum_{l}|G_i(k,l) - \bar{G}(k,l)|^2 \tag{3.61}$$

where d is the number of pixels in the correlation plane, and $G_i(k,l)$ and $\bar{G}(k,l)$ again denote the Fourier transform of $g_i(m,n)$ and $\bar{g}(m,n)$, respectively.

In matrix-vector notation, we can again define the correlation filter as the $d \times 1$ column vector $\mathbf{h}$, the Fourier transform of the i^{th} training image as the $d \times 1$ column vector $\mathbf{x}_i$ or as the $d \times d$ diagonal matrix $\mathbf{X}_i$, and the mean Fourier transform of the training set as the $d \times 1$ column vector $\mathbf{m}_i$ or as the $d \times d$ diagonal matrix $\mathbf{M}_i$. These representations allow $g_i(m,n)$ and $\bar{g}(m,n)$ to be expressed as $d \times 1$ column vectors, $\mathbf{g}_i = \mathbf{X}_i^*\mathbf{h}$ and $\bar{\mathbf{g}} = \mathbf{M}^*\mathbf{h}$. The ASM can now be expressed:

$$ASM = \frac{1}{N \cdot d}\sum_{i=1}^{N}|\mathbf{g}_i - \bar{\mathbf{g}}|^2 \tag{3.62}$$

and simplified:

$$\begin{aligned} ASM &= \frac{1}{N \cdot d} \sum_{i=1}^{N} \left| \mathbf{X}_i^* \mathbf{h} - \mathbf{M}^* \mathbf{h} \right|^2 & \text{(3.63a)} \\ &= \frac{1}{N \cdot d} \sum_{i=1}^{N} \mathbf{h}^+ \left(\mathbf{X}_i - \mathbf{M} \right) \left(\mathbf{X}_i - \mathbf{M} \right)^* \mathbf{h} & \text{(3.63b)} \\ &= \mathbf{h}^+ \left[\frac{1}{N \cdot d} \sum_{i=1}^{N} \left(\mathbf{X}_i - \mathbf{M} \right) \left(\mathbf{X}_i - \mathbf{M} \right)^* \right] \mathbf{h} & \text{(3.63c)} \end{aligned}$$

Defining the $d \times d$ diagonal matrix $\mathbf{S} = \frac{1}{N \cdot d} \sum_{i=1}^{N} \left(\mathbf{X}_i - \mathbf{M} \right) \left(\mathbf{X}_i - \mathbf{M} \right)^*$ we arrive at the final expression for the ASM:

$$ASM = \mathbf{h}^+ \mathbf{S} \mathbf{h} \tag{3.64}$$

The second condition is the *average correlation height* measure. Any correlation peak that signals detection of the specified target should exhibit a large amplitude relative to the rest of the correlation plane. Assuming the peak is located at the origin (as is custom during filter design), the spatial domain representation of the ACH is defined:

$$ACH = \frac{1}{N} \sum_{i=1}^{N} g_i(0,0) \tag{3.65}$$

Using the central limit theorem, we obtain the frequency domain representation:

$$\begin{aligned} ACH &= \frac{1}{N \cdot d} \sum_{i=1}^{N} \sum_{k} \sum_{l} G_i(k,l) & \text{(3.66a)} \\ ACH &= \frac{1}{N \cdot d} \sum_{i=1}^{N} \sum_{k} \sum_{l} X_i^*(k,l) H(k,l) & \text{(3.66b)} \end{aligned}$$

Thus, in matrix-vector notation, the ACH is:

$$ACH = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}^+ \mathbf{h} = \mathbf{m}^+ \mathbf{h} \tag{3.67}$$

The final condition is the *output noise variance* (ONV), and it is used to control noise amplification by the filter. This measure was also used in the development of earlier composite correlation filters, such as the *minimum variance synthetic discriminant* *function* (MVSDF) [27], and is expressed:

$$ONV = \mathbf{h}^{+}\mathbf{C}\mathbf{h} \tag{3.68}$$

where $\mathbf{C}$ is the $d \times d$ noise covariance matrix, in which the estimated noise of the input is assumed to be additive with zero mean. For the case of white noise, the noise covariance $\mathbf{C}$ is just the identity matrix $\mathbf{I}$.

Using the expressions for the ASM, ACH, and ONV, we can now formulate the criterion function $J(\mathbf{h})$ to derive an expression for the MACH filter. The filter should maximize the ACH while simultaneously minimizing the ASM and ONV. To achieve this goal, the criterion function to maximize is:

$$J(\mathbf{h}) = \frac{|ACH|^2}{ASM + ONV} \tag{3.69a}$$

$$= \frac{|\mathbf{m}^{+}\mathbf{h}|^2}{\mathbf{h}^{+}\mathbf{S}\mathbf{h} + \mathbf{h}^{+}\mathbf{C}\mathbf{h}} = \frac{\mathbf{h}^{+}\mathbf{m}\mathbf{m}^{+}\mathbf{h}}{\mathbf{h}^{+}(\mathbf{S}+\mathbf{C})\mathbf{h}} \tag{3.69b}$$

We again arrive at an expression in the form of a Rayleigh quotient (see Appendix A.1.2). Maximizing $J(\mathbf{h})$ with respect to $\mathbf{h}$ yields:

$$\mathbf{m}\mathbf{m}^{+}\mathbf{h} = J(\mathbf{h})(\mathbf{S}+\mathbf{C})\mathbf{h} \tag{3.70}$$

The $d \times d$ matrix $(\mathbf{S}+\mathbf{C})$ is diagonal, and it inversion is therefore trivial, allowing Eq. 3.70 to be written:

$$(\mathbf{S}+\mathbf{C})^{-1}\mathbf{m}\mathbf{m}^{+}\mathbf{h} = J(\mathbf{h})\mathbf{h} \tag{3.71}$$

This expression is now in the form of a standard eigenvalue problem (Appendix A.1). Thus, the filter vector $\mathbf{h}$ is the dominant eigenvector of $(\mathbf{S}+\mathbf{C})^{-1}\mathbf{m}\mathbf{m}^{+}$. The MACH filter is:

$$\boxed{\mathbf{h} = \gamma(\mathbf{S}+\mathbf{C})^{-1}\mathbf{m}} \tag{3.72}$$

where γ is just a normalizing constant.

The MACH filter is a composite correlation filter that is very similar to the simple crosscorrelation using the mean training image. If $\mathbf{C} = \mathbf{I}$, Eq. 3.72 becomes the crosscorrelation for the case of the training set consisting of a single image. The numerator of the expression is responsible for phase canceling and for filtering high frequencies containing noise and sharp edges (assuming the training is low-frequency dominated). For frequencies exhibiting a large deviation between the individual exemplars and the mean, the magnitude of the filter is reduced. The same is true for frequencies with a high noise contribution. Conversely, if the sum of the noise and the deviation within the training is less than one at a particular frequency, the MACH filter will amplify the corresponding frequency in the input. Thus, the MACH filter can be interpreted as the mean-training crosscorrelation with weights applied to each frequency to control attenuation and amplification based on the ASM and ONV. The MACH filter has been used for numerous applications since its inception due to its high tolerance to noise and clutter.

3.2.3 Performance Analysis

Features for image classification can be extracted by post-processing the output correlation plane of the filtered input pattern. Common algorithms detect peaks by assigning a metric (or metrics) that quantify the response of the filtering. The need to *detect* output correlation peaks from filters designed to provide *detection* may seem counterintuitive. In the ideal scenario, the filtered output would contain correlation peaks approximating Dirac delta functions, and a simple threshold would yield the exact location of the reference pattern or patterns. However, variations in the intensity of the input signal directly influence the intensity of the correlation plane. Normalizing the peak amplitude with respect to the energy of the input can overcome this obstacle when potential targets are not obscured by background clutter [31]. When clutter is present, using the intensity as a *correlation performance metric* is insufficient for setting a detection threshold, and may subsequently result in poor classification. By characterizing the correlation output, correlation performance measures not only provide a method for evaluating filters, but often provide useful features for classification.

3.2.3.1 Correlation Performance Measures

Perhaps the most common performance measure for current correlation filtering applications is the well known *peak-to-sidelobe ratio* (PSR) [29], defined as

$$PSR = \frac{peak - \mu}{\sigma} \tag{3.73}$$

where μ and σ are the respective mean and standard deviation of a specified neighborhood surrounding the peak location (but excluding the peak itself). The neighborhood over which μ and σ are calculated should be selected based on the application and held consistent throughout testing and/or feature extraction. Rather than provide a measure of relative intensity, the PSR measures the sharpness of the correlation peak. This attribute of the peak is obviously of great importance, as broad correlation peaks corresponding to targets that are spatially close together will be difficult to resolve. PSR effectively discriminates true peaks from high noise fluctuations, which may otherwise resemble detection of multiple reference patterns, by accounting for variations in areas surrounding high amplitudes. Kumar, et al., [31] note that PSR is a useful tool for correlation-based classification as it characterizes the degree of similarity between regions in the input and the reference signal(s).

Another useful metric for characterizing sharpness is the *peak-to-correlation energy ratio* (PCE), which calculates the ratio between the squared-magnitude of the peak to the energy of the correlation plane $g(n, m)$:

$$PCE = \frac{|peak^2|}{\sum_n \sum_m |g(n,m)|^2} \tag{3.74}$$

Obviously, the energy term in the denominator of Eq. 3.74 can be computed over a specified neighborhood surrounding the *peak* (similar to PSR). In addition, the amplitude of the peak itself is often excluded from the set of pixels used to calculate the denominator [31]. It should be apparent that a lower energy term allows sharp peaks to achieve much higher PCE values than those of broad peaks.

Additional performance measures, which are used less frequently in current applications, are the familiar *signal-to-noise ratio* (SNR) and *full area at half maximum* (FAHM) [29]. SNR is used to quantify the filter's sensitivity to noise in the input, and thus received greater attention during classical correlation filter development when noise tolerance was the primary performance criterion under investigation. While FAHM metric characterizes the shape of the correlation peak, PSR and PCE are more widely accepted in the field of correlation pattern recognition as measures of peak sharpness.

3.2.3.2 ROC Curves and Confusion Matrices

The previous section discussed metrics used to characterize attributes of the correlation plane; we now shift focus to briefly examine two well-known techniques used to interpret the performance of the classification process. The first is the *receiver operating characteristic*, or ROC, curve. A useful survey paper detailing ROC analysis in current signal detection applications was recently published by Fawcett [11]. Assume that we are given a classifier designed to make a decision for whether a target is present or absent within some unknown input pattern. The classification for this binary detection problem (number of classes $c = 2$) can have four possible outcomes:

- *Correct Detection* – target is present in the input and detected by the classifier.
- *Miss* – target is present in the input and undetected by the classifier.
- *Correct Rejection* – target is absent in the input and undetected by the classifier.
- *False Alarm* – target is absent in the input and detected by the classifier.

A ROC curve is used to examine the relationship between the *probability of correct detection* P_D and the *probability of false alarm* P_{FA} as the threshold for detection varies. For many applications, the desired threshold (the *operating point* on the ROC curve) is not the threshold that yields the minimum probability of error. For example, it may only be possible to achieve the required probability of detection by accepting an increased level of false alarms. Figure 3.12 shows an example ROC curve used to analyze this tradeoff.

A few statements about ROC curves can now be made in reference to Figure 3.12. First, the line with unit slope corresponds to the worst possible performance for binary detection [31]. This case can be interpreted as a random guess by the classifier such that the correct decision is made fifty percent of the time regardless of the operating point. The performance of the classifier improves if the probability of detection is increased while simultaneously decreasing the probability of false alarm, effectively causing the ROC curve to approach a step function (corresponding to the best detection possible). The area under the ROC curve, which is commonly referred to as the *power of the detector* (POD), is a method for reducing the performance depicted by the ROC curve to a single scalar value. From the discussion above, it should be apparent

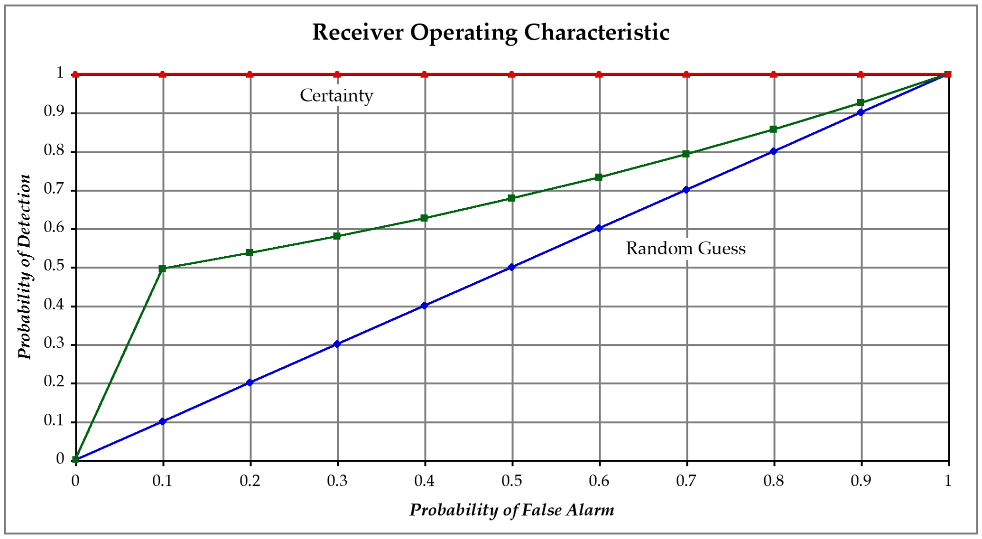


Figure 3.12: Example ROC curve for binary detection – the worst possible performance is depicted by the blue curve with unit slope. Increasing the probability of detection while simultaneously decreasing the probability of false alarm indicates improved classification (green curve). The red curve is a STEP function, which is obtained in the case of perfect classification.

that the POD ranges from 0.5 (guess) to 1.0 (certainty), depending on the performance of the classifier. It should be noted that a classifier may have a higher POD than another while having worse performance over a specific region of the ROC curve.

While ROC curves can be very useful tools for evaluating the performance of multiple classifiers, a couple of important points should be stressed before using them to make a decisive argument over which classifier to use. First and foremost, ROC curves are generally derived subject to a particular application or set of conditions. A specific classifier that exhibits poor performance characteristics for one application may provide more than adequate detection rates for another. Second, some measure of variance is essential when comparing the classifiers. Multiple ROC curves generated from multiple input patterns can be vertically-averaged or threshold-averaged to provide this measure variance [11].

For multi-class problems ($c > 2$), a ROC convex-hull must be generated to represent the entire $P_D - P_{FA}$ space, which is often difficult to interpret. Rather than

		Percentage Classified as Class			
		1	**2**	**3**	**4**
True Class	**1**	99.1%	0.4%	0.0%	0.5%
	2	5.1%	82.4%	1.3%	11.2%
	3	0.0%	0.0%	100.0%	0.0%
	4	0.1%	1.4%	0.2%	98.3%

Figure 3.13: Example confusion matrix for $c = 4$ classes – probabilities along the main diagonal correspond to correct detection; probabilities in diagonal elements are those due to false alarm. For example, 82.4% of the data from class 2 was correctly classified, but 11.2% was incorrectly classified as class 4.

produce different ROC curves for each class (in which all other classes correspond to errors), a *confusion matrix* is often analyzed to evaluate the performance of the classification. The confusion matrix, or *contigency table*, for a particular classifier is a $c \times c$ matrix with rows representing the label of the true class and columns representing the class label specified by the classifier (Figure 3.13.). Probabilities of correct decisions are contained along the c-element main diagonal; probabilities of error, which result from misclassification of the input patterns, are contained within the other $c^2 - c$ off-diagonal elements. To condense the results depicted in a large confusion matrix, it is common to show composite estimates of the accuracy of the classifier. This can be accomplished by simply computing the average, or weighted average, of the probabilities along the main diagonal [44].

While the overall recognition rate may seem useful for comparing different classification schemes, Congalton, et al., [8] notes that this measure of the accuracy is misleading; it overestimates the accuracy of the classifier by neglecting to account for the random probability of correct detection. By incorporating probabilistic information from the off-diagonal components of the confusion matrix, a more accurate measure of the classifier's true performance is obtained. The *kappa coefficient* $\hat{\kappa}$ is expressed:

$$\hat{\kappa} = \frac{p_o - p_c}{1 - p_c} \tag{3.75}$$

where p_o is the overall recognition rate (or simple accuracy) defined:

$$p_o = \frac{1}{N} \sum_{i=1}^{k} c_{ii} \tag{3.76}$$

k is the number of classes, N is the number of exemplars used to construct the confusion matrix, and c_{ii} is the i^{th} diagonal element of the matrix. The proportion of the accuracy due to random chance p_c is:

$$p_o = \frac{1}{N} \sum_{i=1}^{k} c_{ii} \tag{3.77}$$

where

$$c_{it} = \sum_{i=1}^{k} c_{ij} \tag{3.78}$$

and

$$c_{ti} = \sum_{i=1}^{k} c_{ji} \tag{3.79}$$

Furthermore, Congalton, et al., [7] formulate a measure for the variance in the kappa coefficient σ_k^2, and an example of its use in constructing confidence limits using Gaussian statistics is provided in [44]. The use of the kappa coefficient is revisited in Chapter 5 to compare classification schemes for handwritten character recognition.

While confusion matrices are a valuable tool for analyzing multi-class classifiers, it is important to consider how the matrix is generated before making any decisions about which classifier should be used for an application. The performance estimates often vary significantly between data sets that have been sent to the classifier. This intuitively makes sense; we would expect the classification accuracy for the *dependent data set*, consisting of exemplars used to train the classifier, to exceed that which was produced for the *independent data set*, consisting of similar, nontraining input patterns.

3.3 Statistical Methods

The focus of this chapter is now shifted slightly to examine some of the statistical tools that are more relevant to modern pattern recognition systems. These methods characterize the data by statistical parameters (e.g., mean and variance) and can be applied after feature extraction by correlation filtering. It will be shown in Chapter 4 that some of these techniques can even be performed as part of the filtering process to directly improve correlation results.

3.3.1 Principal Components Analysis

Principal components analysis (PCA) is a multivariate statistical technique used to re-express the original data set to better represent its variance-covariance structure. In 1933, Hotelling showed that correlated *variables*, which define the coordinate axes for the original set of *observations*, can be represented as uncorrelated variables in a new vector space [21, 22]. PCA projects the data onto this new coordinate system to achieve two specific objectives: *data reduction* and *data interpretation*.

The relevance to and importance of PCA in pattern recognition problems should be clear. An analogous concept has already been used extensively in this chapter: the conversion between spatial domain and frequency domain representations via the Fourier transform. Obviously, numerous advantages are known to arise from this transformation, including simplifying filtering processes to improving data interpretability. The *discrete Fourier transform* (DFT) projects the original vector (or vectors) onto an orthogonal matrix consisting of complex sinusoids with specific spatial frequencies. In other words, applying the DFT is just a *change of basis*; spatial domain vectors are transformed to frequency domain vectors by a rotation of the coordinate system and a change of scale as shown in Figure 3.14 [41].

The principal component (PC) transform, also commonly referred to as the discrete *Karhunen-Loève transform* (KLT) [13], is similar to the Fourier transform in that it defines a change of basis, but one fundamental difference exists; the PC basis is derived such that the rotation of the coordinate system *diagonalizes the covariance matrix* of the original set of vectors. This diagonalization effectively eliminates any covariance between separate variables in the new basis. Furthermore, the total variance of each variable is redistributed along the main diagonal of this diagonalized covariance ma-

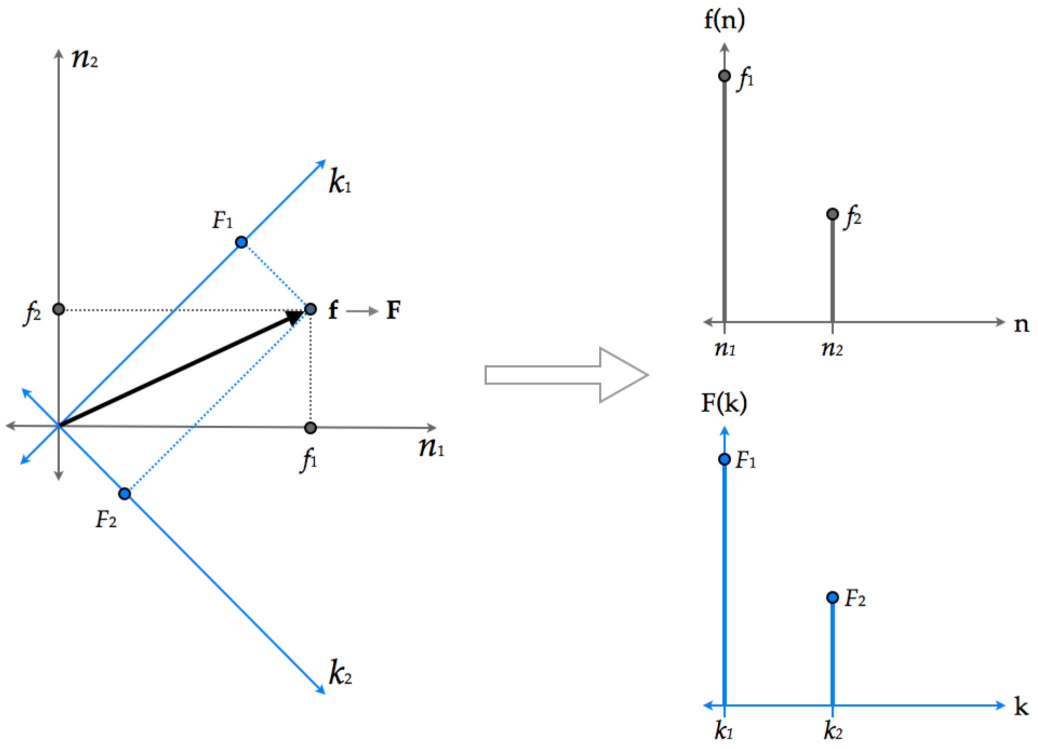


Figure 3.14: 2-point DFT as a change of basis – (Left) the original vector is shown with its original coordinates (f_1, f_2) in the spatial domain basis $[n_1, n_2]$ and with its projected coordinates (F_1, F_2) in the rotated basis (with respect to the original coordinate system) of the frequency domain $[k_1, k_2]$. (Right) Corresponding 2-point signal and 2-point DFT representations of the original vector.

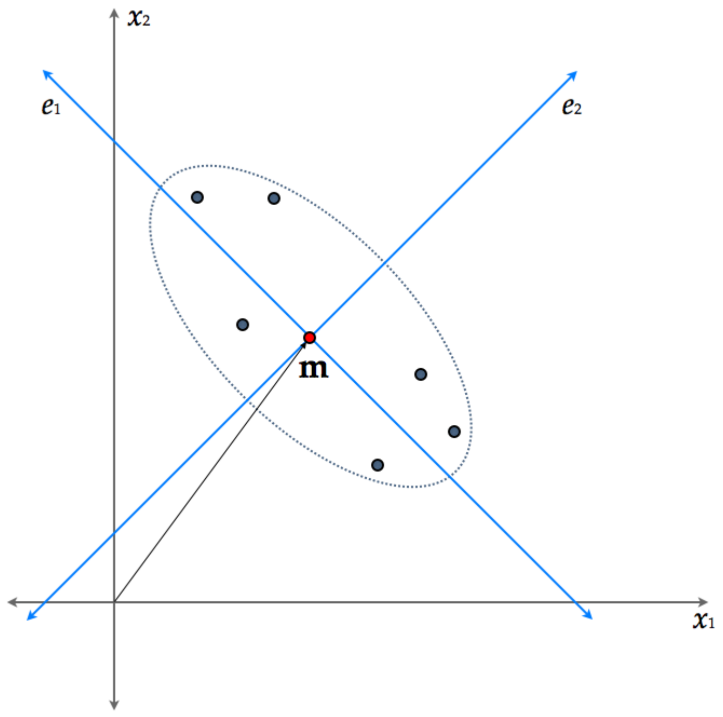


Figure 3.15: PCA as a change of basis in 2-dimensions – x_1 and x_2 are variables defining the original basis. e_1 and e_2 are linear combinations of x_1 and x_2 that define the basis of the PC transform. The origin has been translated to the mean observation vector **m** and rotated such that its axes point in directions of maximum variability.

trix such that the basis vectors point in the directions of maximum variability (Figure 3.15). Thus, while the basis vectors of the Fourier transform are *deterministic*, those of the PC transform can be considered *random*, as they are derived from the initial set of random variables in the data. This new basis is a linear transformation of the original basis, and it is computed to “best” represent the data in a least-squares sense.

3.3.1.1 The PC Transform Basis

To develop a formulation for the basis functions of the PC transform, we can begin by observing a simple optimization problem [9]. Assume that we are given a set of N

d-dimensional vectors $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$. The task is to find a vector $\mathbf{x}_0$ that best represents this set of vectors. In the context of spatial pattern recognition, each of the original vectors $\mathbf{x}_i$ is typically considered to be a particular observation of extracted features (a feature vector). Additionally, assume that the best representation for this particular application is the vector $\mathbf{x}_0$ such that the sum of squared distances to each vector $\mathbf{x}_i$ in the ensemble is minimized. To determine $\mathbf{x}_0$, we minimize the sum-squared error:

$$J_0(\mathbf{x}_0) = \sum_{i=1}^{N} ||\mathbf{x}_0 - \mathbf{x}_i||^2 \tag{3.80}$$

The intuitive solution to this problem is the sample mean vector $\mathbf{m}$,

$$\mathbf{m} = \frac{1}{n} \sum_{i=1}^{N} \mathbf{x}_i \tag{3.81}$$

This solution can be verified by subtracting $\mathbf{m}$ from both $\mathbf{x}_0$ and $\mathbf{x}_i$ and expanding the criterion function:

$$\begin{aligned}
J_0(\mathbf{x}_0) &= \sum_{i=1}^{N} ||(\mathbf{x}_0 - \mathbf{m}) - (\mathbf{x}_i - \mathbf{m})||^2 && (3.82a)\\
&= \sum_{i=1}^{N} ||\mathbf{x}_0 - \mathbf{m}||^2 - 2\sum_{i=1}^{N} (\mathbf{x}_0 - \mathbf{m})^T(\mathbf{x}_i - \mathbf{m}) + \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 && (3.82b)\\
&= \sum_{i=1}^{N} ||\mathbf{x}_0 - \mathbf{m}||^2 - 2(\mathbf{x}_0 - \mathbf{m})^T \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{m}) + \sum_{k=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 && (3.82c)\\
&= \sum_{i=1}^{N} ||\mathbf{x}_0 - \mathbf{m}||^2 + \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 && (3.82d)
\end{aligned}$$

Substitution of $\mathbf{x}_0 = \mathbf{m}$ into Eq. 3.82d forces the first term to zero, essentially minimizing the function by leaving only the second term (the residual error), which is independent of $\mathbf{x}_0$.

Unfortunately, the sample mean describes none of the variability in the original set of vectors about the mean. As stated previously, the goal of PCA is to better represent the variance-covariance structure in the data. Therefore, it is necessary to instead project the data onto some line $\mathbf{x}$ that runs through the sample mean. This effectively provides a 1-dimensional representation of the original set of vectors, rather than the

0-dimensional representation provided by **m**. The equation for this line is simply,

$$\mathbf{x} = \mathbf{m} + a\mathbf{e} \tag{3.83}$$

where **e** is a unit vector specifying the direction of **x**, and a is a real-valued scalar representing the distance between **m** and any point on the line.

Similar to the previous scenario, we now seek to find the line **x** whose sum of squared distances to each $\mathbf{x}_i$ is smallest. Thus, we can find the set of coefficients a_i and the unit vector **e** by again minimizing a sum-squared error criterion J_1.

$$\begin{aligned} J_1(a_1, \ldots, a_n, \mathbf{e}) &= \sum_{i=1}^{N} ||(\mathbf{m} + a_i\mathbf{e}) - \mathbf{x}_i||^2 && (3.84a)\\ &= \sum_{i=1}^{N} ||a_i\mathbf{e} - (\mathbf{x}_i - \mathbf{m})||^2 && (3.84b)\\ &= \sum_{i=1}^{N} a_i^2||\mathbf{e}||^2 - 2\sum_{i=1}^{N} a_i\mathbf{e}^T(\mathbf{x}_i - \mathbf{m}) + \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 && (3.84c) \end{aligned}$$

Eq. 3.84c is differentiated with respect to a_i, and the result is equated to zero to obtain

$$a_i = \mathbf{e}^T(\mathbf{x}_i - \mathbf{m}) \tag{3.85}$$

In words, Eq. 3.85 shows that a_i is the projection of the i^{th} mean-subtracted vector $(\mathbf{x}_i - \mathbf{m})$ onto the unit vector **e**. The ensemble of projections onto this new basis vector are linear combinations of the variables that constitute the original observation vectors. This concept is illustrated in Figure 3.16.

The goal now is to find the "best" direction for **e** by substituting the solution for a_i into Eq. 3.84c. J_1 can now be rearranged:

$$\begin{aligned} J_1(\mathbf{e}) &= \sum_{i=1}^{N} a_i^2 - 2\sum_{i=1}^{N} a_i^2 + \sum_{i=1}^{N} ||(\mathbf{x}_i - \mathbf{m})||^2 && (3.86a)\\ &= -\sum_{i=1}^{N}[\mathbf{e}^T(\mathbf{x}_i - \mathbf{m})]^2 + \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 \\ &= -\sum_{i=1}^{N} \mathbf{e}^T(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T\mathbf{e} + \sum_{i=1}^{N} ||(\mathbf{x}_i - \mathbf{m})||^2 && (3.86b) \end{aligned}$$

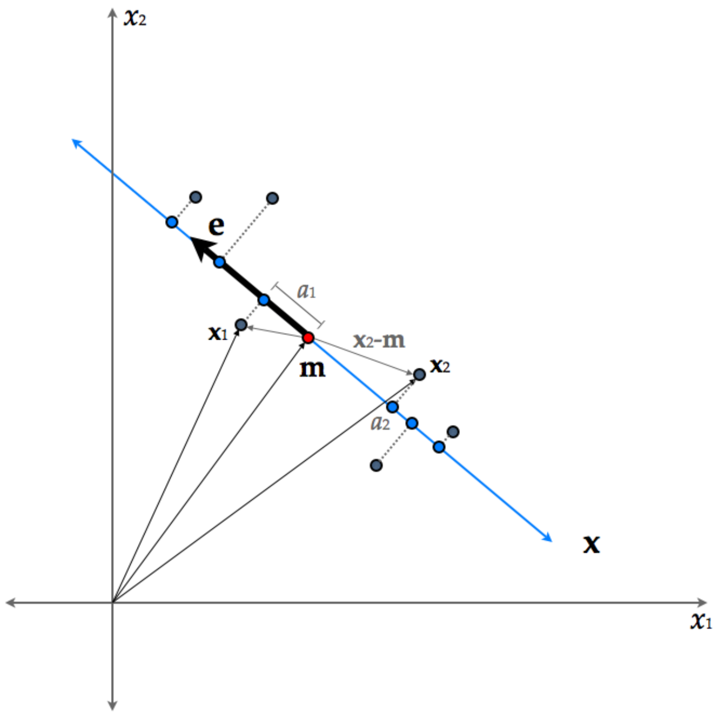


Figure 3.16: Projection of a 2-dimensional data set onto a line – the vectors in the mean-subtracted data set $(\mathbf{x}_i - \mathbf{m})$ are projected onto the line $\mathbf{x}$ in the direction of the unit vector $\mathbf{e}$ that passes through the sample mean. The new set of coefficients a_i correspond to the distance between the projections and $\mathbf{m}$. The criterion function J_0 is used to minimize the sum-squared error between the set of vectors $\mathbf{x}_i$ and the line $\mathbf{x}$, providing the least-squares solution for the PC transform.

Eq. 3.86b provides an important realization and is the key to solving PCA. We can define the scatter matrix $\mathbf{S}$:

$$\mathbf{S} = \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T \tag{3.87}$$

which is just the sample covariance matrix C of the mean-subtracted data multiplied by the scale factor $(N-1)$. This allows J_1 to be rewritten:

$$J_1(\mathbf{e}) = -\mathbf{e}^T \mathbf{S} \mathbf{e} + \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{m}||^2 \tag{3.88}$$

Thus, J_1 is minimized by maximizing the quadratic form $\mathbf{e}^T \mathbf{S} \mathbf{e}$ subject to the constraint $||\mathbf{e}|| = 1$. Referring to the derivation of the MACE filter in Section 3.2.2.2, this constrained minimization problem can be solved using the method of Lagrange multipliers [48] (see Appendix A.2). The Lagrangian function $L(\mathbf{e}; \lambda)$ is:

$$L(\mathbf{e}; \lambda) = \mathbf{e}^T \mathbf{S} \mathbf{e} - \lambda(\mathbf{e}^T \mathbf{e} - 1) \tag{3.89}$$

where λ is the Lagrange multiplier. The derivative of Eq. 3.89 with respect to $\mathbf{e}$ is

$$\frac{du}{d\mathbf{e}} = 2\mathbf{S}\mathbf{e} - 2\lambda \mathbf{e} \tag{3.90}$$

which is set equal to zero to yield the eigenvector equation:

$$\boxed{\mathbf{S}\mathbf{e} = \lambda \mathbf{e}} \tag{3.91}$$

This result shows that the dominant eigenvector of the scatter matrix, corresponding to the largest eigenvalue, is the best line onto which the original mean-subtracted vectors should be projected in a least-squares sense. This eigenvector $\mathbf{e}_1$ is referred to as the first *principal component vector*. Correspondingly, the projections a_i of the mean-subtracted vectors $(\mathbf{x}_i - \mathbf{m})$ onto $\mathbf{e}_1$ comprise the first *principal components* of the data set.

This result can be extended by projecting the original data onto the full set of d unit vectors $(\mathbf{e}_1, \ldots, \mathbf{e}_d)$

$$\mathbf{x} = \mathbf{m} + \sum_{j=1}^{d} a_j \mathbf{e}_j \tag{3.92}$$

The new criterion function J_2 can be written as:

$$J_2 = \sum_{i=1}^{N} ||(\mathbf{m} + \sum_{j=1}^{d} a_{ij}\mathbf{e}_j) - \mathbf{x}_i||^2 \tag{3.93}$$

In similar fashion, it can be shown that J_2 is minimized by projecting each mean-subtracted vector $(\mathbf{x}_i - \mathbf{m})$ on the d eigenvectors $\mathbf{e}_j$ of the scatter matrix $\mathbf{S}$. This projection is the principal component transform; it is illustrated for $d = 2$ dimensions in Figure 3.17. Restated for clarity, the set of eigenvectors $(\mathbf{e}_1, \ldots, \mathbf{e}_d)$ are the principal component vectors that define the basis for PCA, and the projections a_{ij} are the j^{th} principal components of the i^{th} original vector $\mathbf{x}_i$.

3.3.1.2 Revisiting the Objectives of PCA

Thus far, it has been shown that the PC transform effectively represents each original vector $\mathbf{x}_i$ by its principal components a_i in a new coordinate system with axes $\mathbf{e}_j$ that have been rotated to pass through the translated origin $\mathbf{m}$ in the directions of maximum variance (with respect to the original coordinate system). Now the objectives of PCA can be more readily discussed and understood. PCA diagonalizes the covariance matrix, so that the covariance between separate variables is zero. An observation vector that has been projected onto this basis contains realizations of variables that are now mutually uncorrelated. This point is crucial for using PCA to reduce the dimensionality of the data. *Axes in the new basis (eigenvectors) that point in directions with correspondingly large variances (eigenvalues) are assumed to signify the most important and interesting dynamics of the data.* Under this assumption, the eigenvalues can be used to retain a specified percentage of variability by using only a subset of the dominant eigenvectors.

It is common for many variables in a data set to be strongly correlated, and therefore, much of the variance should be captured in the first few PCs. While it is valid for many applications to assume that important characteristics of the data set can be represented by variance, it is necessary to consider the possibility of removing poten-

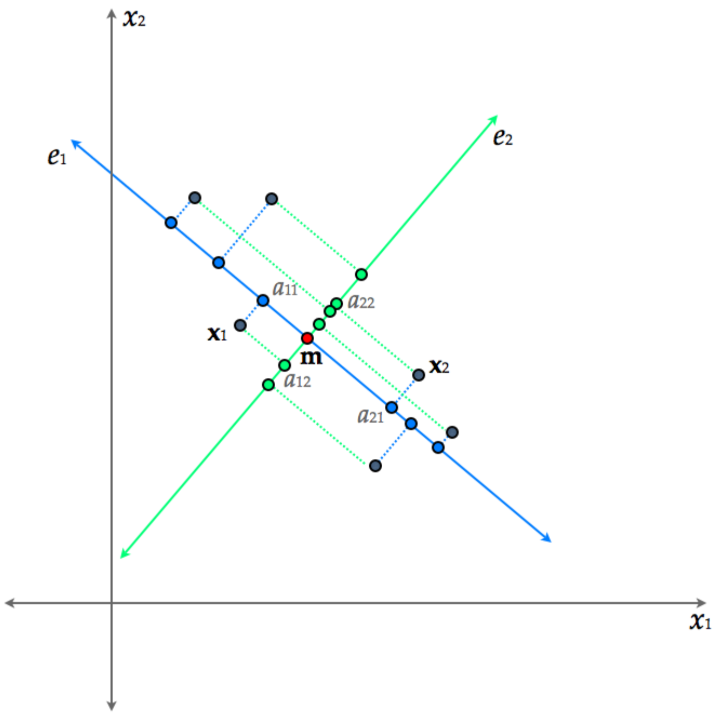


Figure 3.17: PC transform illustration in $d = 2$ dimensions – the original set of mean-subtracted vectors $(\mathbf{x}_i - \mathbf{m})$ are projected onto the eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ of the scatter matrix $\mathbf{S}$, which pass thought the sample mean. a_{ij} are the j^{th} principal components of $\mathbf{x}_i$. Note that the variance of the first PCs a_{1i} is significantly larger than that of the second PCs a_{2i}.

tially useful, but typically hidden, details. For example, Schott [44] notes that unique information can be found in the thermal spectral bands of Landsat TM multispectral data, which is often characterized by low variance. This information may be buried in the noise from other spectral bands that dominates the lower PC vectors (principal component vectors with correspondingly small eigenvalues).

Also worth noting is the distinction between using a linear transform with a deterministic basis, such as the DFT, and one with a random basis, such as PCA, for improving data interpretation. The projection of a set of vectors onto the basis functions of the DFT can easily be analyzed using plots of the real and imaginary or magnitude and phase components. A transformed vector is represented by plotting its projection against the corresponding spatial frequencies of the complex sinusoids. If a vector is instead projected onto the derived PC basis (also referred to as *eigenfunctions*), a plot constructed in similar fashion would show the vector's principal components corresponding to each of the principal component vectors.

Both representations illustrate the linear combination of the new basis necessary to reconstruct (or to partially reconstruct if the dimensionality has been reduced) the original vector. However, the DFT representation is arguably more intuitive, as the eigenfunctions in PCA are characterized by the variances expressed by the corresponding eigenvalues, which are properties of the data rather than by predefined spatial frequencies. A more common approach to interpreting PCA is to plot the PCs for some combination of PC vectors to reveal suspect observations in the data [25].

3.3.1.3 Calculating PCA

Performing PCA is, in fact, quite simple with the support of eigenvector decomposition routines found in most numerical packages. There are two cases to consider. The first scenario, which is arguably the more common of the two, is encountered when the number of variables d is less than the number of observations N. For example, many remote sensing applications require the use of PCA to decorrelate spectral bands in multispectral and hyperspectral image data. The algorithm to compute the PCs for this first case can be condensed into just five steps.

PCA Case 1: Variables $d <$ Observations N

1. Organize the data set into a $d \times N$ matrix $\mathbf{X}$ such that its columns consist of d-element column vectors $\mathbf{x}_i$. The d rows and N columns correspond to the number of variables (or features) and the number of observations (or feature vectors), respectively.

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{d,1} & x_{d,2} & \cdots & x_{d,N} \end{bmatrix}$$

2. Construct a new $d \times N$ matrix $\tilde{\mathbf{X}}$ by subtracting the mean-observation vector $\mathbf{m}$ from each column $\mathbf{x}_i$ in $\mathbf{X}$.

$$\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{m} = \begin{bmatrix} \mathbf{x}_1 - \mathbf{m} & \mathbf{x}_2 - \mathbf{m} & \cdots & \mathbf{x}_N - \mathbf{m} \end{bmatrix}$$

3. Calculate the $d \times d$ scatter matrix $\mathbf{S}$ from the mean-subtracted data matrix $\tilde{\mathbf{X}}$.

$$\mathbf{S} = \sum_{i=1}^{N} [\mathbf{x}_i - \mathbf{m}] [\mathbf{x}_i - \mathbf{m}]^T = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T$$

4. Compute the eigenvectors $\mathbf{e}_j$ of $\mathbf{S}$, and construct a $d \times d'$ matrix $\mathbf{E}$ (where $d' \leq d$) by using each $\mathbf{e}_j$ as a column in order of decreasing eigenvalue δ_j. The number of eigenvectors to retain d' for a particular application can be determined by examining the variance along each new direction, which is specified by the corresponding eigenvalue.

$$\mathbf{E} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_{d'} \end{bmatrix}$$

5. Project the mean-subtracted data matrix $\tilde{\mathbf{X}}$ onto the PC directions $\mathbf{e}_j$ to obtain the

$d' \times N$ matrix $\mathbf{A}$ of principal components.

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{d',1} & a_{d',2} & \cdots & a_{d',N} \end{bmatrix} = \mathbf{E}^T \tilde{\mathbf{X}}$$

The inverse transform is trivial, because the matrix of eigenvectors $\mathbf{E}$ is an orthnormal matrix. Thus, $(\mathbf{E}^T)^{-1} = (\mathbf{E}^T)^T = \mathbf{E}$ and $\mathbf{X} = \mathbf{EA}$. In words, the principal components provide the weights for the linear combination of principal directions (the eigenvectors) that reconstructs, or partially reconstructs (after data reduction), the original mean-subtracted data set.

Intuitively, the second case to consider arises when the number of variables d exceeds the number of observations N. This is commonly referred to as the *small sample size* (SSS) problem [13]. In 1991, Turk and Pentland [49] published work on *Eigenfaces* for face recognition using a PCA subspace of face-shaped eigenfunctions to classify unknown input face images. In their approach, the eigenfaces (PC basis functions) were derived from training face images with the objective to decorrelate pixels. To do so requires computation of the eigenvectors of a $d \times d$ scatter matrix S, which often presents an intractable task (e.g., 64×64 pixel training images produce a 4096×4096 scatter matrix). The following algorithm, which utilizes the procedure in [13], was implemented to overcome this obstacle.

PCA Case 2: Variables $d >$ Observations N

1. Organize the data set into a $d \times N$ matrix $\mathbf{X}$ such that its columns consist of d-element column vectors $\mathbf{x}_i$. The d rows and N columns correspond to the number of variables (or features) and the number of observations (or feature vectors), respectively.

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{d,1} & x_{d,2} & \cdots & x_{d,N} \end{bmatrix}$$

2. Construct a new $d \times N$ matrix $\tilde{\mathbf{X}}$ by subtracting the mean-observation vector $\mathbf{m}$ from each column $\mathbf{x}_i$ in $\mathbf{X}$.

$$\tilde{\mathbf{X}} = \mathbf{X} - \mathbf{m} = \left[\begin{array}{cccc} \mathbf{x}_1 - \mathbf{m} & \mathbf{x}_2 - \mathbf{m} & \cdots & \mathbf{x}_N - \mathbf{m} \end{array} \right]$$

3. Calculate the $N \times N$ Gram matrix $\mathbf{G}$ using the mean-subtracted data matrix $\tilde{\mathbf{X}}$.

$$\mathbf{G} = \tilde{\mathbf{X}}^T \tilde{\mathbf{X}}$$

4. Compute the eigenvectors $\mathbf{k}_j$ of $\mathbf{G}$, and construct an $N \times N$ matrix $\mathbf{K}$ by using each $\mathbf{k}_j$ as a column in order of decreasing eigenvalue δ_j.

$$\mathbf{K} = \left[\begin{array}{cccc} \mathbf{k}_1 & \mathbf{k}_2 & \cdots & \mathbf{k}_N \end{array} \right]$$

5. Calculate the $d \times N'$ matrix $\mathbf{E}'$ of eigenvectors $\mathbf{e}_j$ (where $N' \leq N$) by projecting the columns of $\mathbf{K}$ onto the rows of $\tilde{\mathbf{X}}$, i.e., by using each vector $\mathbf{k}_j$ to determine linear combinations of the mean-subtracted data. The vectors $\mathbf{e}_j$ are the dominant eigenvectors of the scatter matrix $\mathbf{S}$, and their corresponding eigenvalues δ_j are those of both $\mathbf{G}$ and $\mathbf{S}$. The number of eigenvectors to retain N' for a particular application can be derived by examining the variance along each new direction, which is specified by the corresponding eigenvalues. It should be noted that some degree of data reduction is unavoidable for case 2 ($d > N$).

$$\mathbf{E}' = \tilde{\mathbf{X}}\mathbf{K} = \left[\begin{array}{cccc} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_{N'} \end{array} \right]$$

6. Project $\tilde{\mathbf{X}}$ onto the PC directions $\mathbf{e}_j$ to obtain the $N' \times N$ matrix $\mathbf{A}$ of principal components.

$$\mathbf{A} = \left[\begin{array}{cccc} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N',1} & a_{N',2} & \cdots & a_{N',N} \end{array} \right] = \mathbf{E'}^T \tilde{\mathbf{X}}$$

3.3.2 Linear Discriminant Analysis

To begin this section on *linear discriminant analysis* (LDA), it makes sense to refer back to the objectives of PCA. The PC transform diagonalizes the scatter matrix of the "full" data set, redistributing the variance such that the derived basis vectors point in directions of maximum variability. Recall that the projection of the original data onto the PC basis (or vector subspace after removing eigenvectors with correspondingly small eigenvalues) is used for purposes of both data interpretation and data reduction. While certainly a powerful tool for many applications, the PC basis is not necessarily useful for discriminating classes.

In 1936, Fisher [12] published his famous work on discriminating populations in classification problems, in which he showed that the criterion function $\mathbf{J}(\mathbf{w})$ in Eq. 3.94 can be maximized to best separate two classes of data that have been projected onto the linear discriminant vector $\mathbf{w}$:

$$J(\mathbf{w}) = \frac{(m_1 - m_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{3.94}$$

This is the well-known *Fisher criterion*, which is the cornerstone for the development of LDA. In words, the Fisher criterion is the ratio of the between-class scatter $(m_1 - m_2)^2$, which indicates separation of the projected-class means $\mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)$, to the within-class scatter $\sigma_1^2 + \sigma_2^2$, which accounts for the variance of the projected classes σ_1 and σ_2. LDA is the process of determining the vector (or vector subspace) that maximizes this criterion and the subsequent projection of the original data onto this space.

A conceptual explanation of LDA is often facilitated by considering its relation to PCA for a couple of reasons. Like PCA, LDA vectors are derived from the random variables of the original data set, and can therefore be considered random. Also, the original observation vectors are transformed to LDA vectors by first translating the origin of the coordinate system. Perhaps the most significant correspondence is that both transforms are derived to satisfy criterion functions defined in terms of scatter matrices. However, PCA and LDA typically produce very different sets of eigenfunctions. While the PC basis vectors point in directions of maximum scatter for the overall data set, the LDA directions point in the directions of maximum class discrimination assuming that the classes are mutually exclusive. It should be stressed that the c-1 vectors produced using LDA almost never span the original vector space (c is the number

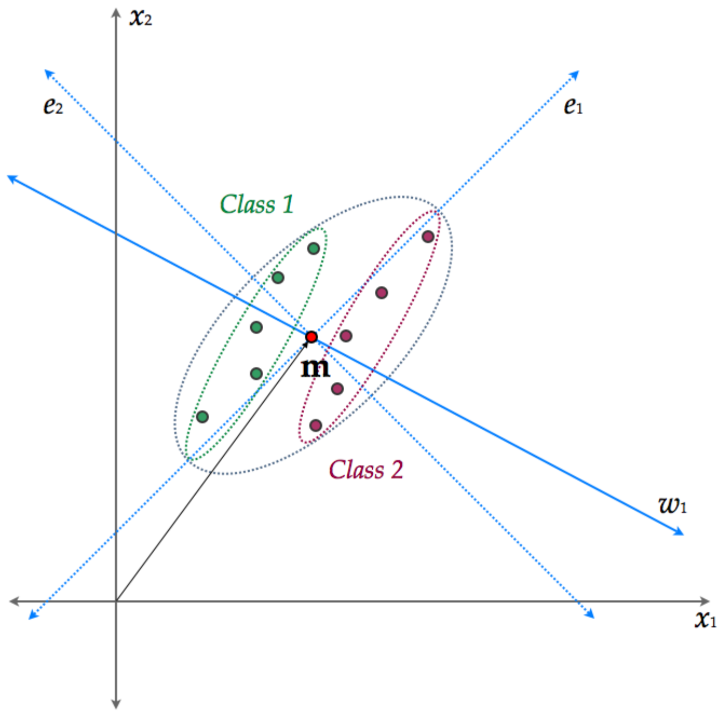


Figure 3.18: LDA vs. PCA in 2-dimensions – x_1 and x_2 are variables defining the original basis, e_1 and e_2 are linear combinations of x_1 and x_2 that define the basis of the PC transform, and w_1 is a linear combination of x_1 and x_2 that defines the vector subspace of LDA. The coordinate system has been translated to the mean observation vector **m**. For LDA, the number of variables has been reduced to $c - 1$ such that the lone axis e_1 points in the direction of maximum class discrimination.

of classes). There are also cases in which the LDA directions are not orthogonal (with respect to the original vector space). Thus, LDA should be considered a projection onto a vector subspace, rather than a change of basis like the DFT or PCA (prior to data reduction). A comparison of LDA to PCA is illustrated for a 2-dimensional, 2-class case in Figure 3.18.

3.3.2.1 The LDA Vector Subspace

Assume that we are given a set of N, d-dimensional vectors $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$, with N_k vectors belonging to subset D_k, where k denotes the class label and $0 \leq k \leq c$. The general multiple discriminant case of the LDA algorithm for the c-class problem [9] can be formulated by revisiting the Fisher criterion in Eq. 3.94. Choosing $\mathbf{w}$ to maximize $J(\cdot)$ yields a vector that maximizes the ratio of the between-class scatter $(\mathbf{m}_1 - \mathbf{m}_2)^2$ and the within-class scatter $\sigma_1^2 + \sigma_2^2$ of the projected data. This idea can be extended by defining the criterion function in terms of the between-class matrix $\mathbf{S}_B$ and within-class scatter matrix $\mathbf{S}_W$ for an arbitrary number of classes.

For multiple classes, the within-class scatter matrix $\mathbf{S}_W$ is just the sum of the scatter matrices of the individual classes:

$$\mathbf{S}_W = \sum_{k=1}^{c} \mathbf{S}_k \tag{3.95}$$

where $\mathbf{S}_k$ is:

$$\mathbf{S}_k = \sum_{i=1}^{N_k} (\mathbf{x}_i - \mathbf{m}_k)(\mathbf{x}_i - \mathbf{m}_k)^T \tag{3.96}$$

and $\mathbf{m}_k$ is just the mean of the k^{th}-class

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{x}_i \tag{3.97}$$

Likewise, the between-class scatter matrix $\mathbf{S}_B$ is a weighted sum of the separation of class means $\mathbf{m}_k$ to the total mean $\mathbf{m}_t$:

$$\mathbf{S}_B = \sum_{k=1}^{c} N_k (\mathbf{m}_k - \mathbf{m}_t)(\mathbf{m}_k - \mathbf{m}_t)^T \tag{3.98}$$

It is worth noting that the total scatter matrix (of the entire data set) $\mathbf{S}_T$ is just the sum of $\mathbf{S}_W$ and $\mathbf{S}_B$.

The numerator of the Fisher criterion can be expressed in terms of the projected class means:

$$\begin{aligned}(m_1 - m_2)^2 &= \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T\mathbf{w} && (3.99a)\\ &= \mathbf{w}^T\mathbf{S}_B\mathbf{w} && (3.99b)\end{aligned}$$

For multiple discriminants, the numerator is just $\mathbf{W}^T\mathbf{S}_B\mathbf{W}$ where $\mathbf{W}$ is a matrix specifying the vector subspace of discriminating directions. Similarly, we can express the denominator of the Fisher criterion in terms of the projected class variances:

$$\begin{aligned}\sigma_1^2 + \sigma_2^2 &= \mathbf{w}^T\mathbf{C}_1\mathbf{w} + \mathbf{w}^T\mathbf{C}_1\mathbf{w} && (3.100a)\\ &= \mathbf{w}^T(\mathbf{C}_1 + \mathbf{C}_2)\mathbf{w} && (3.100b)\end{aligned}$$

Scale factors have no effect on the direction of $\mathbf{w}$, so we can replace $(\mathbf{C}_1 + \mathbf{C}_2)$ by $\mathbf{S}_W$ [37]. The denominator then becomes $\mathbf{W}^T\mathbf{S}_W\mathbf{W}$. Eq. 3.101 is the Fisher criterion, in terms of $\mathbf{S}_B$ and $\mathbf{S}_W$, as an explicit function of $\mathbf{W}$

$$J(\mathbf{W}) = \frac{|\mathbf{W}^T\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^T\mathbf{S}_W\mathbf{W}|} \tag{3.101}$$

where the determinants of the numerator and denominator are used as scalar measures of scatter [9].

Eq. 3.101 is a Rayleigh quotient (see Appendix A.1.2), and maximizing $J(\mathbf{W})$ with respect to $\mathbf{W}$ yields a generalized eigenvalue problem:

$$\mathbf{S}_B\mathbf{W} = \mathbf{S}_W\mathbf{W}J(\mathbf{W}) \tag{3.102}$$

In the words, the columns of $\mathbf{W}$, which define the LDA vector subspace, are the generalized eigenvectors that simultaneously diagonalize $\mathbf{S}_B$ and $\mathbf{S}_W$. $J(\mathbf{W})$ is just a diagonal matrix $\mathbf{\Lambda}$ containing the corresponding eigenvalues. The concept of LDA as a *simultaneous diagonalization* problem (Appendix A.1.1) will be discussed in the following section.

For the case when $\mathbf{S}_W$ is nonsingular, Eq. 3.102 can be rewritten as a conventional eigenproblem:

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{W} = \mathbf{W}\mathbf{\Lambda} \tag{3.103}$$

Unfortunately, $\mathbf{S}_W$ is often singular for many applications, which is a direct result of using data sets in which the number of variables exceeds the number of observations. In the following section, we examine a method to overcome this obstacle for LDA computation.

3.3.2.2 Calculating LDA

In practice, the two cases considered in PCA computation are also relevant for LDA. While the Case 2 scenario (more variables d than observations N) is becoming increasingly popular in imaging applications, case 1 (more observations N than variables d) is reviewed here for completeness and comparison. The algorithm to compute the LDA subspace for the case in which $d < N$ can be condensed into seven steps.

LDA Case 1: Variables d < Observations N

1. Organize the data into c, $d \times N_k$ matrices $\mathbf{X}_k$, where c is the number of classes, d is the number of pixels, and N_k is the number of observations in the k^{th} class. Each column of a matrix $\mathbf{X}_k$ is an observation vector from class k.

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{x}_{k_1} & \mathbf{x}_{k_2} & \cdots & \mathbf{x}_{k_{N_k}} \end{bmatrix} = \begin{bmatrix} x_{k_{1,1}} & x_{k_{1,2}} & \cdots & x_{k_{1,N_k}} \\ x_{k_{2,1}} & x_{k_{2,2}} & \cdots & x_{k_{2,N_k}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k_{d,1}} & x_{k_{d,2}} & \cdots & x_{k_{d,N_k}} \end{bmatrix}$$

2. Calculate the total mean vector $\mathbf{m}_t$ and the class-mean vector $\mathbf{m}_k$. N is the total number of observations.

$$\mathbf{m}_t = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i = \begin{bmatrix} \frac{1}{N}\sum_{i=1}^{N} x_{1,i} & \frac{1}{N}\sum_{i=1}^{N} x_{2,i} & \cdots & \frac{1}{N}\sum_{i=1}^{N} x_{d,i} \end{bmatrix}^T$$

$$\mathbf{m}_k = \frac{1}{N_k}\sum_{i=1}^{N_k}\mathbf{x}_{k_i} = \begin{bmatrix} \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{1,i}} & \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{2,i}} & \cdots & \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{d,i}} \end{bmatrix}^T$$

3. Generate c, $d \times N_i$ matrices $\tilde{\mathbf{X}}_k$ of class-mean-subtracted observation vectors.

$$\tilde{\mathbf{X}}_k = \begin{bmatrix} x_{k_{1,1}} - m_{k_1} & x_{k_{1,2}} - m_{k_1} & \cdots & x_{k_{1,N_k}} - m_{k_1} \\ x_{k_{2,1}} - m_{k_2} & x_{k_{2,2}} - m_{k_2} & \cdots & x_{k_{2,N_k}} - m_{k_2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k_{d,1}} - m_{k_d} & x_{k_{d,2}} - m_{k_d} & \cdots & x_{k_{d,N_k}} - m_{k_d} \end{bmatrix}$$

4. Calculate the $d \times d$ between-class scatter matrix $\mathbf{S}_B$ by weighting the outer product of the mean-subtracted class-mean vectors (the columns of $d \times c$ matrix $\tilde{\mathbf{M}}$) by the number of observations in the k^{th} class N_k.

$$\begin{aligned} \mathbf{S}_B &= \tilde{\mathbf{M}}\mathbf{n}\tilde{\mathbf{M}}^T \\ &= \begin{bmatrix} \mathbf{m}_1 - \mathbf{m}_t & \cdots & \mathbf{m}_c - \mathbf{m}_t \end{bmatrix} \begin{bmatrix} N_1 \\ N_2 \\ \vdots \\ N_c \end{bmatrix} \begin{bmatrix} \mathbf{m}_1 - \mathbf{m}_t & \cdots & \mathbf{m}_c - \mathbf{m}_t \end{bmatrix}^T \end{aligned}$$

5. Calculate the $d \times d$ within-class scatter matrix $\mathbf{Sw}$ by summing the individual class scatter matrices $\mathbf{S}_k$. Each $\mathbf{S}_k$ is just the outer product of the class-mean-subtracted observation vectors, the columns of $\tilde{\mathbf{X}}_k$.

$$\mathbf{S}_W = \sum_{k=1}^{c} \mathbf{S}_k = \sum_{k=1}^{c} \tilde{\mathbf{X}}_k \tilde{\mathbf{X}}_k^T$$

6. Compute the eigenvectors $\mathbf{w}_j$ of the $d \times d$ matrix $\mathbf{S}_W^{-1}\mathbf{S}_B$, and construct the $d \times (c-1)$ matrix $\mathbf{W}$ by using each $\mathbf{w}_j$ as a column in order of decreasing eigenvalue δ_j. The dominant eigenvectors define the LDA subspace.

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_{c-1} \end{bmatrix}$$

7. Project the matrices $\tilde{\mathbf{X}}_k$ onto the LDA vectors $\mathbf{w}_j$ to obtain the $(c-1) \times N_k$ matrices $\mathbf{A}_k$, which contain the observation vectors from the k^{th} class represented in a space designed to minimize the within-class scatter and maximize its separation

from all other classes.

$$\mathbf{A}_k = \mathbf{W}^T \tilde{\mathbf{X}}_k$$

The inverse LDA transform is not as trivial as the inverse PC transform. $\mathbf{W}$ is no longer an orthogonal matrix, and is in fact only square if $d = (c-1)$. The $(c-1)$ column vectors of $\mathbf{W}$, while composed of d elements, do not span the full d-dimensional space. We can calculate the *pseudoinverse* of $\mathbf{W}$ as $\mathbf{W}^\dagger = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$. The least-squares approximation to the inverse transformation is $\hat{\tilde{\mathbf{X}}}_k = (\mathbf{W}^T)^\dagger \mathbf{A}_k$.

The second case for computing LDA arises when the number of variables exceeds the number of observations. In this case, both the within-class scatter matrix $\mathbf{S}_W$ and the between-class scatter matrix $\mathbf{S}_B$ are singular. For many years, this obstacle delayed the development of a direct LDA solution for high-dimensional data sets (e.g., in imaging applications).

Perhaps the most prevalent usage of this case of LDA is in face recognition applications. In 1997, Belhumeu, et al., presented their work on *Fisherfaces*, a face recognition algorithm that uses LDA to generate a vector subspace for class discrimination [1]. Their proposed method addressed the small sample size problem by using PCA as a preprocessing step to reduce the dimensionality of the data set. Thus, the full data set is first projected onto a PCA vector subspace such that the approximate within-class scatter matrix $\hat{\mathbf{S}}_W$ is now nondegenerate. While this method exhibits improved recognition rates over the eigenface technique [49] presented in 3.3.1.3, it has been noted that that reducing dimensionality prior to computing LDA can remove information vital to class discrimination [23].

In 2001, Yu and Yang [54] published an algorithm to directly compute LDA for high-dimensional data sets, which they referred to as Direct-LDA. This method was derived by recognizing that the *null space* of the within-class scatter matrix contains the most discriminative information, and that LDA can be manipulated as a simultaneous diagonalization problem (Appendix A.1.1). Yu and Yang's Direct LDA approach has been widely accepted in the face recognition field, and it is the foundation for the following Case 2 algorithm.

LDA Case 2: Variables $d >$ Observations N

1. Organize the data into c, $d \times N_k$ matrices $\mathbf{X}_k$, where c is the number of classes,

d is the number of total pixels, and N_k is the number of observations in the k^{th} class. Each column of a matrix $\mathbf{X}_k$ is an observation vector from class k.

$$\mathbf{X}_k = \begin{bmatrix} \mathbf{x}_{k_1} & \mathbf{x}_{k_2} & \cdots & \mathbf{x}_{k_{N_k}} \end{bmatrix} = \begin{bmatrix} x_{k_{1,1}} & x_{k_{1,2}} & \cdots & x_{k_{1,N_k}} \\ x_{k_{2,1}} & x_{k_{2,2}} & \cdots & x_{k_{2,N_k}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k_{d,1}} & x_{k_{d,2}} & \cdots & x_{k_{d,N_k}} \end{bmatrix}$$

2. Calculate the total mean vector $\mathbf{m}_t$ and the class-mean vector $\mathbf{m}_k$. N is the total number of observations.

$$\mathbf{m}_t = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i = \begin{bmatrix} \frac{1}{N}\sum_{i=1}^{N} x_{1,i} & \frac{1}{N}\sum_{i=1}^{N} x_{2,i} & \cdots & \frac{1}{N}\sum_{i=1}^{N} x_{d,i} \end{bmatrix}^T$$
$$\mathbf{m}_k = \frac{1}{N_k}\sum_{i=1}^{N_k}\mathbf{x}_{k_i} = \begin{bmatrix} \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{1,i}} & \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{2,i}} & \cdots & \frac{1}{N_k}\sum_{i=1}^{N_k} x_{k_{d,i}} \end{bmatrix}^T$$

3. Generate c, $d \times N_i$ matrices $\tilde{\mathbf{X}}_k$ of class-mean-subtracted observation vectors.

$$\tilde{\mathbf{X}}_k = \begin{bmatrix} x_{k_{1,1}} - m_{k_1} & x_{k_{1,2}} - m_{k_1} & \cdots & x_{k_{1,N_k}} - m_{k_1} \\ x_{k_{2,1}} - m_{k_2} & x_{k_{2,2}} - m_{k_2} & \cdots & x_{k_{2,N_k}} - m_{k_2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k_{d,1}} - m_{k_d} & x_{k_{d,2}} - m_{k_d} & \cdots & x_{k_{d,N_k}} - m_{k_d} \end{bmatrix}$$

4. Calculate the $N \times N$ between-class Gram matrix $\mathbf{G}_B$ by weighting the inner product of the mean-subtracted class-mean vectors (the columns of $d \times N$ matrix $\tilde{\mathbf{M}}$) by the number of observations in the k^{th} class N_k.

$$\begin{aligned} \mathbf{G}_B &= \tilde{\mathbf{M}}^T\tilde{\mathbf{M}}\mathbf{n} \\ &= \begin{bmatrix} \mathbf{m}_1 - \mathbf{m}_t & \cdots & \mathbf{m}_c - \mathbf{m}_t \end{bmatrix}^T \begin{bmatrix} \mathbf{m}_1 - \mathbf{m}_t & \cdots & \mathbf{m}_c - \mathbf{m}_t \end{bmatrix} \begin{bmatrix} N_1 \\ N_2 \\ \vdots \\ N_c \end{bmatrix} \end{aligned}$$

5. Compute the eigenvectors $\mathbf{v}_j$ of $\mathbf{G_B}$, and construct a $c \times c$ matrix $\mathbf{V}$ by using each

$\mathbf{v}_j$ as a column in order of decreasing eigenvalue.

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_c \end{bmatrix}$$

6. Calculate the $d \times c'$ matrix $\mathbf{Y}$ of eigenvectors $\mathbf{y}_j$ (where $c' \leq c$) by projecting the columns of $\mathbf{V}$ onto the rows of $\tilde{\mathbf{M}}$, i.e., by using each vector $\mathbf{v}_j$ to determine linear combinations of the mean-subtracted class-mean vectors. The vectors $\mathbf{y}_j$ are the dominant eigenvectors of the between-class scatter matrix $\mathbf{S}_B$, and their corresponding eigenvalues δ_{B_j} are those of both $\mathbf{G}_B$ and $\mathbf{S}_B$. Only the c' eigenvectors $\mathbf{y}_j$ with nonzero eigenvalues should be retained:

$$\mathbf{Y} = \tilde{\mathbf{M}}\mathbf{V} = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_{c'} \end{bmatrix}$$

7. Calculate the $d \times c'$ matrix $\mathbf{Z}$ of scaled eigenvectors $\mathbf{z}_j$ by normalizing each $\mathbf{y}_j$ by the square root of its corresponding eigenvalue δ_{B_j}. The vector subspace specified by the columns of $\mathbf{Z}$ unitizes $\mathbf{S}_B$, i.e., diagonalizes $\mathbf{S}_B$ to the identity matrix $\mathbf{I}$.

$$\begin{aligned} \mathbf{Z} &= \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_{c'} \end{bmatrix} \\ &= \mathbf{Y}\boldsymbol{\Delta}_B^{-0.5} = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_{c'} \end{bmatrix} \begin{bmatrix} \delta_{B_1} & 0 & \cdots & 0 \\ 0 & \delta_{B_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \delta_{B_{c'}} \end{bmatrix} \end{aligned}$$

8. Calculate the $c' \times c'$ matrix $\mathbf{Z}^T\mathbf{S}_W\mathbf{Z}$, which corresponds to the within-class scatter matrix $\mathbf{S}_W$ represented in the column space of $\mathbf{Z}$. $\mathbf{Z}^T\mathbf{S}_W\mathbf{Z}$ can be expanded to $\mathbf{Z}^T\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\mathbf{Z}$, where $\tilde{\mathbf{X}}$ is a $d \times N$ matrix with columns of all class-mean-subtracted observation vectors.

$$\begin{aligned} \mathbf{Z}^T\mathbf{S}_W\mathbf{Z} &= (\mathbf{Z}^T\tilde{\mathbf{X}})(\tilde{\mathbf{X}}^T\mathbf{T}) \\ &= \left(\mathbf{Z}^T \begin{bmatrix} \mathbf{x}_1 - \mathbf{m}_{k_1} & \mathbf{x}_2 - \mathbf{m}_{k_2} & \cdots & \mathbf{x}_N - \mathbf{m}_{k_N} \end{bmatrix}\right) \\ &\quad \left(\begin{bmatrix} \mathbf{x}_1 - \mathbf{m}_{k_1} & \mathbf{x}_2 - \mathbf{m}_{k_2} & \cdots & \mathbf{x}_N - \mathbf{m}_{k_N} \end{bmatrix}^T \mathbf{Z}\right) \end{aligned}$$

9. Compute the eigenvectors $\mathbf{u}_i$ of $\mathbf{Z}^T\mathbf{S}_W\mathbf{Z}$, and construct a $c' \times c'$ matrix $\mathbf{U}$ by using each $\mathbf{u}_i$ as a column in order of decreasing eigenvalue. The vectors $\mathbf{u}_i$ specify linear combinations of the column vectors of $\mathbf{Z}$; this operation effectively defines a new subspace that diagonalizes the within-class scatter matrix $\mathbf{S}_W$ in the column space of $\mathbf{Z}$, which in turn diagonalizes the between-class scatter matrix $\mathbf{S}_B$ in its inherent design.
$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_{c'} \end{bmatrix}$$

10. Calculate the $d \times c'$ matrix $\mathbf{W}$ of column vectors $\mathbf{w}_j$, which specify the LDA vector subspace. The matrix $\mathbf{W}$ simultaneously diagonalizes both $\mathbf{S}_B$ and $\mathbf{S}_W$. If $c' \neq (c-1)$, then $columns > (c-1)$ should be discarded.
$$\mathbf{W} = \mathbf{ZU}$$

11. Project the matrices $\tilde{\mathbf{X}}_k$ onto the LDA vectors $\mathbf{w}_j$ to obtain the $(c-1) \times N_k$ matrices $\mathbf{A}_k$. $\mathbf{A}_k$ contains the observation vectors from the k^{th} class represented in a space designed to minimize the within-class scatter and maximize its separation from all other classes.
$$\mathbf{A}_k = \mathbf{W}^T\tilde{\mathbf{X}}_k$$

3.4 Probabilistic Networks

We now shift focus to discuss probabilistic networks, a topic whose importance to character recognition on the Archimedes Palimpsest will become clear in Chapter 4.

3.4.1 Modeling Uncertainty

In many pattern recognition applications, it is necessary to examine the inherent uncertainty at each stage in the decision-making process. Several components in the system will affect the classification decision, and thus the uncertainty in these components will affect the uncertainty in the classification decision. Evidence concerning the state of a system component can be used to estimate the state of other components, including the classification results.

Bayesian networks (also called *probabilistic networks*) represent a powerful tool for modeling the propagation of uncertainty in components (or variables) that share a causal relationship. These networks consist of sets of nodes that represent the variables in the system, each with its associated finite set of mutually exclusive states. Nodes are connected via directed edges, which describe the causality between variables.

Over the past decade, Bayesian networks have been used in a variety of applications such as medical diagnosis [38] and diagnosis-and-repair modules [42]. Probabilistic networks have even been used in character and word segmentation tasks. For example, Maragoudakis, et al., [36] demonstrated the advantages of learning SVM parameters via Bayesian networks in a handwritten character segmentation task. The following sections outline the underlying theory, which will serve as an introduction to the preliminary network model discussed in chapter 4.

3.4.2 Overview of Simple Network Design

Bayesian networks are *directed acyclic graphs* (DAGs), that is, a system of connected nodes in which no path exists from one node back to itself. DAGs can be *singly-connected* (also called *polytrees*), in which only one undirected path exists between any two nodes in the network, or *multiply-connected*, in which multiple paths can exist (Figure 3.19).

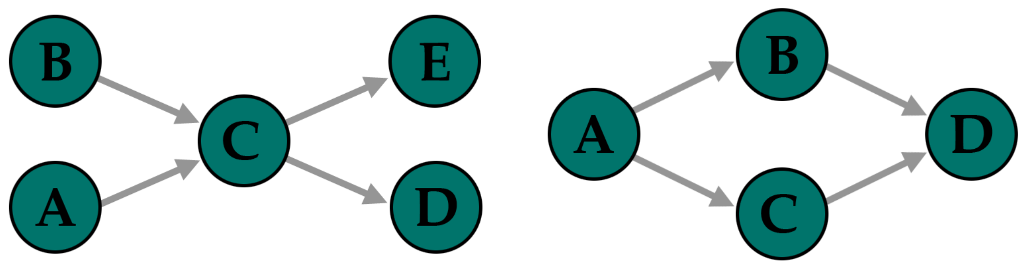


Figure 3.19: Example of a singly-connected (Left) and multiply-connected (Right) network.

Each node in the system contains an associated probability table which follows the network structure. For the remainder of this chapter, note that a variable in the network will be represented by an uppercase character and a given state of a variable by the corresponding lowercase character. In the multiply-connected network in Figure 3.19, no *directed edge* exists that leads to variable A. The probability table for variable A would consist only of $P(A)$, which (of course) contains each $P(a_i)$, where a_i represents the i^{th} mutually exclusive state of A. Variables B and C each have one directed edge denoting causal dependence on variable A. B and C are referred to as the children to parent node A, and therefore have associated conditional probability tables (CPTs) containing $P(B|A)$ and $P(C|A)$, respectively. Likewise, variable D is causally dependent on variables B and C, and its CPT contains the values from $P(D|B,C)$. It is apparent, even from this small-scale example, that the size of the CPTs in Bayesian networks become increasingly difficult to work with in realistic problem domains.

Before discussing the methods for relaying probabilistic information through a Bayesian network, it is important to review some of the relevant underlying mathematics in classical probability. The product rule for two events (variable states) a and b is:

$$P(a \cap b) = P(a|b)P(b) \tag{3.110a}$$

$$P(a \cap b) = P(b|a)P(a) \tag{3.110b}$$

where $P(a \cap b)$ is the joint probability of the occurrence of both events. Using these expressions, simple algebraic manipulation yields *Bayes' theorem* which relates the two conditional probabilities:

$$P(a|b) = \frac{P(b|a)P(a)}{P(b)} \tag{3.111}$$

When considering an additional event, the product rule can be expressed:

$$\begin{aligned} P(a \cap b \cap c) &= P(a|b \cap c)P(b \cap c) && \text{(3.112a)} \\ &= P(b|a \cap c)P(a \cap c) && \text{(3.112b)} \\ &= P(c|a \cap b)P(a \cap b) && \text{(3.112c)} \end{aligned}$$

It follows that Bayes' theorem can also be extended to cases in which an event is conditionally dependent on more than one event.

$$\begin{aligned} P(a|b \cap c) &= \frac{P(b|a \cap c)P(a|c)}{P(b|c)} && \text{(3.113a)} \\ &= \frac{P(c|b \cap a)P(a|b)}{P(c|b)} && \text{(3.113b)} \end{aligned}$$

Continuing in this progression to the case of n events, the product rule can be expanded to yield the general expression:

$$\begin{aligned} P(x_1 \cap \ldots \cap x_n) &= P(x_n|x_{n-1} \cap \ldots \cap x_1)P(x_{n-1} \cap \ldots \cap x_1) \\ &= P(x_n|x_{n-1}, \ldots, x_1)P(x_{n-1}|x_{n-2}, \ldots, x_1) \cdots P(x_2|x_1)P(x_1) \\ &= \prod_{i=1}^{n} P(x_i|x_{i-1}, \ldots, x_1) && \text{(3.114)} \end{aligned}$$

which is the *chain rule*; it shows that a joint probability of n events can be expressed as the product of conditional probability terms.

Of great importance in Bayesian networks is the property of *conditional independence*. If two events a and c are independent given event b then it follow that

$$P(a|b \cap c) = P(a|b) \tag{3.115}$$

In other words, c has no influence on the probability of a if b is known. Conditional independence plays its role in Bayesian networks in the concept of *d-separation*. Two variables A and B in a network are d-separated if an intermediate variable X exists and one of the following two constraints is satisfied.

1. The connection between A and B is serial or diverging and hard evidence has been applied to X
2. The connection between A and B is converging and evidence has not been applied to X or any descendants of X

For example, referring back to polytree in Figure 3.19, if evidence is applied to variable C (i.e., the state of C is known), then the variables A and D are d-separated because the first constraint is satisfied. Evidence from A to D is blocked by C, which indicates that A and D are conditionally independent given C:

$$P(D|C, A) = P(D|C) \tag{3.116}$$

The first constraint also shows that D and E are d-separated through a diverging connection. Contrarily, variables A and B are not d-separated in their converging connection, as the evidence has indeed been applied to the intermediate variable C. If evidence had been applied to B instead of C, then the second constraint would be satisfied (d-separation) between variables A and D and A and E.

Using the concept of d-seperation, the chain rule in Eq. 3.114 can now be modified for the full joint probability distribution represented in a Bayesian network. The node structure, defined by the directed edges, denotes the causal dependencies among the variables within the system. Recall that each node has a corresponding CPT whose elements consist of the conditional probabilities specified for given states in the associated parent variables. That is, a variable A would have a CPT containing the values from $P(A|Parents(A))$. Bayesian networks are constructed such that each node is conditionally independent (d-separated) of any preceding nodes in the hierarchical structure given evidence in its parents. Using this property and the chain rule in Eq. 3.114, an entry in the joint distribution represented in a Bayesian network can be computed using Eq. 3.117,

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|parents(X_i)) \tag{3.117}$$

where x_i denotes a state (event) of variable X_i, and $parents(X_i)$ denotes the states which correspond to the parents of variable X_i. This expression shows that queries

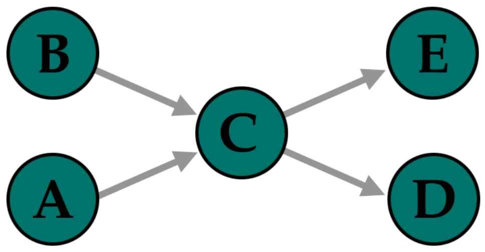


Figure 3.20: Singly-connected network (polytree) referenced in the discussion.

about the system, in the form of conditional probabilities, can be answered using the information described by the network. Therefore, the product rule in Eqs. 3.110a and 3.110b can be rewritten:

$$P(x|\mathbf{e}) = \frac{P(x \cap \mathbf{e})}{P(\mathbf{e})} = \alpha P(x \cap \mathbf{e}) \tag{3.118}$$

where $\mathbf{e}$ denotes any evidence states, and $\alpha = (P(\mathbf{e}))^{-1}$ is the normalization constant calculated by computing the posterior probability distribution $P(x|\mathbf{e})$ for all states x_i in the query variable X. Any query $P(x|\mathbf{e})$ about the domain modeled by the Bayesian network can now be answered by computing an entry from the full joint distribution $P(x \cap \mathbf{e})$. Eq. 3.117 shows this to be equivalent to multiplying relevant entries in the CPTs of the network. In the following section, methods are discussed for computing these types of queries in Bayesian networks.

3.4.3 Probabilistic Inference

Once the nodes in a Bayesian network have been initialized, probabilistic information can be transmitted through the network. Three types of variables exist in Bayesian networks.*Query variables* X are those under investigation given a set of *evidence variables* E with corresponding observed (known) states $\mathbf{e}$. The remaining variables are typically referred to as *hidden variables* Y.

Figure 3.20 shows the polytree from the previous section. Assume that it is necessary to calculate the posterior probability distribution for A given an observed event at C and D. In this scenario, A is the query variable, C and D are evidence variables, and

B and *E* are hidden variables. This query can be calculated from the joint distribution in Eq. 3.119.

$$P(A|c \cap d) = \alpha P(A, c, d) \tag{3.119}$$

The *marginalization* operation frequently used in inferencing algorithms is:

$$P(A) = \sum_B P(A \cap B) \tag{3.120}$$

In words, marginalization effectively removes a particular variable from a joint distribution by summing the entries corresponding to each of the states associated with the variable. Thus, Eq. 3.119 can be rewritten in terms of the full joint distribution by marginalizing over the hidden variables *B* and *E* in the query:

$$P(A|c \cap d) = \alpha \sum_b \sum_e P(A, B, c, d, E) \tag{3.121}$$

As shown in the previous section, the joint distribution can be expressed as a product of conditional probabilities (Eq. 3.122a). Eq. 3.122b illustrates an equivalent, less computationally intensive solution to the query.

$$\begin{aligned} P(A|c \cap d) &= \alpha \sum_b \sum_e P(A)P(b)P(c|a,b)P(d|c)P(e|c) && \text{(3.122a)} \\ &= \alpha P(A) \sum_b P(b)P(c|A,b)P(d|c) \sum_e P(e|c) && \text{(3.122b)} \end{aligned}$$

Inference calculation in Bayesian networks can be computed exactly or approximately. As mentioned earlier, as the number of variables (and variable states) increases, the full joint distribution represented by the network becomes intractably large (and so the CPTs associated with each node). Variable elimination [42][55] is a common exact inferencing algorithm for Bayesian networks, and it is given special attention here due to its relevance in the Archimedes Palimpsest network discussed in the following chapter.

In variable elimination, queries are evaluated by first eliminating irrelevant nodes from the computation. In the scenario of Eq. 3.122b, the last term sums to unity, indicating that removing the node for variable *E* does not affect the result. In other words, variables that are not ancestors of query or evidence variables are eliminated from the calculation to reduce complexity. Factors, that is, vectors of probabilities associated with the remaining nodes, are generated, and then the *pointwise product* is computed while back-tracking through the network. Quite simply, the pointwise product computes the union of the set of variables represented in the factors. Hidden variables encountered during this process are immediately summed-out (marginalized). The resulting factor should contain the likelihood for each state in the query variable. A possible algorithm for variable elimination is:

Algorithm 3.4.1: VARIABLE ELIMINATION(X, e, BN)

```
comment: X is the query variable
–remove variables irrelevant to query
–generate factors for remaining variables
for each factor in reverse order of causal dependence
    pointwise product current factor an any previous factor
    if new factor contains hidden variable
      then marginalize over hidden variable
normalize
return (X)
comment: return value is a new distribution over X
```

Polytrees exhibit the useful property that the time and space complexity of exact inference (using variable elimination) is linear in the size of the network. Approximate inference techniques, such as random sampling algorithms [24][42], are typically required in multiply-connected networks, as exact inference results in exponential time and space complexity.

–Basic research is what I am doing when I don't know what I am doing.

Werner Von Braun

–No human investigation can be called real science if it cannot be demonstrated mathematically.

Leonardo da Vinci

4 Approach

We now discuss the role(s) of the pattern recognition topics of Chapter 3 in the completion of this body of work. It may be useful to revisit the objectives outlined in Chapter 2 before examining the approach employed here. While the overarching goal of this research has been to develop and maintain a character recognition tool for the Archimedes Palimpsest Project, three other specific objective categories also exist. This chapter describes the approach for each, loosely following the chronological progression of the work encompassed by this dissertation.

4.1 Spatial Correlation Filter Analysis

Each of the correlation filters detailed in Section 3.2, whether of classical or composite design, exhibits specific attributes that determine its performance when applied to an arbitrary application. For example, the simple matched spatial filter (MSF) was derived to be optimal (in a least-squares sense) [4] for detecting a known signal in the presence of additive white Gaussian noise (WGN). When using correlation pattern recognition to determine the true class of a character contained in a noisy image, one would expect the MSF to provide a more useable result than that produced by the ideal matched filter. However, if both correlation filters were applied to an identical input image that contains a negligible amount of low-frequency noise, the result is just the opposite: the ideal MSF produces an approximate Dirac delta function when matched to the true-class reference and an amplified noise floor when matched to any false-class references. Therefore, performance metrics corresponding to these peaks will show larger separation between true and false-class characters, providing better discrimination than metrics corresponding to the characteristic broad peaks produced by the MSF. We are presented with an analogous scenario when using composite correlation filters, which provide additional tolerance to within-class distortion. For images of the Archimedes Palimpsest, it may be advantageous to apply a specific correlation filter for some regions, but other filters to regions with different spatial properties (clean, noisy, cluttered, etc.). This notion is revisited in the following section on the design of a recognition system, but for now, this implies that it is necessary to analyze multiple correlation filters.

For correlation to be a useful means of character recognition on images of the palimpsest, it requires tolerance to both noise and clutter and the ability to accommodate small within-class distortions while maintaining a high level of class discrimination. From Section 3.2, it should be clear that designing a filter to simultaneously satisfy each of these criteria would be exceptionally difficult and impractical. The remainder of this section will examine the performance of individual correlation filters which, when used in conjunction, provide each of these attributes.

During the first stages of this research, before recognizing the Archimedes Palimpsest as a possible application, the performance of several classical correlation filters (including the then-newly derived complement filter [26]) was evaluated using reference patterns dominated by low frequencies, additive WGN, and the performance metrics detailed in Section 3.2.3.1 [50][52]. As expected, several papers describe this "general" performance analysis for composite correlation filters [3][28][30]. Characterizing the performance of a filter for a wide array of conditions is certainly useful, because that design may be considered as an initial filtering candidate for target applications. However, the filter must be directly applied to data from the desired application to observe recognition rates, which should determine whether the design is integrated into the classification process.

A series of confusion matrices was generated and analyzed to evaluate the performance of the individual correlation filters (Section 3.2) on images of the Archimedes Palimpsest. Recall that receiver operating characteristic (ROC) curves are typically used to compare filters for binary detection problems (e.g., target "present" vs. target "absent"). The Greek alphabet alone has 24 characters, both as upper and lowercase, and ligatures often exist, which increases the number of distinct characters.

The first step in completing this process was to extract a training library of "clean" images of the Greek characters from unobstructed regions of the palimpsest, such as the gutter or side margins of the book. As mentioned in the previous chapter, at least one additional set of nontraining characters, i.e., an independent data set, should be used to evaluate the classification performance. Thus, a test set of clean and a set of obscured characters were assembled in addition to the original training library. Each set, *training*, *clean*, and *obscured*, included ten 64×64 pixel images per character class.

When generating a composite correlation filter, it is necessary to center the pattern of interest in each image from a set of training exemplars to obtain the desired peak. Failing to position the pattern consistently across the full set will either blur the peak or create multiple peaks from mismatched linear phase terms. Each of the 64×64 pixel images from the *training* sets was initially a 148×148 region that was extracted with the appropriate character approximately centered. The first visually centered training image from each character class was used as a reference for all other images in the set. From this reference, the locally nonlinear matched filter (LNMF) was generated and applied to the other images within the class set to produce correlation planes from which peak locations were used to determine the spatial translation correction. After applying this shift, the images where then cropped to 128×128 and subsequently downsampled to 64×64 (Figure 4.1). Note that the translation correction is not needed for the independent sets due to the property of shift invariance inherent in correlation filter design.

The confusion matrices calculated using this data set are provided in Chapter 5. With the development and implementation of a scheme for quantifying recognition rates on the Archimedes Palimpsest, we can now examine the design of the preliminary character recognition system.

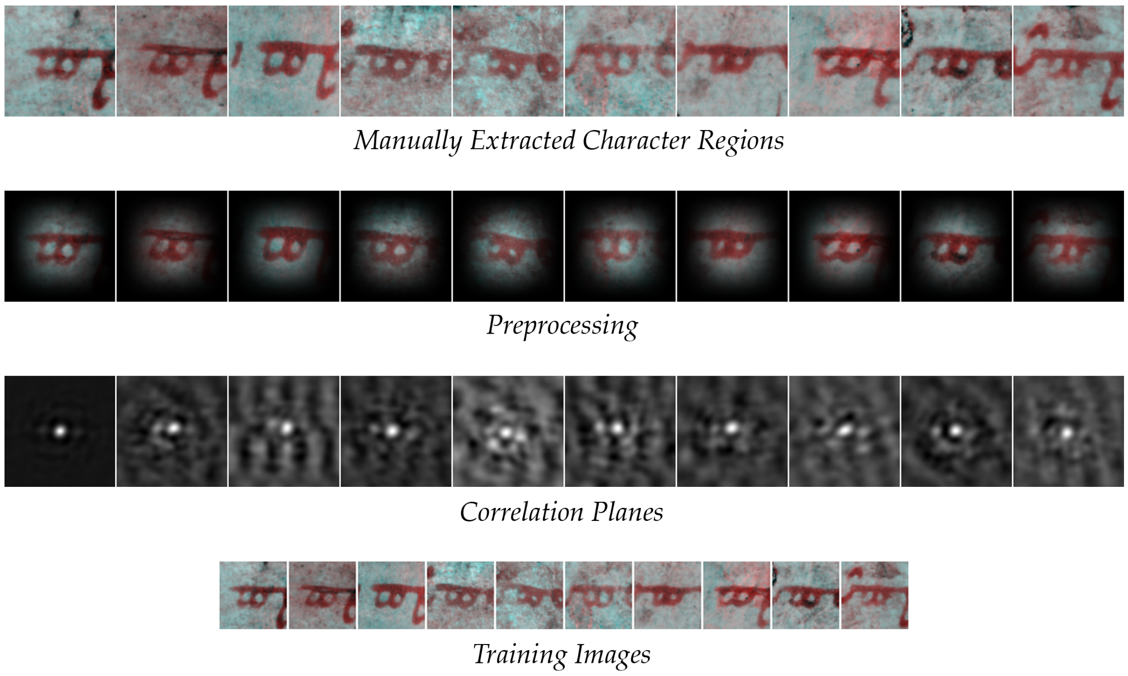


Figure 4.1: Example training image generation for the character 'π' – 148 $\times$ 148 pixel images of the approximately-centered 'π' character are extracted from clean regions of the palimpsest. Each image in the set is preprocessed by applying an apodizing window to minimize edge-effects and emphasize the spatial structure of 'π'. A correlation filter is then constructed using the first image in the set as the reference pattern, and correlated to every other image to produce a set of correlation planes, which are shown here as images. Any translation relative to the reference is corrected, and the images are then cropped to 128 $\times$ 128 and resampled to 64 $\times$ 64.

4.2 Archimedes Character Recognition Tool Design

At the symposium held in April of 2004 at the Walters Art Museum in Baltimore, MD, the scope of this research became quite clear during a discussion with Dr. Reviel Netz, of Stanford University. Unfortunately, this sudden sense of clarity was accompanied by a realization that many obstacles would ensue in the forthcoming imaging task. At this stage of the Archimedes Palimpsest Project, Netz stated that a tool designed to assist the user in the identification of characters in obscured regions would be of great assistance to those transcribing the digital images. After a preliminary evaluation of correlation filter performance for arbitrary applications [50][52], it seemed intuitive to apply this work to the design of a character recognition system. Feature extraction by correlation offered several benefits, including fast computation time, tolerance to noise and within-class distortions, and most importantly, shift invariance. The ability of the feature extractor to essentially bypass an application-specific segmentation process guaranteed that the tool could be easily modified to accommodate additional spatial pattern recognition tasks from other application areas (Section 5.2.3). Work on a correlation-based character recognition system for the use of those transcribing the palimpsest commenced immediately following the symposium.

4.2.1 Correlation Pattern Recognition System

Common to the development of almost all software design, it was imperative to consider how the end-user would use the recognition tool to achieve the his or her goals. We will assume that the end-user is a scholar of *Classics*, and he or she intends to use the tool in the transcription process. A graphical-user-interface (GUI) was developed in the IDL^{TM} programming language to allow a fully functional system to be distributed to the scholars for local usage in the IDL^{TM} virtual machine. The goal of the preliminary design was to provide the user with a list of probabilities for each character class from whichever region-of-interest was selected, based solely on spatial correlation results. As mentioned in the previous section, the spatial characteristics of regions selected from the palimpsest images will vary based on the amount of noise, overtext, mold, fire damage, etc. As such, the performance of a correlation filter will also vary with the ROI selected by the user. *Thus, rather than use a single correlation filter and an associated performance metric as the full classification scheme, it was determined that* *a series of decidedly useful correlation filters should be employed to construct a feature vector for a ROI.* This collection of correlation filters can be predetermined for a particular application by using the analysis discussed in Section 4.1.

The system was designed to require selection only of a file containing default settings for the desired application (e.g., the training, correlation filter, and performance metric sets), load an input image scene, select a region-of-interest, and press the "run" button. However, several computations are performed in the background before the results are presented to the user. A high-level diagram of the preliminary recognition system is shown in Figure 4.2. First, the training and ROI images are preprocessed by applying an apodizing window to minimize leakage during computation Fourier transforms, and to reduce the amplitudes of spatial features within the image that do not correspond to the segmented character. To ensure equal contribution of each training image in construction of the composite correlation filters, the integrated energy of the training images is normalized such that the sum of the squared pixel values (in each image) is unity [46]. Selected correlation filters are then applied to the ROI using each character class to generate the filter transfer functions. For a selected composite correlation filter, this method produces one correlation surface per character class; a classical spatial matched filter produces N_k correlation surfaces per class, where N_k is the number of training exemplars belonging to class k. After each surface is generated, the selected performance metrics are extracted to characterize the quality of the spatial matching and then stored in the feature vector of the appropriate class.

The result of this process is a feature vector for each class whose length L is the product of the number of correlation outputs produced for the class and the number of performance metrics used [53]. A feature vector for the ROI is created in the same fashion by matching the ROI to itself (i.e., the autocorrelation) using each filtering scheme and the successive performance metrics. The data used for classification are the ROI feature vector and a matrix of class feature vectors for each character with the number of rows equal to the length of each feature vector L and the number of columns equal to the number of character classes c. An example of the output using this data-formatting convention if the PSR and PCE metrics are used is shown in Figure 4.3.

Principal components analysis is typically employed at this stage in the system to reduce the volume of data and, more importantly, to remove any unnecessary fea-

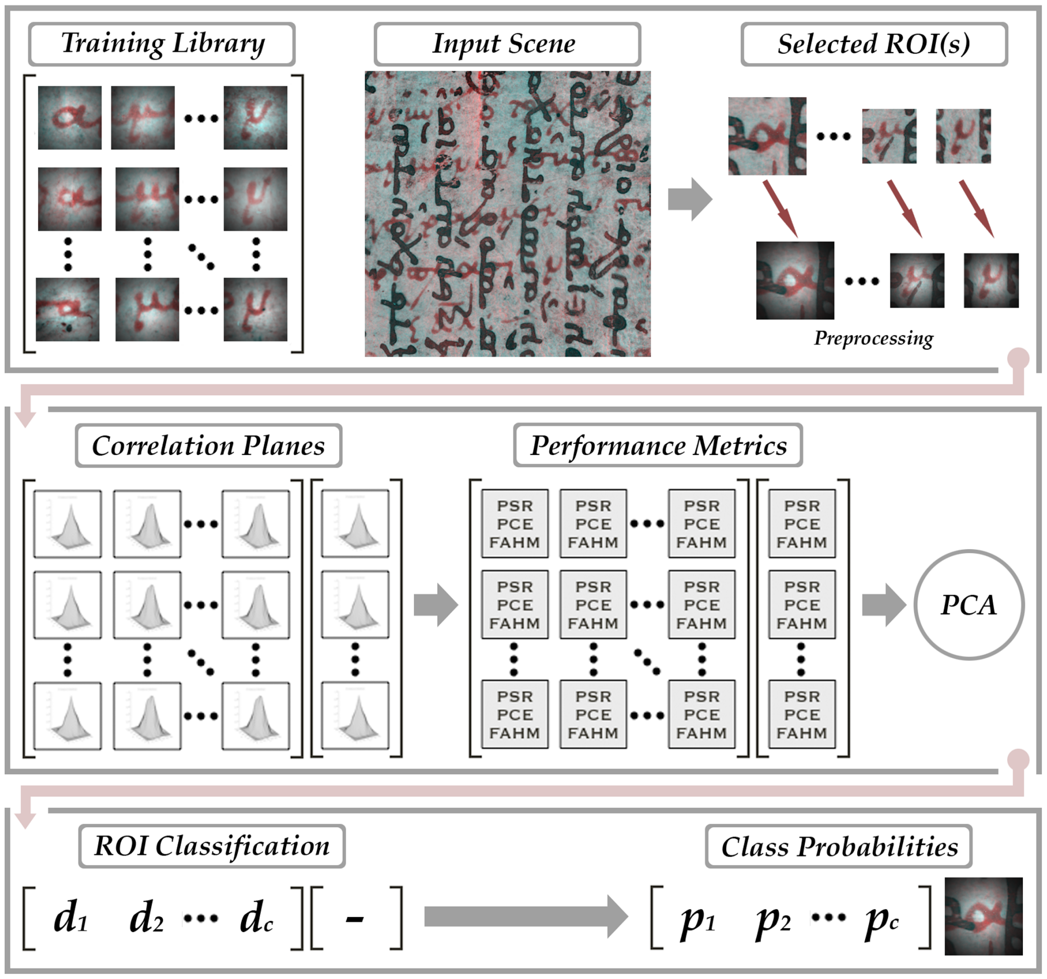


Figure 4.2: High-level diagram of the CPR system – (Top) the inputs to the recognition system include the training library of characters from the underwriting and ROI(s) selected from the input scene, both of which are preprocessed by applying an apodizing window. (Middle) Class and ROI feature vectors are generated using performance metrics from the respective correlation and autocorrelation planes. PCA can be performed to reduce the dimensionality of feature space and partially remove the effects of poor correlation. (Bottom) The relative differences between Euclidean distances to the ROI feature vector $d_1, d_2, \ldots, d_c$ are used to assign probabilities to each class $p_1, p_2, \ldots, p_c$.

$$\begin{array}{cc}
\begin{array}{cccc} \textbf{class } \mathit{1} & \textbf{class 2} & & \textbf{class } c \end{array} & \textbf{ROI} \\
\begin{bmatrix}
\begin{bmatrix} \text{PSR}_{1,1} \\ \text{PCE}_{1,1} \end{bmatrix} & \begin{bmatrix} \text{PSR}_{1,2} \\ \text{PCE}_{1,2} \end{bmatrix} & \cdots & \begin{bmatrix} \text{PSR}_{1,c} \\ \text{PCE}_{1,c} \end{bmatrix} \\
\begin{bmatrix} \text{PSR}_{2,1} \\ \text{PCE}_{2,1} \end{bmatrix} & \begin{bmatrix} \text{PSR}_{2,2} \\ \text{PCE}_{2,2} \end{bmatrix} & \cdots & \begin{bmatrix} \text{PSR}_{2,c} \\ \text{PCE}_{2,c} \end{bmatrix} \\
\vdots & \vdots & \ddots & \vdots \\
\begin{bmatrix} \text{PSR}_{L,1} \\ \text{PCE}_{L,1} \end{bmatrix} & \begin{bmatrix} \text{PSR}_{L,2} \\ \text{PCE}_{L,2} \end{bmatrix} & \cdots & \begin{bmatrix} \text{PSR}_{L,c} \\ \text{PCE}_{L,c} \end{bmatrix}
\end{bmatrix}
&
\begin{bmatrix}
\begin{bmatrix} \text{PSR}_{1} \\ \text{PCE}_{1} \end{bmatrix} \\
\begin{bmatrix} \text{PSR}_{2} \\ \text{PCE}_{2} \end{bmatrix} \\
\vdots \\
\begin{bmatrix} \text{PSR}_{L} \\ \text{PCE}_{L} \end{bmatrix}
\end{bmatrix}
\end{array}$$

Figure 4.3: Example data used for classification (prior to PCA) – an $L \times c$ matrix of feature vectors belonging to each of the c classes and the L-element ROI (autocorrelation) feature vector.

tures. Performance metrics corresponding to correlation schemes that behave poorly for given spatial characteristics of a particular ROI will exhibit small magnitudes that are relatively constant over each of the character classes. Conversely, metrics from correlation filters that have performed well should show a much higher variance due to classes that correlate well with the ROI.

Note that performance metrics often have different scales, and thus, the correlation matrix (as opposed to the covariance matrix) of the data set *should* be used to calculate the principal components. However, the PCA implementation varies between Case 1 and Case 2 (Section 3.3.1.3) based on the length of the feature vectors. For example, if the MACH filter and PSR performance metric are chosen, then the feature vector for each class is of length $L = 1$. If additional filters or metrics are added to the methods of feature extraction, L can certainly exceed the number of training classes c. Rather than manage the small sample size problem for PCA using the correlation matrix, which would require a normalization matrix, each feature is instead normalized by the maximum of the features from its corresponding feature-extraction method. This allows the relative variation between feature-extraction methods to appear on the same scale for the PCA computation.

The reduced matrix of feature vectors in the PC space contains the useful transformed features for classification, and features resembling noise are essentially removed. Obviously, if PCA is implemented, the ROI autocorrelation feature vector is subsequently projected into the PC space. As the final step in the pattern recognition process, the Euclidean distance between the feature vector of each character class and the ROI autocorrelation feature vector is calculated (Figure 4.4). The relative magnitudes of these distances are used to assign a probability to each class, forming a character probability distribution. Probabilities are calculated as the ratios of the distance for the given class to the distances for all classes. Results from this preliminary system are presented in Chapter 5.

4.2.2 Probabilistic Network Integration

To further improve the performance, it was necessary to address some of the obvious limitations of the character recognition tool. The original system provided classification results based solely on the spatial properties of the training exemplars and ROI image(s). It is clear that the texts of Archimedes are obscured, typically by mold and fire damage, as well as the text of the Euchologion prayerbook. Thus, images of the character from Archimedes' work occasionally provide little or no new useful information for recognition by correlation filtering. Moreover, extreme cases exist where entire characters or words are missing from the leaves of the palimpsest. At this stage in development, the CPR system was limited to discriminating character classes at single locations. Simultaneously discrimination classes at multiple ROIs allows analysis of partial information at each location during the classification task. While a certain level of success had been obtained from early efforts in recognition via correlation methods [53][6], it was determined that a method for incorporating contextual information into the system would undeniably increase the overall success rate.

To accomplish this task, it has been useful to examine the manner in which the end-user will use the character recognition system to transcribe as much text from the manuscript as possible. Ideally, the scholar would select the ROI(s), generate initial results using data both from the images and context, and apply that information to the tool to update the classification based on knowledge of the language. Thus, it is advantageous to model the inherent uncertainty at each stage in the decision-making

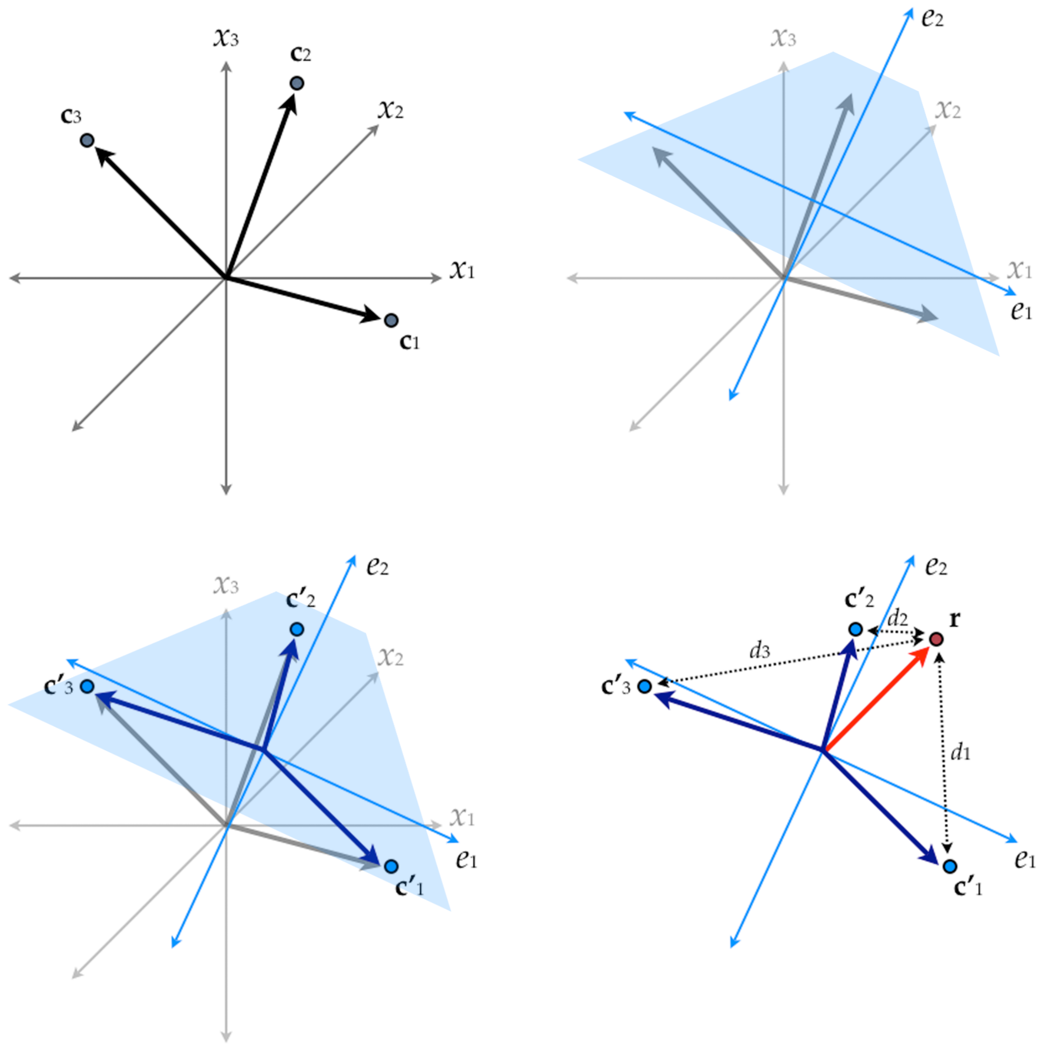


Figure 4.4: Illustration of Euclidean distance calculations in PC space – (Top-Left) feature vectors $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3$ from 3 different classes in the original feature basis x_1, x_2, x_3. (Top-Right) Reduction in dimensionality from three, the hyperspace defined by x_1, x_2, x_3, to two, the hyperplane defined by e_1, e_2, via PCA. (Bottom-Left) Feature vectors $\mathbf{c}'_1, \mathbf{c}'_2, \mathbf{c}'_3$ from the 3 different classes after projection onto the orthonormal-vector subspace e_1, e_2. (Bottom-Right) Euclidean distances d_1, d_2, d_3 calculated in the PC space from each of the transformed class feature vectors to the transformed ROI (autocorrelation) feature vector $\mathbf{r}$.

process.

The Archimedes Palimpsest is a mathematical text and therefore limited in vocabulary; additional texts in other contexts are discussed in Chapter 5. As mentioned in Section 1, earlier efforts in multispectral imaging have contributed to the transcription of nearly eighty percent of the Archimedes text. Using the partial transcription results produced by Reviel Netz of Stanford University [39], a *word dictionary* look-up table (LUT) was constructed. This information is crucial to the development of a network for character recognition.

Perhaps the best method to examine the network model for the Archimedes Palimpsest Project is to illustrate a simple character recognition experiment. Assume that the task is to determine the most likely character orientation for a word consisting of four characters: $l = 4$. Let us also assume that the alphabet of language S consists of only three character classes $c = 3$ such that $S = [s_1, s_2, s_3] = ['A', 'B', 'C']$ and that the observer is certain of one character in the 4-letter word.

This example is clearly intended to parallel the task of designing a probabilistic network design for the Archimedes Palimpsest Project, though on a much smaller scale. The scholars of ancient Greek mathematics often encounter similar situations. However, there are twenty-four character classes to discriminate between (forty-eight when including upper-case versions of the Greek alphabet). Despite this obstacle in computational complexity, the mathematical nature of the text limits the number of possible words to approximately fifteen hundred. This dramatic reduction in the number of valid character orientations qualifies the use of the generated word dictionary as a practical LUT for the problem.

Organizing a Bayesian model for a task typically begins by defining the variables, their states, and the causal dependencies. We start by identifying the hypothesis events, variables that are not observable. Figure 4.5 shows the nodes of variables that correspond to each character in the word or region-of-interest (ROI). More specifically, these variables are ROIs selected by the observer for each character location in a word image of text from the manuscript (Figure 4.6). The likelihood (or certainty) of each state in these nodes is estimated initially by using the only information provided: the word LUT. Note that the ROI_2 node is offset from the remaining ROI nodes to show that it refers to explicit (hard) evidence, that is, the character in this location is known by the observer.

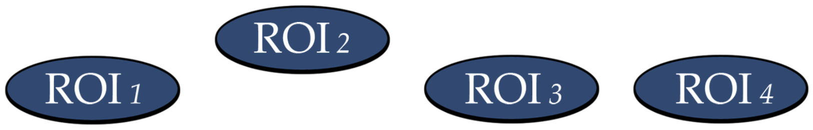


Figure 4.5: ROI nodes for a word consisting of four characters.

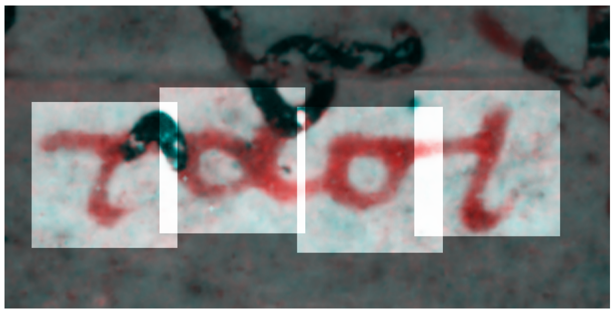


Figure 4.6: Example from an Archimedes Palimpsest pseudocolor image of a four-character word that shows segmentation into ROIs by an observer (a scholar of ancient Greek mathematics). *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest)*

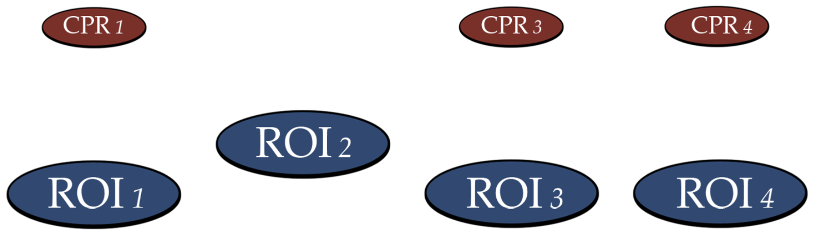


Figure 4.7: CPR results in the system over their corresponding ROI nodes.

The CPR system, which was developed to exploit the spatial properties of images, assigned probabilities to character classes at individual ROIs. Even the simple example here could benefit from this correlation information by incorporating these results into the network. Figure 4.7 shows an updated system for this task with the outcome of correlation pattern recognition. Note that the symbols used to denote CPR are smaller than those of the ROIs to emphasize that the CPR results are *not* nodes in the network. Also note that ROIs containing hard evidence from the user do not require the associated CPR information. Intuitively, if the ROI state is known, then $P(ROI_i) = 1$.

The task now is to determine the direction of the directed edges, or causal relationships, between variables. If we assume that information is transmitted from left to right in the ROIs, then the network graph should resemble that in Figure 4.8 (this assumption is revisited later in this section). This implies that each character in the word is directly influenced by the character which precedes it. The network structure here has the benefit of being a polytree, which permits practical use of the variable elimination algorithm (discussed in Section 3.4.3). As shown, the ROI_3 node is causally dependent on the ROI_2 node, and it is also influenced by the corresponding CPR_3 results. CPR_3 provides evidence using spatial characteristics of the images, and the ROI_2 node provides evidence using contextual information. Recall that for this example, the state of ROI_2 is known.

Initial probabilities can be specified after defining the network structure. We assume that each node has the same states: the three character classes, '*A*', '*B*', and '*C*'. ROI_1 is the only node in the network without a parent node. Hence, ROI_1 has an as-

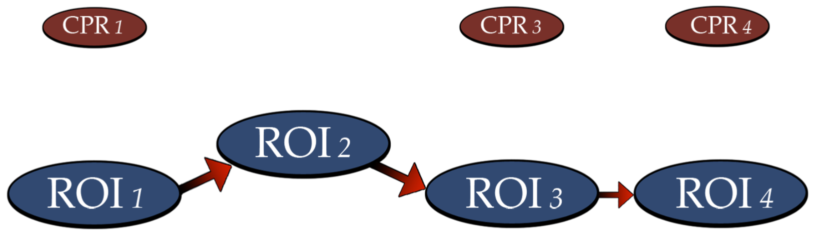


Figure 4.8: Network diagram showing causal relationships.

sociated probability distribution table rather than a full conditional probability table. The CPTs must be computed for each child node given the causal information from its parent(s). Figure 4.9 shows the ROI_3 node with its associated CPT, where the table represents $P(ROI_3|ROI_2)$. More specifically, the elements of the CPT are given by $P(ROI_{3_i}|ROI_{2_j})$ where $i = j = 1, \dots, 3$ and ROI_{3_1} would represent $ROI_3 = `A'$.

The dependency relationships between the ROI variables and the CPR results is unknown. We employ the *naive Bayes' rule* to assume statistical independence between the conditional probabilities $P(ROI_3|ROI_2)$ and $P(ROI_3|CPR_3)$. This can be expressed mathematically as:

$$P(ROI_3|ROI_2, CPR_3) = P(ROI_3|ROI_2) \times P(ROI_3|CPR_3) \tag{4.1}$$

If we have access to the LUT mentioned earlier, then we can determine the $P(ROI_3|ROI_2)$ term. $P(ROI_3|CPR_3)$ reduces to $P(CPR_3)$, because without any knowledge of the context, the probabilities associated with each state in CPR_3 are used as probabilities for the corresponding states in ROI_3. The product of the $P(ROI_i|ROI_j)$ and $P(CPR_i)$ terms allows the CPR information to contribute to initialization and query results independently of the probabilities produced using contextual information.

Returning to the example query, notice that of the $3^2 = 9$ elements in CPT_{ROI_3}, seven have a zero likelihood of occurrence. ROI_2 was stated to be known, and we are assuming $ROI_2 = A$ for this scenario. The bottom 2 rows (6 elements) represent impossible events. The word dictionary LUT can now be used to determine if any states in ROI_3 are invalid. As apparent from the diagram in Figure 4.9, $ROI_3 \neq A$

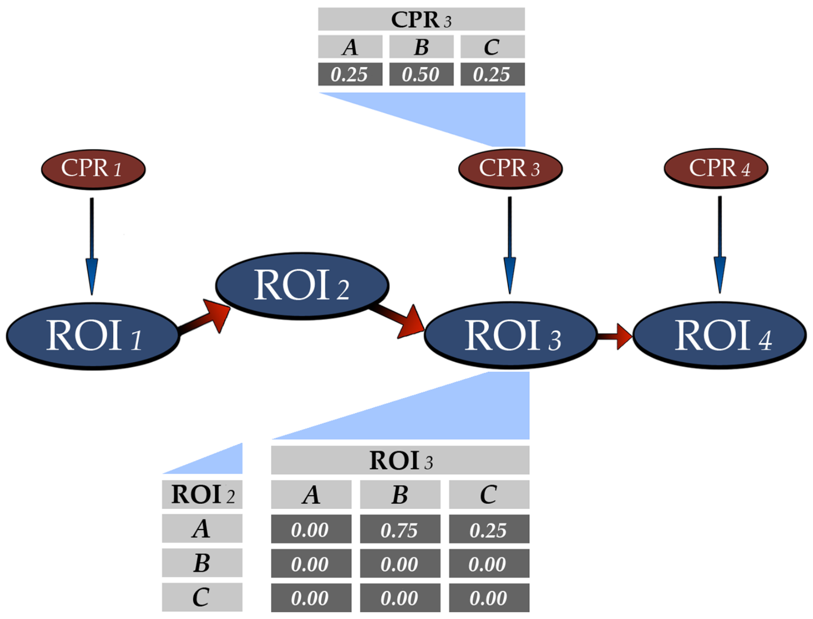


Figure 4.9: Network diagram showing an example CPT associated with ROI 3 when ROI 2 is known to be the character '*A*'.

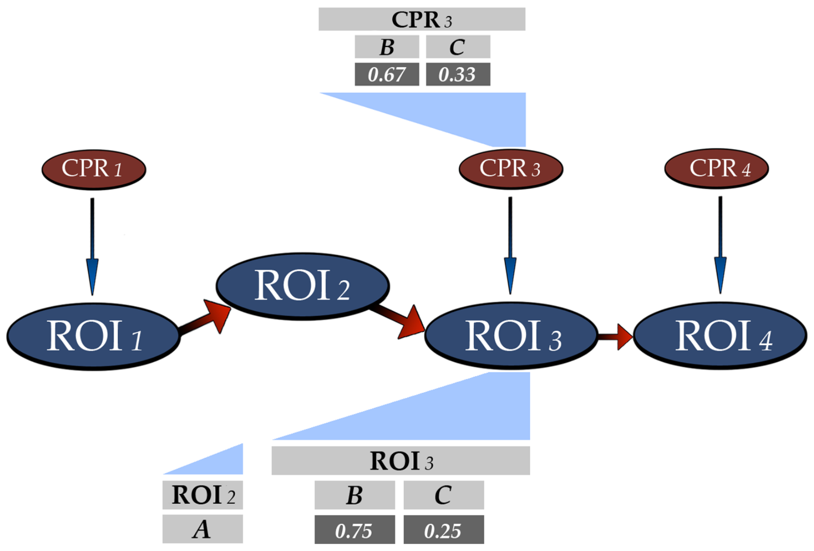


Figure 4.10: Network diagram illustrating the update of adjacent CPR results.

given $ROI_2 = `A'$. Consequently, the $`A'$ state of CPR_3 can be removed from results using the correlation-based metrics of the CPR system. Only two elements in Figure 4.9 require calculation during inference.

Figure 4.10 illustrates the concept of relaying LUT contextual information from the ROI nodes with hard evidence to adjacent CPR results. Eliminating states in the CPR nodes reduces the class discrimination task of the correlation routines and increases the accuracy of the results. After removing states with zero likelihood, the probability distribution specified by a CPR node is renormalized over the remaining states.

This small-scale example is intended to outline the major steps in adding contextual information to classification results which were previously derived using only the spatial structure of the images. This network allows the user to observe the probabilities associated with each state of each variable, that is, each character class at each ROI. Using this information, the surrounding context, and the visual aid provided in the multispectral images, the user can apply hard evidence and perform queries to observe changes in character likelihoods. In addition, the user is able to observe the probability associated with each word in the LUT by treating the probabilities from each ROI as independent events.

To summarize, the general algorithm for the updated system consists of the following processes:

1. **Input:** Training library, input scene, ROI selection, and word LUT
2. **Correlation Pattern Recognition:** Generate CPR results at each ROI
3. **Network initialization:** Generate ROI node probabilities from word LUT and independent CPR results
4. **Input Evidence:** Apply contextual knowledge to select "known" characters
5. **Perform Query:** Generate new probabilities for each character class at the "unknown" ROIs
6. **Examine Probable Words:** Word LUT is viewed in order of probability
7. **Perform Steps 5-6 as Needed**
8. **User Decision:** Based on the CPR, transcription results, and contextual knowledge

The character recognition system was updated to allow use in three different modes: as a CPR system for individual ROIs (characters), as a probabilistic network system for multiple ROIs (word), or as a joint CPR-probabilistic network system (characters and word). Figure 4.11 shows a high-level diagram of the character recognition system after updating its design to integrate the probabilistic network.

In this preliminary network, the model for each word is assigned a polytree structure. It is worth mentioning that though each character is causally dependent on the character that precedes it, the output of a query may not be as intuitive if it were the result from a multiply-connected network in which each character is causally dependent on *all* characters that precede it. Using the polytree, the output could show a likelihood greater than zero for impossible character states for certain character locations given the applied evidence. For example, in a word consisting of eight characters, the state of the sixth character region ROI_6 may be known to be the character 'A'. Referring to the LUT, if the sixth character in an eight-character word is 'α', then it may be true that ROI_8 cannot be a 'γ'. However, a 'γ' in ROI_8 could have a positive likelihood depending on how the 'α' in ROI_6 influences ROI_7. Thus, a multiply-connected network (and a corresponding approximate inferencing scheme) may be desired to apply evidence to force invalid characters to have zero probability.

Results and examples using the joint CPR-probabilistic network system are presented in Chapter 5.

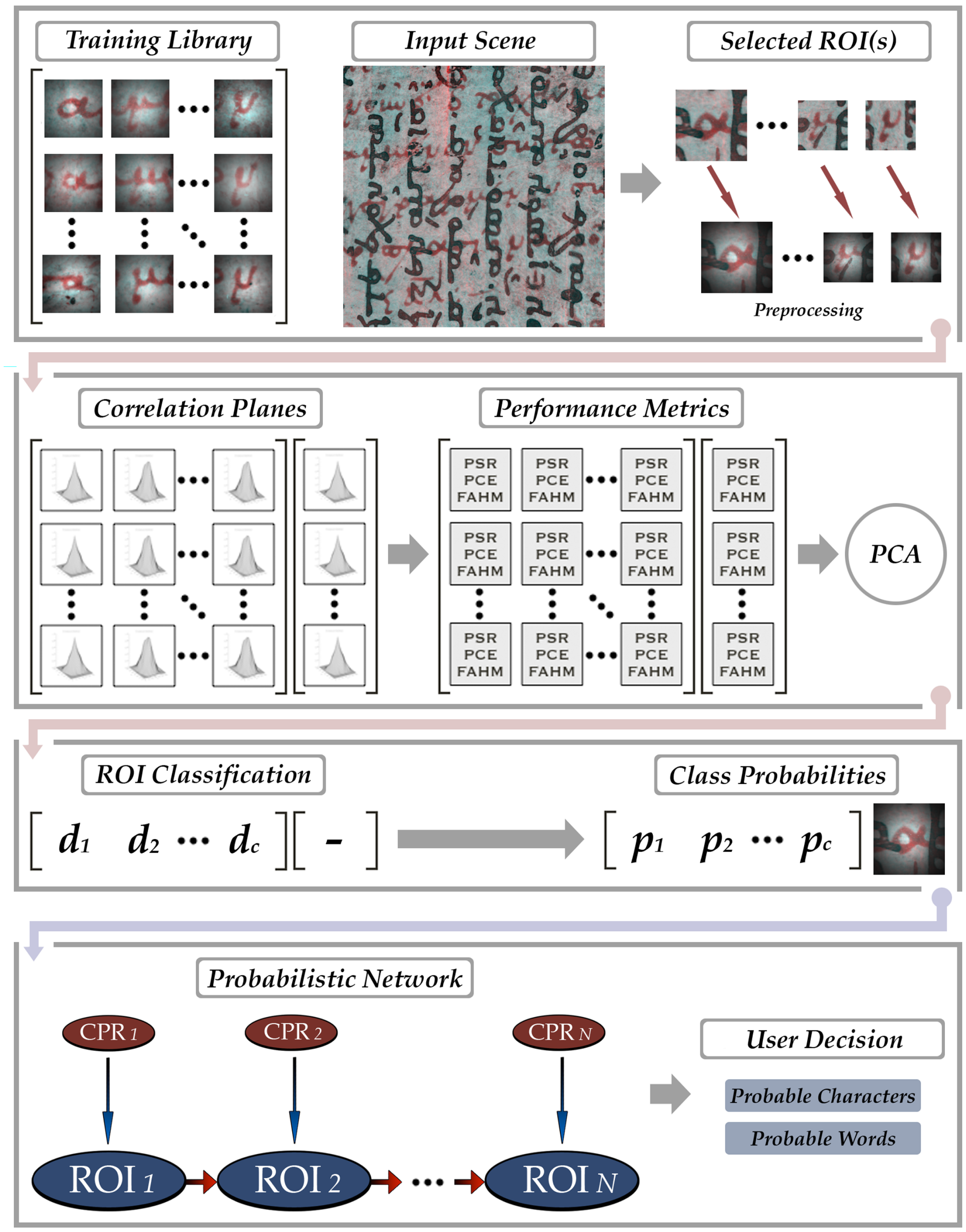


Figure 4.11: High-level diagram of the updated recognition system – CPR class probabilities (for each ROI) and the word LUT are used to initialize the network. The user can then base a decision on probable characters and words from query results.

4.3 Linear Subspace Correlation Filtering

After the completion of a series of correlation filter performance evaluations and the introduction of an updated character recognition system to the scholars transcribing the palimpsest, only one objective remained: *the design of an improved correlation filtering method*. As shown in Figure 4.12, the flexibility of the CPR system to accept additional filtering schemes provides a cyclic workflow in which new correlation methods can be evaluated by the system and integrated into its existing filter library. The CPR system was designed to accommodate new development and provide a framework for future work in the area of spatial pattern recognition for a broad range of applications.

This section attempts to address the topics of *how spatial correlation filters are designed* and *how they might be improved*. It is this discussion that, in large part, requires the lengthy presentation of material on correlation filtering in Chapter 3. All of the correlation filters discussed in Section 3.2 were derived either in the space or frequency domain. However, the first composite correlation filter, the SDF filter [19], noted that derivation of the correlation design in another space may provide better results. With this in mind, we revisit the concept of phase representation in correlation filter design to facilitate a discussion on the projection of phase information onto vectors in spaces other than the Fourier or spatial basis.

4.3.1 Improving Phase Representations in Correlation Filtering

The complement matched filter (Section 3.2.1.4) is a useful conceptual tool to demonstrate the importance of both the magnitude and phase components of a correlation filter. For any order N, the complement matched filter contain the same phase spectrum and thus performs the same phase canceling when applied to the input. Recall that this phase canceling contributes to peak formation at the location of the reference pattern. Ideally, the phase of the reference pattern contained within the input is completely removed from the output, leaving only the linear phase term(s) to specify the detection location(s). The order N controls the shape of the magnitude spectrum, allowing the complement matched filter to exhibit characteristics of the ideal matched filter, the POMF, or even the MSF. Again, the magnitude of a correlation filter controls the shape of the correlation peak and the tradeoff between peak sharpness and

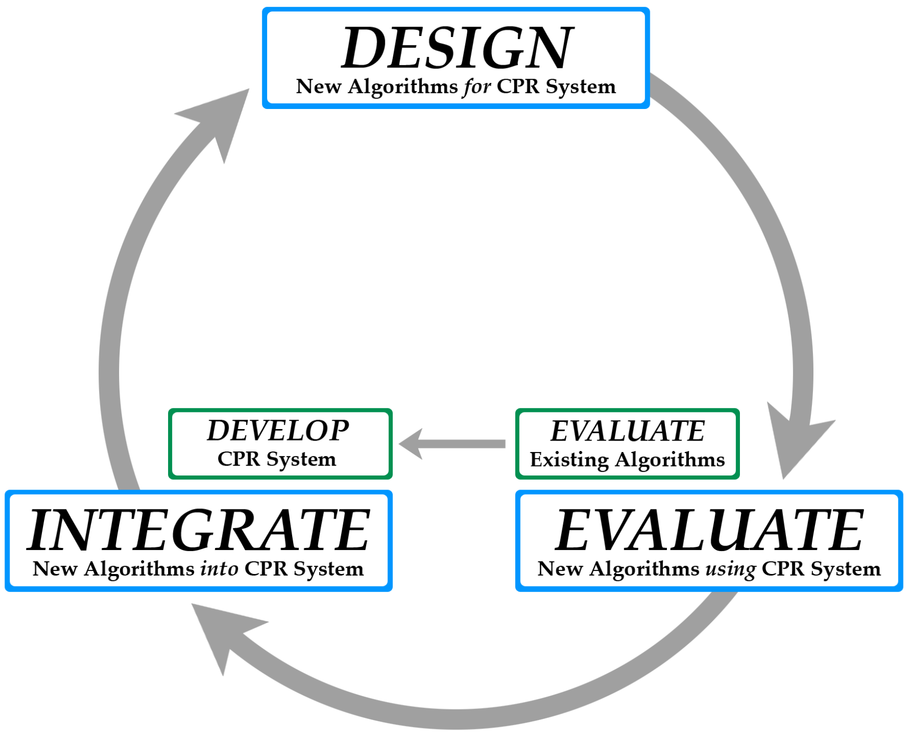

DESIGN
New Algorithms *for* CPR System
DEVELOP
CPR System
EVALUATE
Existing Algorithms
INTEGRATE
New Algorithms *into* CPR System
EVALUATE
New Algorithms *using* CPR System

Figure 4.12: The cyclic workflow provided by the CPR system – After the initial correlation filter performance evaluations and the development of the recognition system, the inherent flexibility of the design allows newly derived correlation algorithms to be evaluated using common performance metrics and integrated into the recognition process for the desired application.

noise tolerance. Obviously, correlation is not possible without phase information (for realistic imaging scenarios); we will focus on improving phase representation in filter design as a starting point.

It has already been noted that the complement matched filter, the ideal matched filter, the POMF, and the MSF all contain identical phase terms. A brief look through Section 3.2 should confirm that the same is true for all of the classical correlation filtering methods discussed. Even more interesting is the fact that the SDF filter, the MACE and UMACE filters, and the MACH filter all share the same "average" phase term for phase canceling. As shown in Section 3.2.2, composite correlation filters typically aim to minimize or maximize certain criterion functions under a given set of constraints. After solving the optimization problem, the resulting transfer function for each of these filters manipulates the magnitude component while retaining the average phase term. Thus, a question to consider is: *can the phase information of a composite correlation filter be represented better than the mean phase of the training set?*

To begin forming an answer to this question, refer back to the derivation of the principal component basis functions in Section 3.3.1.1. The problem was first formulated by finding a new vector $\mathbf{x}_0$ to best represent (in a least-squares sense) a given set of N vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$. It was shown that $\mathbf{x}_0$ must be the mean vector for the set of N training vectors. Equivalently, if the immediate task now is to find the "best" phase representation for a set of training phase vectors, the mean phase minimizes the sum-squared error criterion (Eq. 3.80). PCA allows reduction of dimensionality while maintaining a least-squares representation of the data. Perhaps the previous question should be modified: *Does the least-squares solution for the phase representation satisfy the inherent objectives of correlation filtering?*

In 2004, Savvides, et al., [43] published a new approach to correlation filtering that relied on PCA to produce a representation of the phase information from a training set. With the advent of eigenface techniques for face verification [49] and advances in modern computing technology, it was now possible to implement PCA on high-dimensional data sets consisting of more variables than observations (e.g., Case 2 PCA in Section 3.3.1.3). More specifically, PCA could now be computed on image data consisting of fewer training exemplars than total pixels. A concept that had seen little or no attention since its brief inception during SDF filter development in the early 1980's could now be revisited: the development of a correlation filter in PC space [19]. Savvides, et al., noted that the UMACE filter was just an average phase filter combined with prewhitening of the spectrum, i.e., multiplication by the inverse magnitude spectrum of the reference to obtain a flat correlation spectrum. To achieve illumination tolerance for face recognition applications, all magnitude information was removed from the input functions by assigning unit magnitude to the output spectrum; this is analogous to the SPOMF in Section 3.2.1.2. Instead, they focused on how to better represent phase in the filtering process.

The key to their algorithm is the linear subspace generated for the phase of the filter using the PC transform, with the term "corefaces" used to denote the eigenfunctions. It is important to note that Case 2 PCA is employed so that variables are pixels and the observations are training images; this calculation forces a reduction in dimensionality to the number of observations. Unlike the other filtering methods discussed thus far, the input phase Φ_I is correlated to the reconstructed input phase Φ_R, which is generated by projecting Φ_I onto the corefaces and then back to the Fourier basis. Phase canceling is performed in the frequency domain to produce a sharp, spatial domain correlation peak for true-class inputs with well-reconstructed phase. Thus, the Corefaces algorithm has these features: (a) *ignores magnitude information*, (b) *removes features with low variance from the phase spectrum of the training set by reducing dimensionality*, and (c) *relies on a PC subspace projection and subsequent reconstruction to the Fourier basis to provide class discrimination*. The results presented by Savvides, et al., [43] show that Corefaces outperformed other techniques such as the MACE filter, Fisherfaces [1], and variations of principal components analysis in the detection of faces with varying illumination.

The Corefaces algorithm for multi-class discrimination can be summarized into five steps:

1. Generate the PC subspace for each of the c classes
2. Project the input phase Φ_I onto each subspace
3. Reconstruct the input phase by projecting back to the frequency domain
4. Correlate the reconstructed phase Φ_R to the input phase Φ_I
5. Analyze the resulting peaks (using performance metrics) for classification

A block diagram of this approach is shown in Figure 4.13.

The purpose for including this extended discussion on Corefaces is in many ways analogous to the purpose for reviewing the ideal matched filter; both algorithms serve

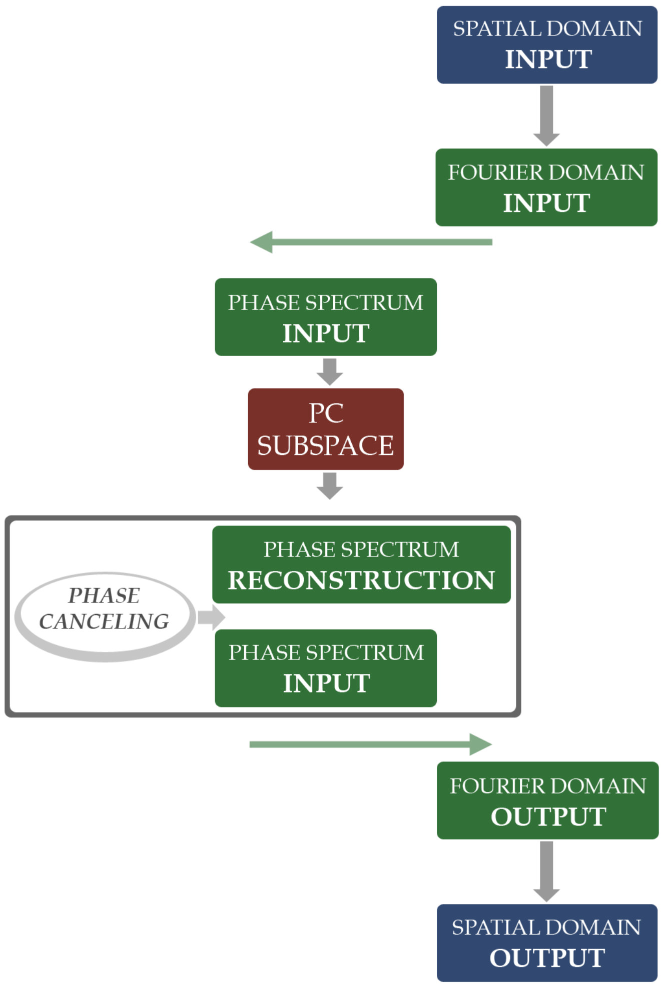


Figure 4.13: Block diagram of the Corefaces algorithm – the phase spectrum of the input Φ_I is projected onto a PC subspace (for the test class) and then projected back to the Fourier domain, producing a reconstructed phase spectrum Φ_R. After phase canceling, the inverse Fourier transform produces correlation plane in the spatial domain.

as a foundation upon which new correlation filtering methods can be derived. The recent discovery of the Corefaces approach is certainly more effective than the ideal matched filter for most realistic pattern recognition applications. However, the matching schemes are similar in that Corefaces sets the stage for filter development in a PC subspace, just as the ideal matched filter did in the Fourier basis. Corefaces is just one example of how linear subspace projection can be used to design the filter transfer function. This methodology will be referenced multiple times during the next section, which details the design of a new correlation filtering scheme.

There are a few caveats in Corefaces that require attention. By examining Figure 4.13, it is apparent that this approach would be exceedingly difficult to implement optically. Each filter is constructed by a projection onto (and out of) a subspace, which would require multiple "filtering" stages just to produce an approximation. Fortunately, most correlation filtering techniques are now performed using digital systems. In addition, the flat magnitude spectrum that Corefaces assigns the output correlation plane may be ineffective for applications containing significant high-frequency variability. The lowpass nature of the magnitude spectra inherent in many correlation filters serves to filter noise and small distortion.

For the following discussion, the restriction that will be of foremost concern is the preservation of shift invariance in the filtering process. Recall that spatial translation produces a linear-phase term, which is the only remaining phase information after applying the inherent phase canceling in correlation filtering (assuming ideal conditions). If dimensionality had not been reduced while calculating the PC space, we would expect perfect reconstruction of the phase information in the frequency domain. Obviously, this scenario would serve no purpose for filter design. Thus, there is a new question to be mindful of: *How does projection onto a linear subspace affect the linear phase component?*

Corefaces represents a special case of applying a linear transformation to the spectral phase. We desire reconstruction of the phase of the reference pattern Φ_F and elimination of the linear-phase component Φ_L. More specifically, the subspace should adequately represent the reference pattern Φ_F only, such that phase canceling the input phase Φ_I with the reconstructed-input phase Φ_R yields the linear-phase term Φ_L to specify the peak location:

$$e^{i\Phi_I} \cdot e^{-i\Phi_R} = e^{i(\Phi_F+\Phi_L)} \cdot e^{-i\Phi_F} = e^{i\Phi_L} \tag{4.2}$$

Again, the Corefaces approach assumes that (a) Φ_F *will reconstruct well from the PC subspace*, and (b) Φ_L *will be almost entirely removed by dimensionality reduction*. This process typically produces a useable result, because PCA is the optimal transform (in a least-squares sense) for reconstruction. Savvides, et al., [43] note that Corefaces has empirically demonstrated *shift-tolerance*; the linear-phase term Φ_L is predominant after phase canceling, producing a correlation peak at the correct spatial location of the reference pattern.

4.3.2 Improving Class Discrimination in Correlation Filtering

We can now discuss the possibility of using a linear subspace approach in spatial correlation to achieve the desired filtering characteristics. Many possibilities exist, but we will focus specifically on class discrimination for the correlation filter developed in this work. While Corefaces relied on the PC subspace to provide class discrimination, we seek a different linear subspace that is "optimized" to maximize class separability. To meet this goal, the transformation that will be exploited is specified by linear discriminant analysis, which was discussed in Section 3.3.2. By maximizing the Fisher criterion (Eq. 3.94), LDA produces eigenfunctions that point in the directions we desire, that is, directions that maximize between-class scatter under the constraint that the within-class scatter is minimized. Though it may already be clear, it should be noted that LDA produces a single linear subspace to represent variability in all classes.

Similar to the PCA implementation in the Corefaces algorithm, we are again concerned with the small sample size problem; the number of variables d exceeds the number of observations N. For this task, the variables, which define the dimensionality, are the pixels, and the observations are the training exemplars. The work of Turk and Pentland on eigenfaces in the early 1990's revealed a method to overcome the SSS problem for PCA [49]. Methods to solve this problem for LDA did not emerge until almost a decade later. The Direct-LDA algorithm proposed by Yu and Yang [54] recognizes LDA as a simultaneous diagonalization problem that diagonalizes the between-class and within-class scatter matrices in similar fashion to Case 2 PCA. This approach, which is outlined in Section 3.3.2.2, will be used throughout the remainder of this work.

We must again address the question: *How does projection onto the LDA subspace affect the linear phase component?* When the input phase Φ_I is projected from the Fourier basis onto the LDA vector subspace, the discriminating features of that phase will be retained, and other features will be attenuated or removed completely. If the input contains a translated version of the reference, it is uncertain how the linear phase term Φ_L will be reconstructed. Unlike Corefaces, which assumes that the reconstructed input phase Φ_R contains a close approximation of the reference pattern phase spectrum Φ_F, we now have a reconstruction that will often exhibit different structure than the input. Using PCA to generate the linear subspace guarantees that the reconstruction is the least-squares representation of a true-class input, thereby facilitating correlation between Φ_I and Φ_R. Without (a) *the PC subspace* or (b) *knowledge of how the linear phase term will be reconstructed*, we must adopt a new approach. Fortunately, reconstruction of the input information can be correlated to a reconstruction of the training information (rather than to it's own representation prior to projection). However, preservation of the linear phase term Φ_L presents a considerable obstacle. The assumption that reconstruction removes the linear phase term Φ_L is now irrelevant. We are forced to shift focus, projecting only the magnitude information onto an LDA subspace in correlation filter design.

The next question that requires attention is: *How does dimensionality reduction affect the correlation process?* Case 2 PCA requires dimensionality reduction to the number of observations N, while Case 2 LDA reduces dimensionality to $c - 1$ directions. For example, if there are $c = 3$ classes, each represented by one hundred 64×64 pixel training images, LDA reduces dimensionality from $d = 4096$ to $d = 2$. If each input is represented by a 2-element vector, it makes little sense to design or implement a correlation filter within the LDA subspace. Instead, the LDA representation can be projected back onto the original basis, essentially using the linear subspace projection as a means to filter unwanted information.

We proceed by examining the following three propositions, all of which take advantage of LDA in correlation filter design:

1. Project the input *image* onto an LDA *image* subspace, and correlate the input with the reconstruction

2. Project the input *magnitude* onto an LDA *magnitude* subspace, and use the difference between the input and reconstructed magnitude spectrums to design Fourier domain preprocessing filters for each class

3. Project the input *magnitude* onto an LDA *magnitude* subspace, and multiply the reciprocal of the reconstructed magnitude by the mean reconstructed magnitude of the training to produce the output magnitude spectrum

The first idea produces exceptional correlation peaks for input images that contain an untranslated reference pattern. However, the peak degrades rapidly once the reference pattern is translated due to poor reconstruction of the linear phase term. Without shift invariance, the first proposition useless for spatial correlation. The second idea may have some value; class-specific preprocessing filters could be applied before using each of the correlation filters such that frequencies that provide little discriminatory information are attenuated. However, preprocessing filters of this nature can also cause false-class inputs to resemble true-class inputs, which is precisely what this correlation scheme is designed to avoid.

The third and final proposition was pursued in this work. Figure 4.14 illustrates the series of projections in this process using vector diagrams. To begin, the LDA vector subspace is constructed using magnitude spectra of the training data. Input images (lexicographically-ordered into vectors) are transformed to the frequency domain, and the magnitude spectra are extracted and projected onto the derived subspace. For illustration purposes, the origins of the LDA subspace and the magnitude basis are shown aligned. In practice, the mean training magnitude vector must be subtracted from any input magnitude vectors prior to projection. The dimensionality is reduced to $(c-1)$ directions, and the input vectors are then projected back onto the d-dimensional magnitude space. Reconstructed magnitude spectra have components of the within-class variability removed and maintain high separation from other classes.

Figure 4.15 shows a high-level block diagram of the new filtering approach, which will be referred to as *linear subspace correlation filtering* (LSCF) from this point forward. As shown, the algorithm exploits the Corefaces approach to develop a representation for the phase of the filter and an LDA approach to assign the filter a magnitude spectrum. While the term "linear subspace correlation filtering" will be used to denote this combined PCA-LDA subspace filtering approach, the LDA subspace projection of the

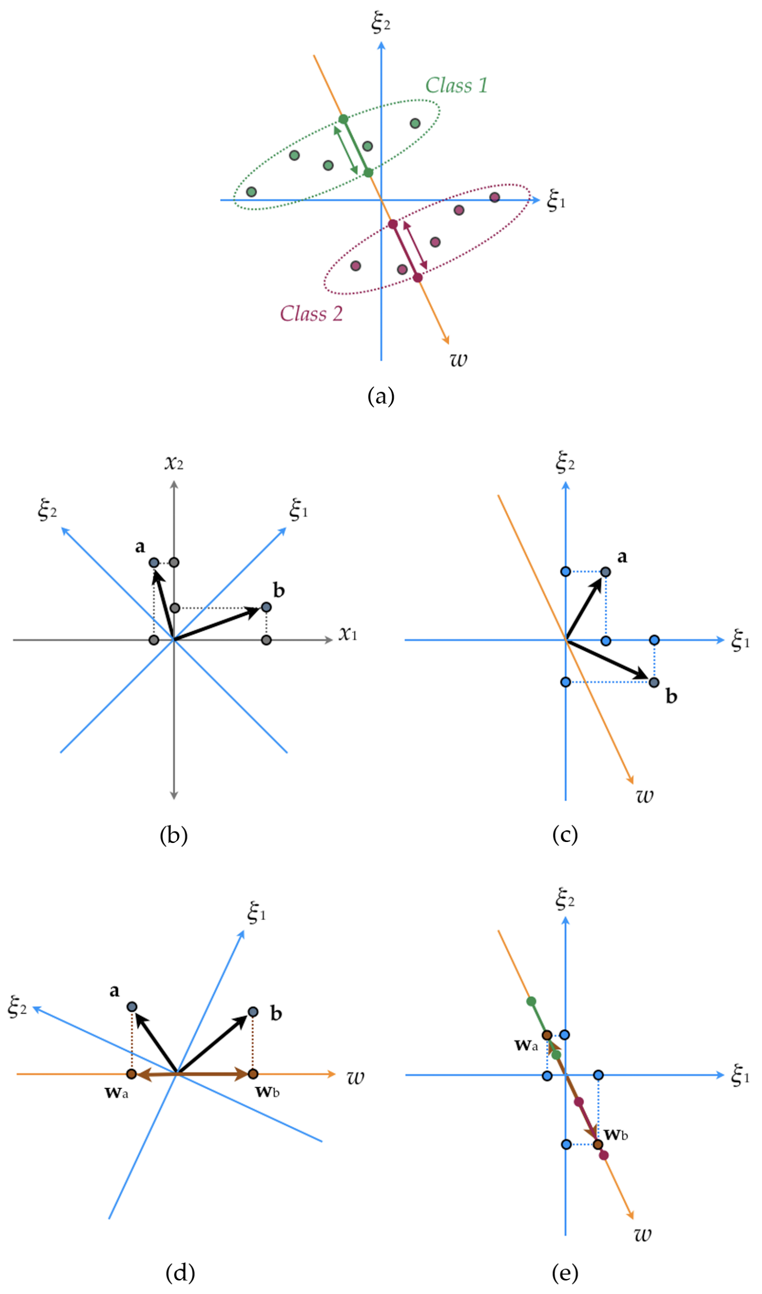


Figure 4.14: LDA subspace projection in $d = 2$ dimensions – (a) The LDA eigenfunction w derived from the training data. Arbitrary input vectors **a** and **b** are shown in the (b) spatial basis $[x_1, x_2]$ and (c) frequency basis $[\xi_1, \xi_2]$. (e) The input vectors are projected onto the LDA subspace $[w]$ yielding $\mathbf{w}_a$ and $\mathbf{w}_b$, and subsequently (f) projected back onto the frequency basis. As shown, **a** and **b** belong to *class 1* and *class 2*, respectively.

magnitude component could also be combined with additional phase manipulation schemes (e.g., an "average" symmetric phase-only matched filter).

The LSCF algorithm may be thought of as a correlation filter with magnitude and phase components that verify the accuracy of each other. After reconstruction from the LDA subspace, the product of the input and true-class mean magnitude spectra produces a result similar to a simple Euclidean distance calculation in the subspace. Magnitude information is nonunique, and therefore, this technique may produce a similar output for inputs that belong to different classes. In Corefaces, reconstructed phase spectra from different classes may exhibit strong similarity as a result of the information loss from reducing dimensionality. For many applications, the percentage of false alarms produced using the Corefaces algorithm should decrease by combining the resulting phase spectrum with the discriminatory information provided by the LDA-magnitude approach.

Before proceeding with an example of the LSCF algorithm, we can develop its mathematical formulation by first constructing the $d \times (c-1)$ LDA subspace matrix $\mathbf{W}$, which contains the LDA discriminant functions $\mathbf{w}_i$ as its columns (Case 2 LDA in Section 3.3.2.2). The magnitude spectra of the training exemplars from a specific class k can be lexicographically ordered into the column vectors of a $d \times N_k$ matrix $\mathbf{X}_k$. To remain consistent throughout this work, d is the number of pixels, and N_k is the number of training exemplars in the k^{th} class. Projecting these vectors onto the LDA subspace produces a $(c-1) \times N_k$ matrix $\mathbf{Y}_k$, which contains low-dimensional representations of the k^{th}-class training spectra:

$$\mathbf{Y}_k = \mathbf{W}^T \mathbf{X}_k \tag{4.3}$$

Unlike PCA, the discriminant functions defining the LDA vector subspace typically are not orthonormal. Instead of simply projecting the columns of $\mathbf{Y}_k$ onto the columns of $\mathbf{W}$ as we would for the inverse PC transform, we compute the pseudoinverse $\mathbf{W}^\dagger = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$. Projecting the columns of $\mathbf{Y}_k$ onto the columns of $\mathbf{W}^\dagger$ yields a $d \times N_k$ matrix containing the reconstructed-magnitude spectra $\mathbf{X}'_k$:

$$\mathbf{X}'_k = (\mathbf{W}\mathbf{W}^T)^{-1}\mathbf{W}\mathbf{Y}_k \tag{4.4}$$

Original magnitude spectra of $\mathbf{X}_k$ are now represented by $\mathbf{X}'_k$, which contains the

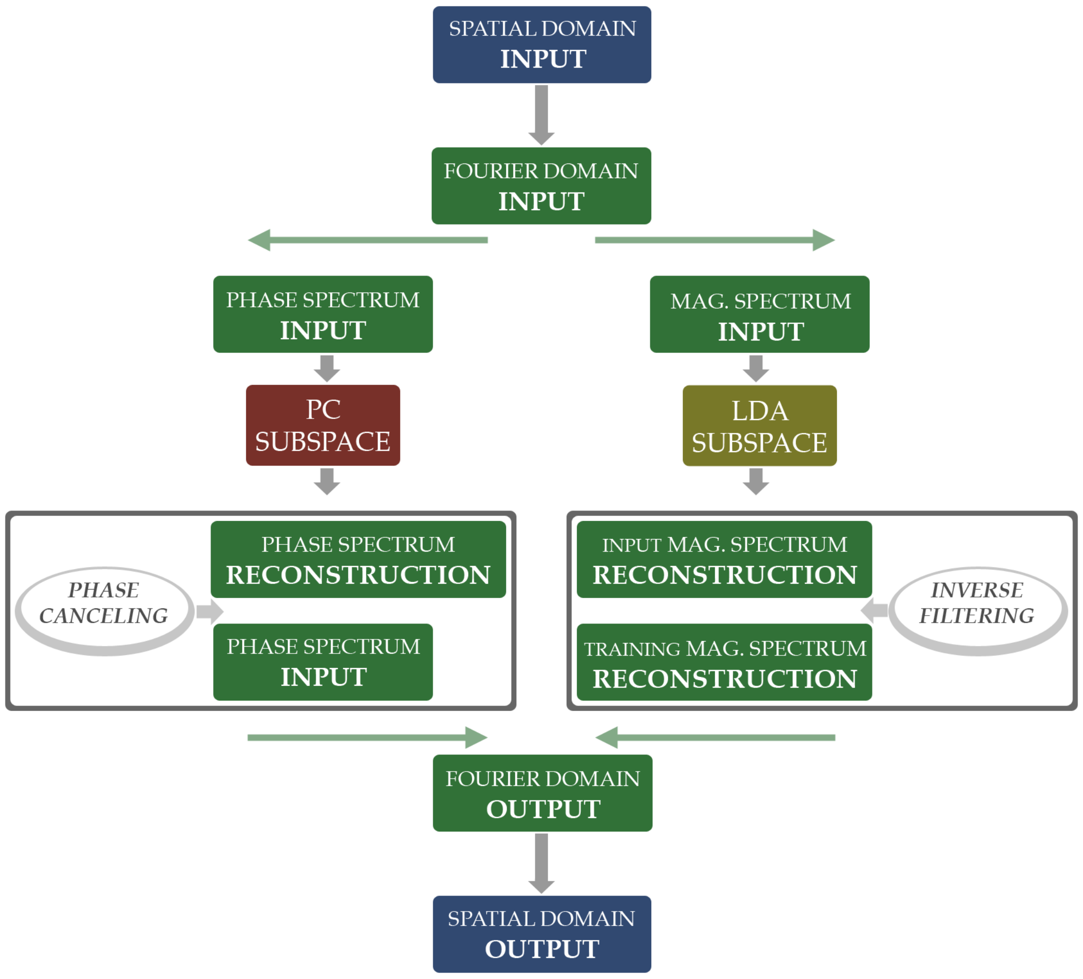


Figure 4.15: Block diagram of LSCF – the phase spectrum of the input is projected onto the PC subspace and follows the Corefaces algorithm. The input magnitude spectrum is projected onto the LDA subspace and back onto the frequency domain, and the reconstruction is subject to inverse filtering with the average training magnitude reconstruction of the test class. After both completion of both processes, the magnitude and phase spectra are combined to produce the output correlation spectrum.

partially-reconstructed magnitudes. The within-class variability is reduced, class separation is maximized, and dimensionality is remains equal to the number of pixels.

We are concerned with matching the reconstructions of the input magnitude spectra to the mean reconstruction of the magnitude spectra for class k, which is denoted by the d-element column vector $\mathbf{x}'_k$:

$$\begin{aligned} \mathbf{x}'_k &= \mathbf{X}'_k\mathbf{u} && (4.5a) \\ &= (\mathbf{W}\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{W}^T)\mathbf{X}_k\mathbf{u} && (4.5b) \end{aligned}$$

The vector $\mathbf{u}$ contains equal elements to compute the mean reconstruction $\mathbf{x}'_k$:

$$\mathbf{u} = \begin{bmatrix} \frac{1}{N_k} & \frac{1}{N_k} & \cdots & \frac{1}{N_k} \end{bmatrix} \quad (4.6)$$

A $d \times d$ diagonal matrix $\mathbf{D}_k$ can then be constructed with the elements of $\mathbf{x}'_k$ along the main diagonal. The filter vector $\mathbf{h}_k$ can be expressed as

$$\mathbf{h}_k = \mathbf{D}_k^{-1}\mathbf{p}_k \quad (4.7)$$

where $\mathbf{p}_k$ is a d-element column vector containing the desired phase spectrum. Using the Corefaces approach, the elements of $\mathbf{p}_k$ consist of the conjugate of the reconstructed reference phase spectrum using the k^{th}-class PC subspace. To obtain the output correlation spectrum, the transfer function $\mathbf{h}_k$ would be applied to an input containing the original phase spectrum and a reconstructed magnitude spectrum after LDA subspace projection. The correlation plane produced can then be evaluated to determine if the input belongs to class k.

In summary, the primary contribution to this filter design is the $(\mathbf{W}\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{W}^T)$ term in Eq. 4.5b. This term is responsible for the removal of uwanted information in the input magnitude spectrum upon projection onto the subspace. *Characteristics from each class of magnitude spectra that do not contribute to class separation are removed, yielding spectra that are easier to discriminate*. The process developed here may be interpreted as a method for obtaining a shift-invariant partial-LDA classifier. It would be necessary to retain the linear phase term to achieve true shift-invariant LDA.

An example of the LSCF approach applied to synthetically-generated characters is shown in Figures 4.16-4.19. An image of a character from the text of Archimedes was

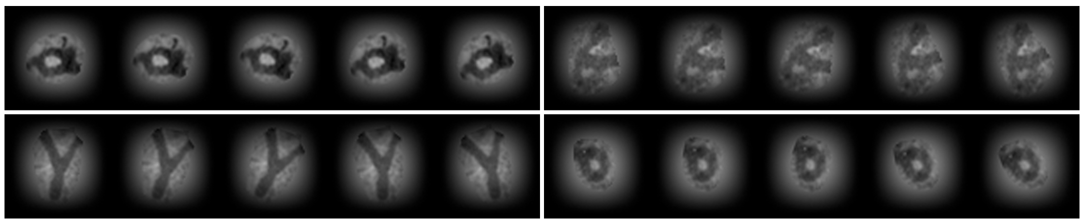


Figure 4.16: Archimedes character example – training images for the $c = 4$ classes. For all classes of characters, each of the 5 images is rotated by a multiple of 10 degrees ([0,-1,-2,+1,+2] as shown).

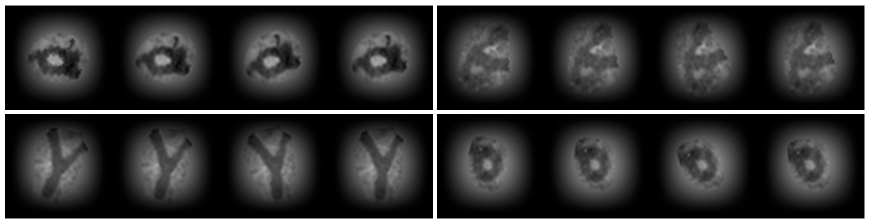


Figure 4.17: Archimedes character example – test images for the $c = 4$ classes. For all classes of characters, each of the 4 images is rotated by a multiple of 5 degrees ([-3,-1,+3,+1] as shown).

extracted and rotated for four different classes (‘α’, ‘ϵ’, ‘γ’, and ‘o’). Training and test character sets were created by varying the rotation of the character and applying an apodizing window. Both the Corefaces and LSCF algorithms produce sharp, visible correlation peaks for the true-class character. However, the high-noise floor produced by the LSCF algorithm for false-class inputs decreases the PSR, making the discrimination task easier for the classifier (Figure 4.20). While this example is intended to show the concept underlying the LSCF design, Chapter 5 presents results that demonstrate the usefulness of this algorithm in lowering the false alarm rate.

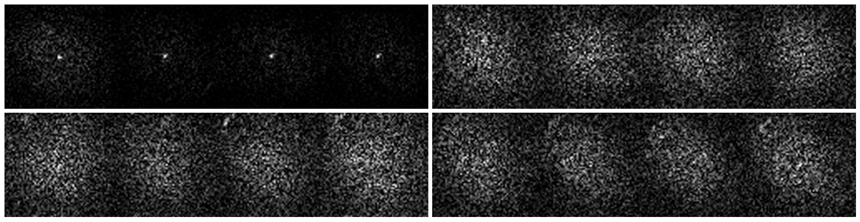


Figure 4.18: Archimedes character example – resulting correlation planes (shown as images) for test images from the first class the Corefaces algorithm, i.e., the phase was reconstructed with the each of the 4 PC subspaces.

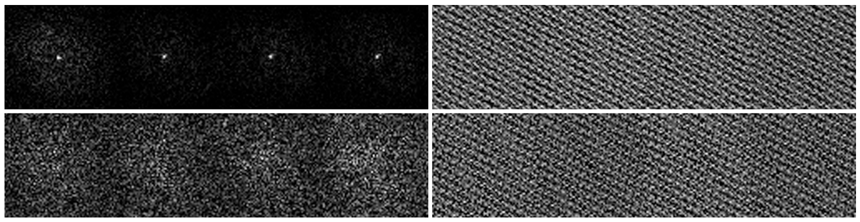


Figure 4.19: Archimedes character example – resulting correlation planes (shown as images) for test images from the first class using LSCF algorithm.

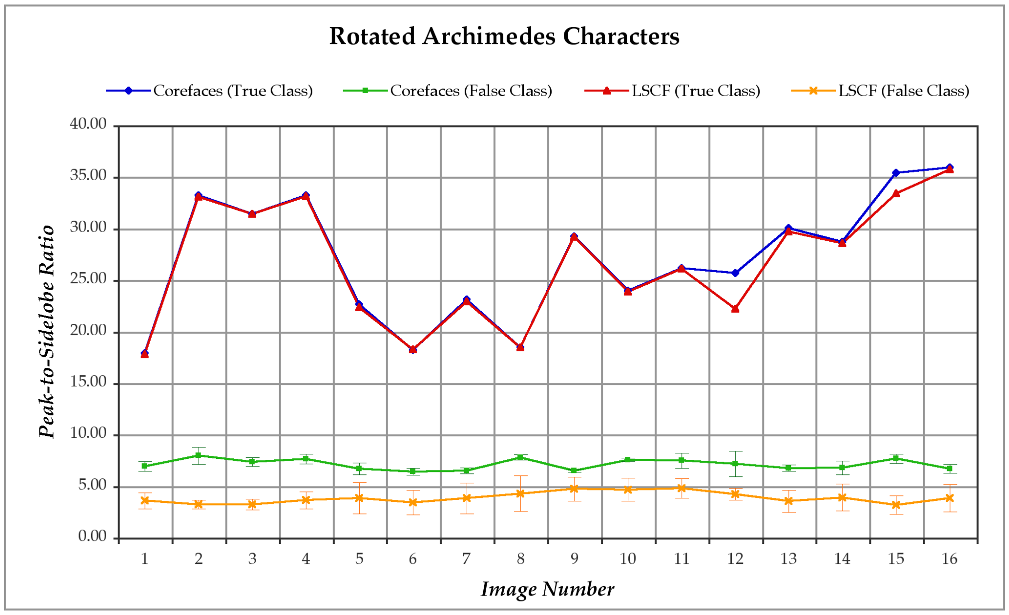


Figure 4.20: Archimedes character example – comparison of the true-class PSR and mean false-class PSR values for all 16 images using both the Corefaces and LSCF algorithm.

–Sixty percent of the time, it works every time.

Anchorman: The Legend of Ron Burgandy (2004)

5 Results

This chapter presents the results for the three objective categories defined in Chapter 2 and further examined in Chapter 4. Though results from the first two sections pertain to work completed for the Archimedes Palimpsest Project, additional outcomes from other applications are discussed to demonstrate the flexibility of the CPR design. The effectiveness of the LSCF algorithm is explored in the last section of this chapter, completing the objectives of this body of work and providing a framework for future research.

5.1 Correlation Applied to the Archimedes Palimpsest Imagery

The ability of a correlation filter to provide consistent, easily discernable peaks for accurate classification depends on the spatial characteristics of the input. For many applications, including the Archimedes Palimpsest, it is necessary to accommodate a variety of backgrounds that may surround or obscure the pattern-of-interest. Many applications require that the output correlation plane not only specify *where the target* *is*, but maintain high discrimination between classes to determine *what the target is*, which is also relevant for the palimpsest. While several correlation filters discussed in Chapter 3 have been characterized for specific attributes (e.g., noise tolerance [50]), a collection of confusion matrices is now discussed to illustrate how CPR is used to identify characters on various areas of parchment containing the text of Archimedes' work.

Figures 5.1-5.5 show confusion matrices calculated using the *clean test set* of characters, which were manually extracted from unobstructed regions of the palimpsest in similar fashion to the collection of the training exemplars. To produce these results, four classes have been omitted from both the training and test sets due to difficulty in finding and extracting these characters ('β', 'ψ', 'ξ', and 'ζ') from "clean" regions in the available images. The results for this independent data set show the performance of individual and multiple correlation filters used to achieve character classification in areas of the manuscript that are relatively easy to decipher. It should be noted that the uninteresting case of matching the training set to itself produced overall recognition rates of 100% for the correlation schemes that are typically applied to the palimpsest imagery.

The confusion matrix in Figure 5.1 shows classification results for the *clean test set* of characters, consisting of 200 images, using the locally nonlinear matched filter. The LNMF is of classical design, thereby requiring the use of N_k correlation filters for each class, where N_k is the number of training exemplars in class k. Using the peak-to-sidelobe ratio to characterize these peaks yields feature vectors consisting of $L = 10$ descriptors. PCA was then employed to reduce and possibly remove the effects of filters that performed poorly by retaining only 95% of the variability between feature vectors. Though characters such as 'δ' and 'ω' show a high percentage of misclassification, the overall recognition rate for this data set was 78%.

Figure 5.2 shows a confusion matrix for an identical set of parameters, though the MACH filter has been selected to produce the correlation results. The MACH filter is a composite filter, which produces only a single filter and correlation plane for the training exemplars from a particular class. If only one performance metric is used to characterize the peak, as is the case here, then the feature vector consists of only one element, and PCA is useless. Comparing Figure 5.2 to Figure 5.1, the simplicity of the MACH filter composite design (one filter as opposed to ten) and its inherent tolerance to within-class distortion did not produce a higher overall recognition rate than that of the LNMF. Classification of the characters 'δ' and 'o' have marginally improved, while every test image containing 'η' or 'ι' was misclassified. The respective kappa coefficients using the individual LNMF and MACH filter were 76.3% and 66.3%, which produces a z-score of $z = 2.17$ after calculating the variance σ_κ^2. This z-score indicates that the LNMF should produce more accurate classification of clean characters than the MACH filter at a confidence level of 98.5%.

Figure 5.3 shows improved classification when combining the LNMF and MACH filter to produce correlation planes for feature vector generation. Using PCA to essentially "filter out" any poor filter performance, the overall recognition rate improved 81%. Using z-scores from the variance in kappa, we find that for this data set, the performance of this combined filter approach only improves upon the performance of the individual LNMF at a confidence level of 77.0%. Figure 5.4 shows results for an identical set of parameters, but the input test character is assumed to be classified correctly if the true class is among the three most probable classes. Allowing a "misclassification tolerance" of two classes shows valuable results; as mentioned earlier, the goal is not to produce a single classification result as in typical target detection problems, but to provide intermediate results to allow the user to apply his or her own decisions to arrive at a conclusion. For example, the scholar may use the five most probable character classes at each of the selected ROIs to transcribe the full word under inspection. Figure 5.4 shows that a recognition rate of 93% was achieved when the three most probable classes were observed, indicating that most of the misclassified characters exhibited a higher likelihood than 85% (17 out of 20) of the classes.

The confusion matrix in Figure 5.5 is included to emphasize the importance of PCA in the recognition chain. The only parameter to change between Figure 5.4 and Figure 5.5 was the use of PCA to reduce (and possibly remove) the effects of "noisy" features, which do not contribute to classification. If one of the LNMFs performed poorly due to specific attributes of the pattern or background (e.g., high within-class distortion, clutter, etc.), the corresponding features that the filter produced may be relatively similar for each class (low variance). The dimensionality of the feature vectors will have increased, and very little "information" will have been added to discriminate classes, essentially increasing the similarity between class feature vectors. This effect is apparent in the reduced recognition rate after removing PCA, from 93% to 82%. A standard z-score of 3.51 is obtained using kappa statistics; this indicates that when using the LNMF-MACH approach, PCA improves the classification of clean characters at a confidence level greater than 99.9%.

Comparing Figure 5.6 to Figure 5.1, the overall recognition rate dropped from 78% to 56% after introducing a more difficult set of test images. Characters such as 'γ' and 'κ' that were classified correctly for each image in the *clean test set* now show poor classification. The performance of the MACH filter applied to obscured and clean characters (Figure 5.7 and Figure 5.2) shows a similar result; the overall recognition rate decreases from 68% to 49%, and several character classes were classified incorrectly ('δ', 'ι', 'o', etc.). While the ability of the MACH filter to provide within-class distortion tolerance may not appear obvious, it is important to consider the fact the only one filter mask is used to detect the occurrence of all possible variations of a particular class. Moreover, for highly obscured characters, the characteristically broad correlation peak produced by the MACH filter provides better discrimination capabilities than the other filters discussed in this work.

Again combining the LNMF and MACH filtering schemes to provide correlation results for feature generation (Figure 5.8), we find the overall recognition rate improves only slightly over the use of the individual MACH filter, from 49% to 55%; the rate produced by the LNMF was almost unchanged. This result suggests that the character classified as most probable depended more on obtaining an accurate representation of the character than on the use of a composite mask designed to generalize well. Moreover, this result is almost negligible when considering the corresponding kappa coefficients. Allowing a misclassification tolerance of two classes produced the confusion matrix in Figure 5.9, which shows a dramatic improvement in the overall recognition rate, from 55% to 80%.

The kappa coefficient produced for the case with PCA (Figure 5.9) and without PCA (Figure 5.10) are almost identical. Thus, using PCA to reduce and/or remove poor features is more beneficial when classifying clean characters. This result makes sense, as clean data will produce features with high variance due to better class discrimination.

Assuming these results can be extrapolated for the entire manuscript, the CPR system should be capable of producing classification results that contain the true-class character among the three most probable classes for approximately four out of every five ROIs. This statement also assumes that the ROIs selected share similar spatial characteristics to those described by this obscured set of characters. Obviously, some method of improvement is desired (Section 5.2.1), but achieving these results is promising when considering the number of classes and the variations in the characters.

	Parameters:			*c = 20 Classes*			*Clean Test Set*			*Misclassification Tol. = 0*			*LNMF*			*PSR*			*PCA - 95%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
χ	0%	90%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	0%	0%	20%	0%	0%	10%	0%	20%	10%	0%	10%	0%	0%	10%	0%	0%	0%	0%	20%	0%	10
ε	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10
η	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	0%	10%	0%	50%	0%	0%	0%	10%	0%	0%	0%	0%	10%	0%	0%	0%	20%	10
κ	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	50%	50%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	10%	10
ω	30%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	20%	20%	0%	20%	0%	0%	0%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	40%	0%	0%	10%	30%	20%	0%	0%	10
ϕ	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	20%	0%	10%	0%	0%	50%	0%	10%	0%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	10%	0%	0%	20%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	60%	10
Overall																					77.5%

Figure 5.1: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 77.5% with a kappa coefficient of 76.3%.

		Parameters:		*c = 20 Classes*			*Clean Test Set*			*Misclassification Tol. = 0*			*MACH*			*PSR*			*PCA - 100%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	50%	0%	0%	0%	0%	0%	0%	20%	0%	0%	0%	30%	0%	0%	0%	0%	0%	0%	0%	0%	10
χ	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	0%	0%	30%	0%	10%	0%	0%	10%	20%	0%	0%	0%	0%	10%	0%	10%	0%	0%	10%	0%	10
ε	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10
η	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
γ	0%	30%	0%	0%	0%	70%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	10%	0%	0%	0%	20%	0%	0%	0%	0%	0%	50%	0%	10%	0%	0%	10%	0%	10
κ	0%	10%	0%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	70%	10%	0%	0%	0%	0%	0%	0%	0%	20%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	60%	0%	0%	0%	0%	30%	0%	0%	0%	0%	10
ω	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	0%	10%	20%	0%	0%	0%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	30%	0%	0%	0%	40%	20%	0%	0%	0%	10%	0%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	10
σ	0%	0%	0%	60%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	30%	10%	0%	0%	10
τ	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	70%	10
Overall																					68.0%

Figure 5.2: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – MACH filter, and PSR. The calculated recognition rate over the set of $c = 20$ classes was 68.0% with a kappa coefficient of 66.3%.

	Parameters:			*c = 20 Classes*			*Clean Test Set*			*Misclassification Tol. = 0*			*LNMF, MACH*			*PSR*			*PCA - 95%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	90%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
χ	0%	90%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	10%	10%	20%	0%	10%	10%	0%	30%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	0%	0%	10
η	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	0%	10%	0%	50%	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	20%	10
κ	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	60%	40%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	10%	0%	0%	0%	10%	10
ω	20%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	50%	0%	0%	20%	0%	0%	0%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	0%	0%	10%	20%	0%	0%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	10%	0%	0%	0%	10%	50%	0%	20%	0%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	10%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	10
Overall																					80.5%

Figure 5.3: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 80.5% with a kappa coefficient of 79.5%.

	Parameters:			*c = 20 Classes*			*Clean Test Set*			*Misclassification Tol. = 2*			*LNMF, MACH*			*PSR*			*PCA - 95%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
χ	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	10%	10%	40%	0%	10%	10%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
η	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	0%	0%	0%	70%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	20%	10
κ	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ω	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	10%	0%	0%	0%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	10%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	10
Overall																					93.0%

Figure 5.4: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 93.0% with a kappa coefficient of 92.6% after allowing a misclassification tolerance of 2 classes.

			Parameters:	*c = 20 Classes*			*Clean Test Set*			*Misclassification Tol. = 2*			*LNMF, MACH*			*PSR*			*PCA - 100%*		
	α	**χ**	**δ**	**ε**	**η**	**γ**	**ι**	**κ**	**λ**	**μ**	**ν**	**ω**	**o**	**ϕ**	**π**	**ρ**	**σ**	**τ**	**θ**	**υ**	*Images*
α	90%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
χ	0%	90%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	10%	10%	20%	0%	10%	10%	0%	30%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10
η	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	0%	10%	0%	50%	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	20%	10
κ	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	60%	40%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	10%	0%	0%	0%	10%	10
ω	20%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	50%	0%	0%	20%	0%	0%	0%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	0%	0%	10%	20%	0%	0%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	10%	0%	0%	0%	0%	60%	0%	20%	0%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	10%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	10
Overall																					81.5%

Figure 5.5: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, and PSR. The calculated recognition rate over the set of $c = 20$ classes was 81.5% with a kappa coefficient of 80.5% after allowing a misclassification tolerance of 2 classes.

	Parameters:			*c = 20 Classes*			*Obscured Test Set*			*Misclassification Tol. = 0*			*LNMF*			*PSR*			*PCA - 95%*		
	α	**χ**	**δ**	**ε**	**η**	**γ**	**ι**	**κ**	**λ**	**μ**	**ν**	**ω**	**o**	**ϕ**	**π**	**ρ**	**σ**	**τ**	**θ**	**υ**	*Images*
α	60%	0%	0%	0%	0%	10%	0%	10%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	10
χ	0%	60%	0%	0%	0%	40%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	0%	0%	20%	0%	0%	10%	0%	10%	30%	0%	0%	0%	0%	30%	0%	0%	0%	0%	0%	0%	10
ε	10%	0%	0%	50%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	20%	10%	0%	0%	10
η	0%	0%	0%	0%	80%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	10%	20%	30%	0%	0%	10%	0%	10%	0%	0%	10%	0%	0%	0%	10%	0%	0%	10
ι	0%	0%	0%	0%	10%	10%	30%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	30%	0%	0%	10
κ	0%	0%	0%	0%	30%	0%	0%	30%	20%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	10%	10
λ	0%	0%	0%	10%	0%	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	30%	0%	50%	10%	0%	0%	0%	0%	10%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ω	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	50%	0%	0%	10%	10%	0%	10%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	30%	0%	10%	30%	0%	10%	10%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	70%	0%	10%	10%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10%	0%	0%	50%	0%	20%	10%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	20%	10%	0%	60%	10%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10%	10%	0%	0%	10%	60%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	10%	0%	0%	10%	70%	0%	10
υ	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	20%	0%	0%	0%	0%	0%	0%	10%	0%	50%	10
Overall																					55.5%

Figure 5.6: Confusion matrix for the *obscured test set* of characters from the text of Archimedes using the following parameters – LNMF, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 55.5% with a kappa coefficient of 53.2%.

Parameters:				*c = 20 Classes*			*Obscured Test Set*			*Misclassification Tol. = 0*			*MACH*			*PSR*			*PCA - 100%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	30%	0%	0%	0%	0%	0%	0%	20%	0%	20%	0%	10%	0%	20%	0%	0%	0%	0%	0%	0%	10
χ	0%	80%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10
δ	10%	0%	10%	0%	0%	0%	0%	20%	10%	0%	0%	0%	0%	40%	0%	10%	0%	0%	0%	0%	10
ε	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	10
η	0%	0%	0%	0%	30%	0%	0%	50%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	10
γ	0%	10%	0%	0%	0%	50%	0%	30%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10
ι	0%	0%	0%	10%	0%	0%	10%	20%	0%	0%	0%	0%	0%	30%	0%	0%	0%	30%	0%	0%	10
κ	0%	0%	10%	10%	20%	0%	0%	60%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	20%	0%	0%	70%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	10%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	20%	0%	0%	0%	0%	10
ω	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	70%	0%	10%	10%	0%	0%	0%	0%	0%	10
o	10%	0%	0%	20%	0%	10%	0%	0%	10%	10%	0%	0%	0%	10%	0%	0%	0%	10%	10%	10%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	0%	10%	10
π	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	60%	0%	0%	20%	0%	0%	10
ρ	0%	0%	0%	10%	0%	0%	0%	10%	0%	40%	0%	0%	0%	10%	0%	10%	0%	10%	10%	0%	10
σ	0%	0%	0%	30%	0%	0%	0%	0%	0%	0%	0%	10%	0%	10%	0%	0%	40%	10%	0%	0%	10
τ	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	10%	0%	10%	60%	0%	0%	10
θ	0%	0%	0%	30%	0%	0%	0%	0%	10%	0%	0%	0%	0%	20%	0%	0%	0%	0%	40%	0%	10
υ	0%	10%	0%	10%	0%	0%	0%	10%	0%	0%	20%	0%	0%	20%	0%	0%	0%	0%	0%	30%	10
Overall																					49.0%

Figure 5.7: Confusion matrix for the *obscured test set* of characters from the text of Archimedes using the following parameters – MACH filter, and PSR. The calculated recognition rate over the set of $c = 20$ classes was 46.3% with a kappa coefficient of 53.2%.

	Parameters:			*c = 20 Classes*			*Obscured Test Set*			*Misclassification Tol. = 0*			*LNMF, MACH*			*PSR*			*PCA - 95%*		
	α	χ	δ	ε	η	γ	ι	κ	λ	μ	ν	ω	o	ϕ	π	ρ	σ	τ	θ	υ	*Images*
α	60%	0%	0%	0%	0%	10%	0%	10%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	10
χ	0%	60%	0%	0%	0%	40%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	0%	0%	20%	0%	0%	10%	0%	10%	30%	0%	0%	0%	0%	30%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	50%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	20%	10%	10%	0%	10
η	0%	0%	0%	0%	70%	0%	0%	10%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	10%	20%	30%	0%	0%	0%	0%	20%	0%	0%	10%	0%	0%	0%	10%	0%	0%	10
ι	0%	0%	0%	0%	10%	10%	30%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	30%	0%	0%	10
κ	0%	0%	0%	10%	30%	0%	0%	30%	10%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	10%	10
λ	0%	0%	0%	10%	0%	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	30%	0%	50%	10%	0%	0%	0%	0%	10%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ω	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	50%	0%	0%	10%	10%	0%	10%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	30%	0%	10%	40%	0%	10%	10%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	70%	0%	10%	10%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10%	0%	0%	50%	0%	20%	10%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	20%	10%	0%	60%	10%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10%	10%	0%	0%	10%	60%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	10%	0%	0%	10%	70%	0%	10
υ	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	20%	0%	0%	0%	0%	0%	0%	10%	0%	50%	10
Overall																					55.0%

Figure 5.8: Confusion matrix for the *obscured test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 55.0% with a kappa coefficient of 52.6%.

		Parameters:		*c = 20 Classes*			*Obscured Test Set*			*Misclassification Tol. = 2*			*LNMF, MACH*			*PSR*			*PCA - 95%*		
	α	**χ**	**δ**	**ε**	**η**	**γ**	**ι**	**κ**	**λ**	**μ**	**ν**	**ω**	**o**	**ϕ**	**π**	**ρ**	**σ**	**τ**	**θ**	**υ**	*Images*
α	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	10
χ	0%	80%	0%	0%	0%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	0%	0%	30%	0%	0%	10%	0%	0%	30%	0%	0%	0%	0%	30%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	10%	0%	10
η	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	0%	10%	80%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
ι	0%	0%	0%	0%	10%	0%	60%	0%	0%	0%	0%	0%	0%	0%	0%	20%	0%	10%	0%	0%	10
κ	0%	0%	0%	0%	10%	0%	0%	90%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	10%	0%	70%	10%	0%	0%	0%	0%	10%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ω	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	70%	0%	0%	10%	10%	0%	10%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	80%	0%	10%	0%	0%	0%	10%	0%	10
ϕ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	10%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10%	0%	0%	70%	0%	10%	0%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	10%	0%	80%	0%	0%	0%	10
τ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	90%	0%	0%	10
θ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	10
υ	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	20%	0%	0%	0%	0%	0%	0%	10%	0%	60%	10
Overall																					80.0%

Figure 5.9: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, PSR, and PCA to retain 95% of the variability in the feature vector. The calculated recognition rate over the set of $c = 20$ classes was 80.0% with a kappa coefficient of 78.9% after allowing a misclassification tolerance of 2 classes.

	Parameters:			*c = 20 Classes*			*Obscured Test Set*			*Misclassification Tol. = 2*			*LNMF, MACH*			*PSR*			*PCA - 100%*	
	α	**χ**	**δ**	**ε**	**η**	**γ**	**ι**	**κ**	**λ**	**μ**	**ν**	**ω**	**o**	**ϕ**	**π**	**ρ**	**σ**	**τ**	**θ**	**υ**	*Images*
α	80%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	10
χ	0%	80%	0%	0%	0%	10%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
δ	10%	0%	60%	20%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10
ε	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
η	0%	0%	0%	0%	90%	0%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
γ	0%	0%	0%	10%	10%	60%	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
ι	0%	0%	0%	0%	10%	0%	50%	0%	20%	0%	10%	0%	0%	0%	0%	0%	0%	10%	0%	0%	10
κ	10%	0%	0%	0%	10%	0%	0%	50%	20%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	10
λ	0%	0%	0%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	0%	0%	0%	10%	10%	0%	0%	10
μ	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ν	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10
ω	0%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	80%	0%	0%	0%	0%	0%	10%	0%	0%	10
o	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%	10
ϕ	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	10%	0%	0%	80%	0%	0%	0%	0%	0%	0%	10
π	0%	0%	0%	10%	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	70%	0%	0%	10%	0%	0%	10
ρ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%	0%	0%	50%	0%	30%	10%	0%	10
σ	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%	10
τ	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	0%	10
θ	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	90%	0%	10
υ	10%	0%	0%	0%	0%	0%	0%	0%	10%	0%	30%	0%	0%	0%	0%	0%	0%	0%	0%	50%	10
Overall																					78.0%

Figure 5.10: Confusion matrix for the *clean test set* of characters from the text of Archimedes using the following parameters – LNMF and MACH filter, and PSR. The calculated recognition rate over the set of $c = 20$ classes was 78.0% with a kappa coefficient of 76.8% after allowing a misclassification tolerance of 2 classes.

5.2 Application of the Pattern Recognition Tool

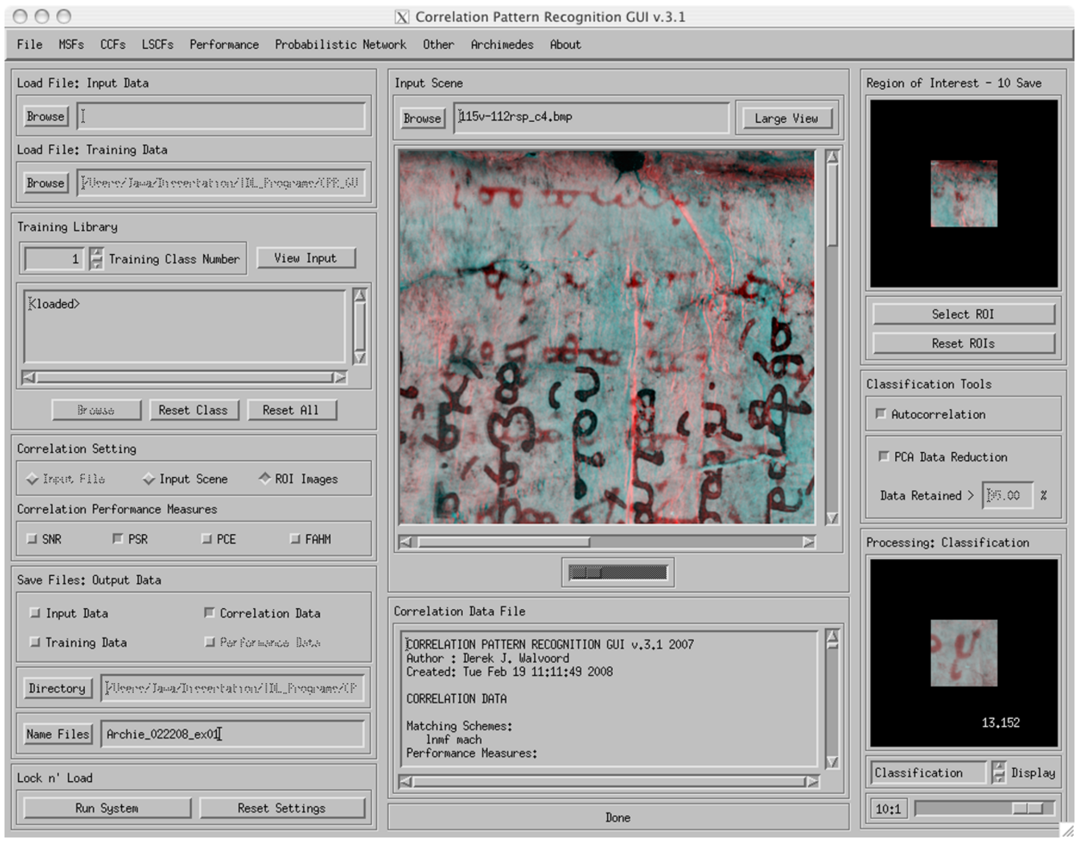
This section examines the use of the pattern recognition tool on the Archimedes Palimpsest. Two additional applications are included to demonstrate the flexibility of the CPR design: a fifteenth-century. Hebrew colophon and fiducial-marker detection in 3-dimensional MRI breast imaging.

5.2.1 Archimedes Palimpsest Character Recognition

Using the results from Section 5.1 as a foundation for the approach to recognize hand-written characters, we now demonstrate how the current pattern recognition tool can be applied to images of the Archimedes Palimpsest. To improve upon the results obtained from spatial correlation methods, the probabilistic network discussed in Section 4.2.2 was integrated into the system, providing three sources of information: the images of the palimpsest, the word dictionary look-up table (LUT) generated from Reviel Netz's partial transcription, and most importantly, the user's knowledge of the context and language. While Section 5.1 provides quantitative results for the correlation pattern recognition system, we are now restricted to qualitative discussion using examples.

Figure 5.11 shows the CPR interface used to select training data, ROIs, and applicable correlation filters for computing classification results. In the following example [51], ROIs have been selected from a word consisting of ten characters, some of which lack visual contrast with the background parchment. Once these character probabilities (classification results) have been computed at each ROI using a series of correlation algorithms, a GUI for the probabilistic network is created. Figure 5.12 shows the interface generated for this word after initializing the network using statistics from Reviel Netz's transcription and the CPR results.

On the right-hand side of Figure 5.12, the change in the probabilities for each query variable is shown after hard evidence was applied to the second, eighth, and tenth character in word. The most probable character for each of these ROIs is initially displayed. At any time, even before computing a query, the user can view the words in the LUT in order of likelihood. Figure 5.13 shows the five most probable word results for this example after performing the query above. The first result here is the correct word for this palimpsest region.



Input Scene: 115v-112rsp_c4.bmp

Training Class Number: 1

<loaded>

Name Files: Archie_022208_ex01

Data Retained > 95.00 %

13.152

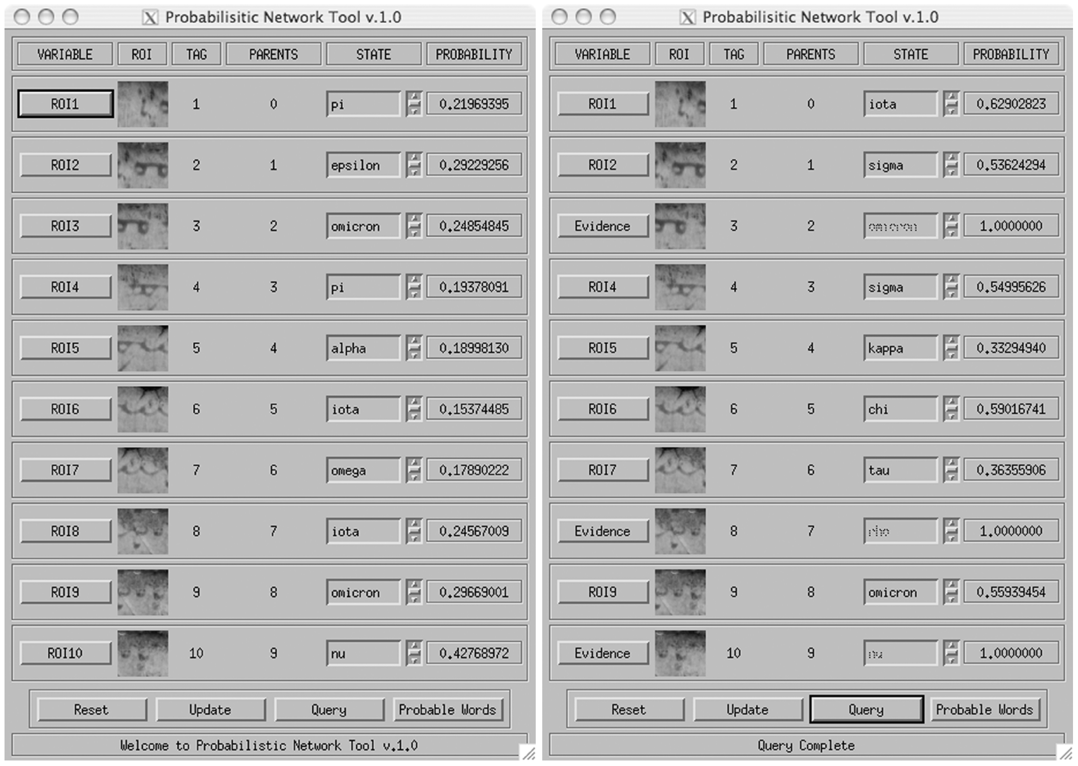
10:1

```
CORRELATION PATTERN RECOGNITION GUI v.3.1 2007
Author : Derek J. Walvoord
Created: Tue Feb 19 11:11:49 2008

CORRELATION DATA

Matching Schemes:
   lnmf mach
Performance Measures:
```

Figure 5.11: The character recognition interface used to generate CPR results. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

The user can now base transcription decisions on the information from the palimpsest images, the word dictionary LUT created from the partial transcription, and most importantly, his or her knowledge of the context and language.

Figure 5.14 attempts to summarize this entire decision-making process to transcribe a degraded word region. As shown, eight ROIs have been selected, which correspond to the "approximate" locations of the eight characters that constitute the word. The first table shows the resulting CPR probabilities of the five most probable character classes for each of the eight ROIs. Note that $c = 36$ character classes were present in the training set. Thus, for ROI5, CPR results show 'α' as the most prob-



Probabilisitic Network Tool v.1.0

VARIABLE	ROI	TAG	PARENTS	STATE	PROBABILITY
ROI1		1	0	pi	0.21969395
ROI2		2	1	epsilon	0.29229256
ROI3		3	2	omicron	0.24854845
ROI4		4	3	pi	0.19378091
ROI5		5	4	alpha	0.18998130
ROI6		6	5	iota	0.15374485
ROI7		7	6	omega	0.17890222
ROI8		8	7	iota	0.24567009
ROI9		9	8	omicron	0.29669001
ROI10		10	9	nu	0.42768972

Reset Update Query Probable Words

Welcome to Probabilistic Network Tool v.1.0

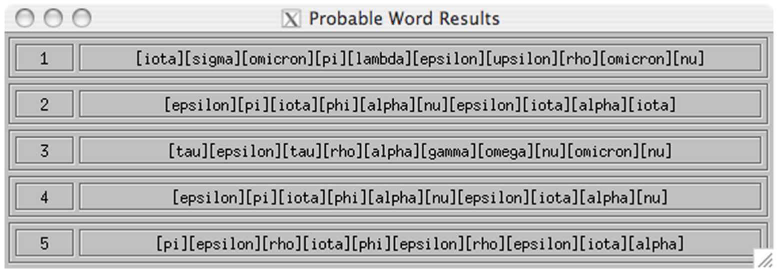
Probabilisitic Network Tool v.1.0

VARIABLE	ROI	TAG	PARENTS	STATE	PROBABILITY
ROI1		1	0	iota	0.62902823
ROI2		2	1	sigma	0.53624294
Evidence		3	2	omicron	1.0000000
ROI4		4	3	sigma	0.54995626
ROI5		5	4	kappa	0.33294940
ROI6		6	5	chi	0.59016741
ROI7		7	6	tau	0.36355906
Evidence		8	7	rho	1.0000000
ROI9		9	8	omicron	0.55939454
Evidence		10	9	nu	1.0000000

Reset Update Query Probable Words

Query Complete

Figure 5.12: The probabilistic network tool before (Left) and after (Right) adding evidence and inferencing a query. *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

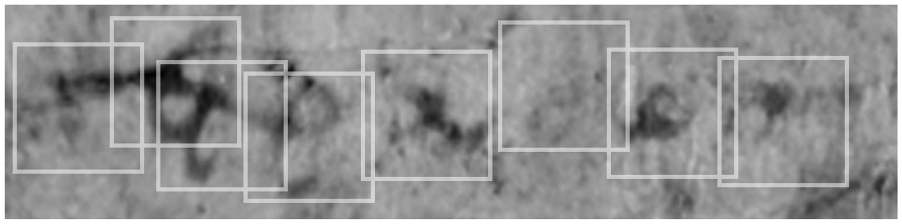


Probable Word Results

1	[iota][sigma][omicron][pi][lambda][epsilon][upsilon][rho][omicron][nu]
2	[epsilon][pi][iota][phi][alpha][nu][epsilon][iota][alpha][iota]
3	[tau][epsilon][tau][rho][alpha][gamma][omega][nu][omicron][nu]
4	[epsilon][pi][iota][phi][alpha][nu][epsilon][iota][alpha][nu]
5	[pi][epsilon][rho][iota][phi][epsilon][rho][epsilon][iota][alpha]

Figure 5.13: The resulting probable words (in order) from the word dictionary LUT.

ROI Selection



Correlation Pattern Recognition Results

ROI 1		ROI 2		ROI 3		ROI 4		ROI 5		ROI 6		ROI 7		ROI 8	
τ	**6.56**	θ	9.15	θ	8.77	ω	6.88	**α**	**10.03**	κ	7.38	**ο**	**8.23**	**σ**	**5.07**
κ	5.54	ξ	5.57	ξ	6.56	ζ	5.92	Η	8.57	Γ	6.98	Ρ	6.57	θ	4.84
λ	5.24	ο	5.53	ο	5.57	ξ	5.49	κ	6.67	π	6.07	ν	5.07	τ	4.82
μ	4.49	**ε**	**5.10**	Ρ	5.16	ε	5.42	Κ	6.65	τ	4.71	ζ	4.58	π	4.66
α	4.23	Ρ	5.04	ε	4.97	ν	4.97	Α	5.68	λ	4.35	ξ	4.17	ο	4.45

Initialized Network Results

ROI 1		ROI 2		ROI 3		ROI 4		ROI 5		ROI 6		ROI 7		ROI 8	
τ	**25.06**	**ε**	**37.58**	ι	17.36	ε	18.85	**α**	**53.33**	τ	39.10	**ο**	**24.25**	**σ**	**30.61**
α	19.29	ο	13.58	θ	16.76	**ρ**	**13.57**	ω	8.53	ν	9.53	ω	22.90	ν	22.02
μ	13.17	π	7.92	α	13.26	ι	12.20	ε	5.65	σ	7.99	α	16.69	ι	20.35
ε	10.58	ρ	6.75	γ	12.83	μ	11.15	λ	4.99	α	6.01	τ	7.02	α	9.31
κ	9.04	μ	4.29	ο	8.90	ο	10.91	ο	4.73	ι	5.48	ι	6.58	ω	7.57

Query Results

ROI 1		ROI 2		ROI 3		ROI 4		ROI 5		ROI 6		ROI 7		ROI 8	
τ Evidence Applied		**ε**	**92.82**	**τ**	**52.99**	ρ Evidence Applied		**α**	**90.80**	τ	51.25	**ο**	**26.69**	**σ**	**30.91**
		ο	7.18	γ	47.01			ο	7.67	ν	9.06	ω	25.84	ν	23.89
								ε	1.53	γ	7.95	α	18.85	ι	21.06
										υ	6.05	ε	6.26	ω	7.39
										λ	4.99	ρ	5.04	υ	6.62

Probable Words

1. τετραγων
2. τετραδοσ
3. τετραπλα

Figure 5.14: Example of the entire decision-making process showing intermediate results at each stage. Probabilities are shown for the most likely character classes at each selected ROI. Character classes shown in bold are correct . *(Photographs produced by The Rochester Institute of Technology and John Hopkins University. Copyright resides with the owner of the Archimedes Palimpsest.)*

able character with a 10.03% likelihood over the remaining 35 classes. The second table shows updated probabilities after initializing the probabilistic network. In this example, hard evidence is now applied to ROI 1 (τ) and ROI 4 (ρ) based on character likelihood and visual information from the image itself. The query results show a large increase in probability for the "correct" character classes for all but ROI 6. At this point, the user can either make an informed decision from the probable words from the LUT or perform another query.

Quite certainly, the most important result is in how the overall pattern recognition system assists the end-user in the classification task. Reviel Netz, the principal scholar in transcribing the Archimedes text, has commented on the tool, saying,

"I was much surprised when the machine actually did paleographic work... we gradually moved to more and more difficult areas [of the palimpsest] until we looked at fragmentary words for which I had guesses, no more. The machine reached my guesses independently. This means either that I think like the machine and that we both can be outperformed by better paleographers, or that the machine is in fact making informed paleographic judgements."

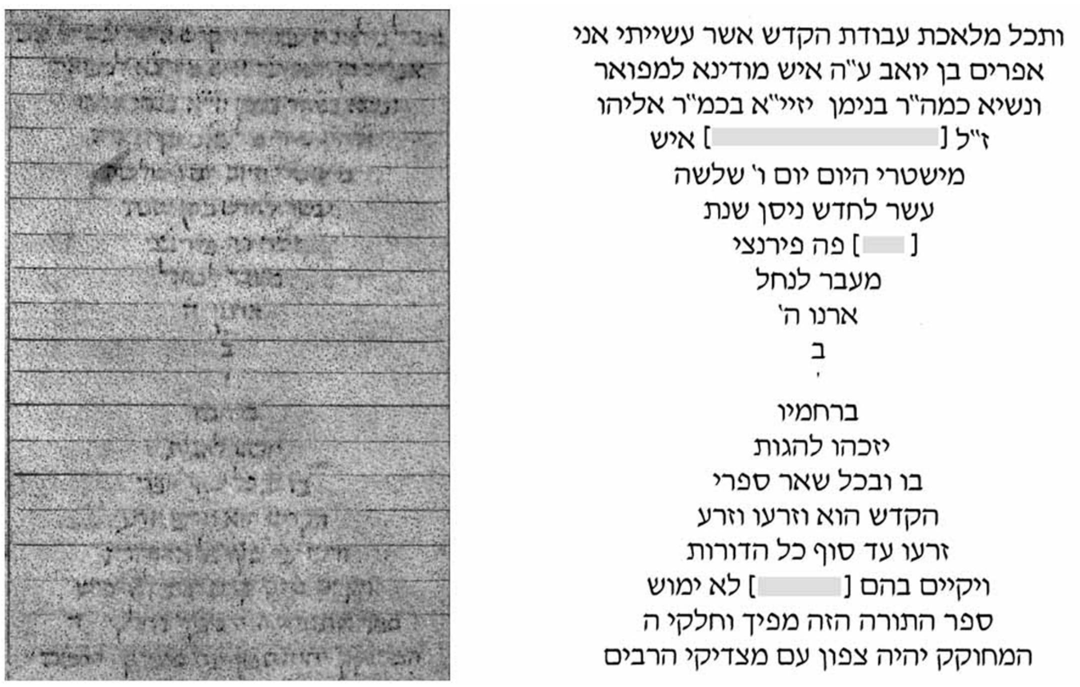
At a conference of scholars in London, the tool was also deemed useful for the purpose of transcribing the text of Hypereides contained within the Archimedes Palimpsest. Natalie Tchernetska of Cambridge University has commented, saying,

"I think the work has great potential, not only for Hypereides pages, but also for similar projects in the future."

5.2.2 Recovering Information from a Hebrew Colophon

While the probabilistic network integrated into the recognition tool was designed for use on the text of Archimedes, the ability to match spatial patterns makes the CPR system useful for many applications. In 2005, Walvoord, et al., [53] demonstrated the use of the system in the recovery of information from a fifteenth-century Hebrew prayer book containing an erased *colophon*. The content of the colophon of a codex is of vital interest to historians; it contains facts about the publication of the book, such as the names of the patron and scribe, the date of publication, the city where the book was copied, etc.

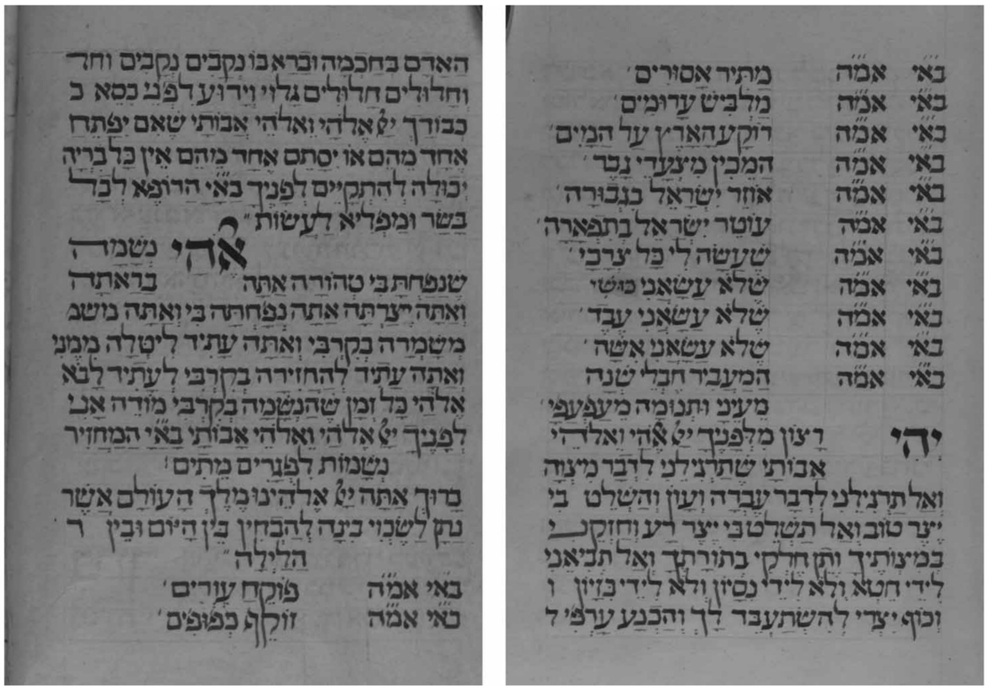
The Hebrew colophon of a Florentine Siddur had been partially transcribed by Evelyn Cohen of the Jewish Theological Seminary of America (JTSA) prior to any imaging. After multiple imaging sessions, digital contrast enhancement provided suf-



ותכל מלאכת עבודת הקדש אשר עשייתי אני
אפרים בן יואב ע"ה איש מודינא למפואר
ונשיא כמה"ר בנימן יזיי"א בכמ"ר אליהו
ז"ל [ ] איש
מישטרי היום יום ו' שלשה
עשר לחדש ניסן שנת
[ ] פה פירנצי
מעבר לנחל
ארנו ה'
ב
'
ברחמיו
יזכהו להגות
בו ובכל שאר ספרי
הקדש הוא וזרעו וזרע
זרעו עד סוף כל הדורות
ויקיים בהם [ ] לא ימוש
ספר התורה הזה מפיך וחלקי ה
המחוקק יהיה צפון עם מצדיקי הרבים

Figure 5.15: (Left) An image of the colophon under longwave ultraviolet light ($\lambda = 365nm$) through an ultraviolet bandpass filter ($\lambda = 345nm$, $\Delta\lambda = 40nm$) and (Right) the transcription of the colophon obtained from the images. Three sections of text are uncertain, including the name of the home town, the patron, and the date.

ficiently clear text for Cohen to read nearly all of the colophon, as shown in the transcription in Figure 5.15. Her transcription indicates that the book was copied by the scribe Efraim ben Joab of Modena in Florence and completed on Friday, the thirteenth day of the month of Nisan, probably in the year 1487 CE. However, the transcription of three sections of text was less certain, as indicated by the gray bars in Figure 5.15. The unclear information includes a reference to the home town of the patron and the year in which the book was copied. This information is sufficiently important to the study of the provenance of the book that the CPR tool was trained to recognize handwritten Hebrew characters using several different features. The library of Hebrew reference characters was generated from images of two unerased pages of the Siddur that are assumed to have been copied by the same scribe (as was customary). The pages of



האדם בחכמה וברא בו נקבים נקבים וחד
וחלולים חלולים גלוי וידוע לפני כסא כ
כבודך יי אלהי ואלהי אבותי שאם יפתח
אחד מהם או יסתם אחד מהם אין כל בריה
יכולה להתקיים לפניך בא״י הרופא לכל
בשר ומפליא לעשות

אלהי נשמה
שנפחת בי טהורה אתה בראתה
ואתה יצרתה אתה נפחתה בי ואתה משמ
משמרה בקרבי ואתה עתיד ליטלה ממני
ואתה עתיד להחזירה בקרבי לעתיד לבא
אלהי כל זמן שהנשמה בקרבי מודה אני
לפניך יי אלהי ואלהי אבותי בא״י המחזיר
נשמות לפגרים מתים

ברוך אתה יי אלהינו מלך העולם אשר
נתן לשכוי בינה להבחין בין היום ובין ר
הלילה

בא״י אמ״ה פוקח עורים
בא״י אמ״ה זוקף כפופים

בא״י אמ״ה מתיר אסורים
בא״י אמ״ה מלביש ערומים
בא״י אמ״ה רוקע הארץ על המים
בא״י אמ״ה המכין מצעדי גבר
בא״י אמ״ה אוזר ישראל בגבורה
בא״י אמ״ה עוטר ישראל בתפארה
בא״י אמ״ה שעשה לי כל צרכי
בא״י אמ״ה שלא עשאני כושי
בא״י אמ״ה שלא עשאני עבד
בא״י אמ״ה שלא עשאני אשה
בא״י אמ״ה המעביר חבלי שנה
מעיני ותנומה מעפעפי

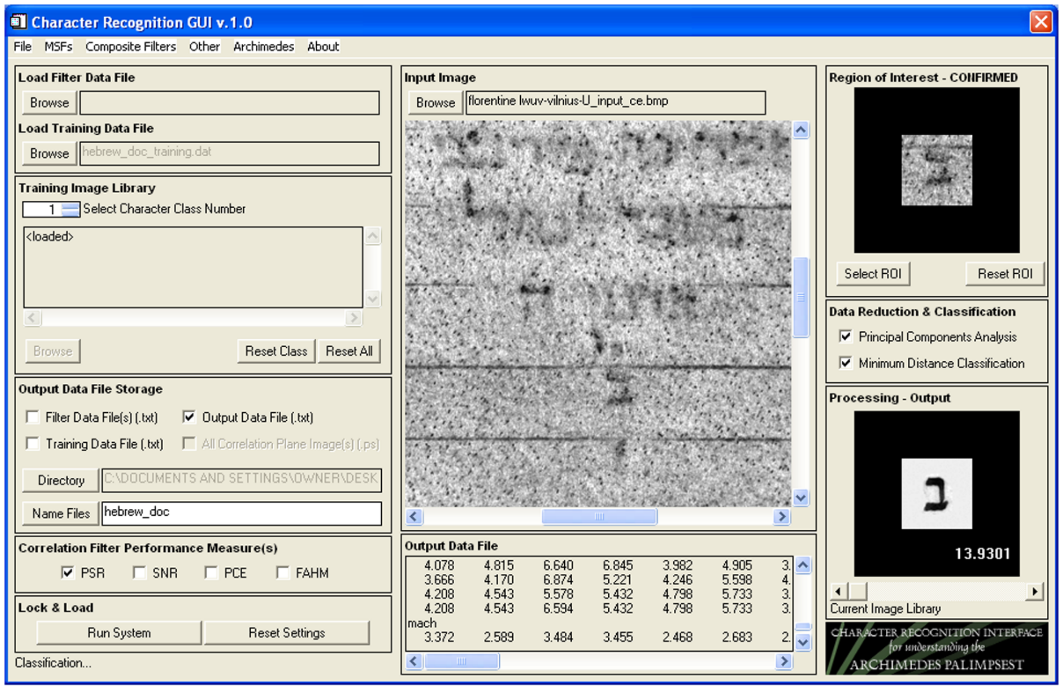
יהי רצון מלפניך יי אלהי ואלהי
אבותי שתרגילני לדבר מצוה
ואל תרגילני לדבר עבירה ועון והשלט בי
יצר טוב ואל תשלט בי יצר רע וחזקני
במצותיך ותן חלקי בתורתך ואל תביאני
לידי חטא ולא לידי נסיון ולא לידי בזיון ו
וכוף יצרי להשתעבד לך והכנע ערפי ל

Figure 5.16: (Left) and (Right) Images of unerased pages from the manuscript used to construct a library of Hebrew reference characters for correlation pattern recognition.

text from which the reference characters were collected are shown in Figure 5.16.

The CPR GUI is shown in Figure 5.17 with the center character of the colophon (the middle of the "hourglass") selected as the ROI from the background image. In this scenario, the most probable character determined by the system matched that selected by Evelyn Cohen in her original transcription. This process of obtaining classification probabilities for individual ROIs was performed at all locations that were difficult to transcribe.

The results recorded for each uncertain character region in the line of text included the five most probable characters determined by the CPR system. This was then sent to Cohen, allowing her to verify the original transcription and to provide insight to difficult character regions that would be transcribed based on contextual information. Figure 5.18 shows an example of these resulting character probabilities for a line of text from the colophon that was of particular interest.


Character Recognition GUI v.1.0
File MSFs Composite Filters Other Archimedes About
Load Filter Data File
Browse
Load Training Data File
Training Image Library
Select Character Class Number
Reset Class
Reset All
Output Data File Storage
Directory
Name Files
Correlation Filter Performance Measure(s)
Lock & Load
Run System
Reset Settings
Input Image
Region of Interest - CONFIRMED
Select ROI
Reset ROI
Data Reduction & Classification
Processing - Output
Current Image Library
CHARACTER RECOGNITION INTERFACE for understanding the ARCHIMEDES PALIMPSEST

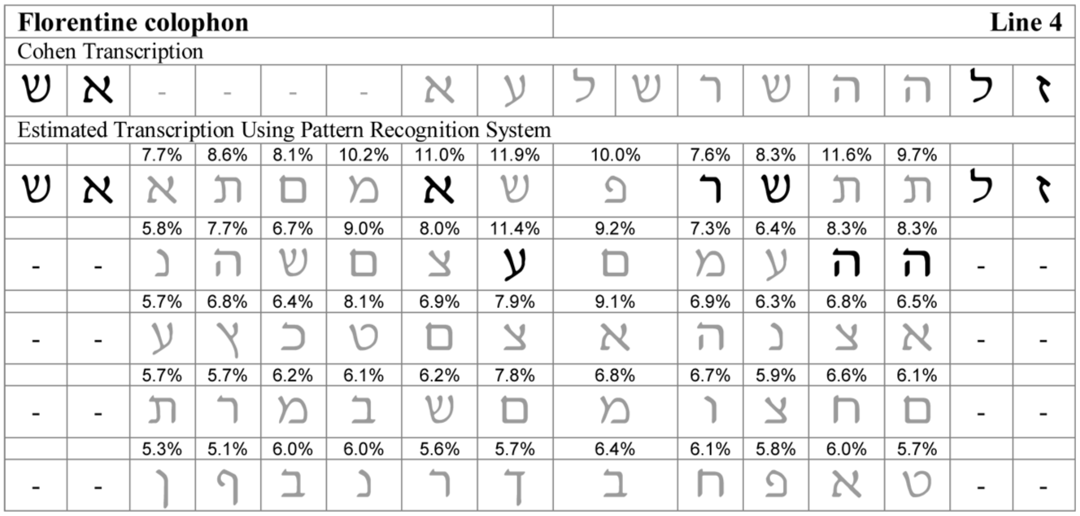
Load Training Data File: hebrew_doc_training.dat
Select Character Class Number: 1
<loaded>
[ ] Filter Data File(s) (.txt) [✓] Output Data File (.txt)
[ ] Training Data File (.txt) [ ] All Correlation Plane Image(s) (.ps)
Directory: C:\DOCUMENTS AND SETTINGS\OWNER\DESK
Name Files: hebrew_doc
[✓] PSR [ ] SNR [ ] PCE [ ] FAHM
Classification...
Input Image: florentine lwuv-vilnius-U_input_ce.bmp
[✓] Principal Components Analysis
[✓] Minimum Distance Classification
13.9301

Output Data File

```
   4.078    4.815    6.640    6.845    3.982    4.905    3.
   3.666    4.170    6.874    5.221    4.246    5.598    4.
   4.208    4.543    5.578    5.432    4.798    5.733    3.
   4.208    4.543    6.594    5.432    4.798    5.733    3.
mach
   3.372    2.589    3.484    3.455    2.468    2.683    2.
```

Figure 5.17: The character recognition interface used to compute probable character matches between the selected ROI and an input training library of hebrew character images.

Unfortunately, the results were not completely successful, as some characters matched and others did not. Selecting a full character in the ROI window (by manual segmentation) became increasingly difficult in areas of the colophon that were severely degraded. Most importantly however, is the fact that some of the results from this recognition process were used to confirm the area of text which contained the date that the Florentine manuscript was originally written [6].



Florentine colophon															**Line 4**
Cohen Transcription															
ש	א	-	-	-	-	א	ע	ל	ש	ר	ש	ה	ה	ל	ז
Estimated Transcription Using Pattern Recognition System															
		7.7%	8.6%	8.1%	10.2%	11.0%	11.9%	10.0%		7.6%	8.3%	11.6%	9.7%		
ש	**א**	א	ת	ם	מ	**א**	ש	פ		**ר**	**ש**	ת	ת	**ל**	**ז**
		5.8%	7.7%	6.7%	9.0%	8.0%	11.4%	9.2%		7.3%	6.4%	8.3%	8.3%		
-	-	נ	ה	ש	ם	צ	**ע**	ם		מ	ע	**ה**	**ה**	-	-
		5.7%	6.8%	6.4%	8.1%	6.9%	7.9%	9.1%		6.9%	6.3%	6.8%	6.5%		
-	-	ע	ץ	כ	ט	ם	צ	א		ה	נ	צ	א	-	-
		5.7%	5.7%	6.2%	6.1%	6.2%	7.8%	6.8%		6.7%	5.9%	6.6%	6.1%		
-	-	ת	ר	מ	ב	ש	ם	מ		ו	צ	ח	ם	-	-
		5.3%	5.1%	6.0%	6.0%	5.6%	5.7%	6.4%		6.1%	5.8%	6.0%	5.7%		
-	-	ן	ף	ב	נ	ר	ך	ב		ח	פ	א	ט	-	-

Figure 5.18: The resulting five most probable characters at each location from the text of the fourth line in the Florentine colophon. Bold characters in the estimated transcription indicate probable characters in agreement with Cohen's original transcription. The percentage values are the probabilities of a match for the corresponding character over all other character classes.

5.2.3 Automated Detection of Fiducial Skin Markers in MRI Data

The flexibility of the CPR system has also been demonstrated through preliminary work on automating the detection of fiducial skin markers for registration of 3-dimensional MRI breast imaging. For many medical imaging procedures, it is common practice to acquire data from multiple imaging modalities (e.g., MRI, PET). It is often desirable to produce a "fused" output that simultaneously exhibits characteristics of the data from each individual modality to reduce the difficulty of the decision-making process for radiologists. Preprocessing for most data fusion algorithms typically provides the necessary registration of the input data (from each modality). Fiducial markers may be used to show common locations between the imaging modalities when the methods of image capture produce outputs with very different spatial structure, as is the case with MRI and PET imagery. The process of automating the detection of these markers has seen limited research in the medical field, and often requires manual selection throughout the 3-dimensional image stack by a human observer. To further demonstrate the flexibility of the CPR system, the design was improved to accommodate full MRI image stacks such that the centroid locations of markers could be determined using the correlation techniques in the filter library. This task takes full advantage of the inherent shift invariance of spatial correlation, which permits detection over the entire scene.

The fiducial skin markers are ellipsoidal (almost spherical) in shape and appear with relatively high contrast in the MRI data. Progressing through the image stack, the shape of an individual marker increases in size until it reaches a maximum, and then its size decreases until the full 3-dimensional marker has been imaged. Thus, the volume of a marker is represented by image slices containing the approximate area, which also increases to a maximum and then decreases.

The objective is to detect each marker (and locate its centroid location) in a noisy background containing additional objects with a large range of intensity values. Thus, simple thresholds or adaptive thresholds fail this task. Any correlation methods employed must exhibit some "normalizing" characteristic, similar to that provided by the correlation coefficient (Section 3.2.1.3). The correlation coefficient accommodates changes in the input image such that regions of high intensity that do not share similar spatial structure with the reference pattern are assigned low values in the output correlation plane, effectively reducing the false alarm rate. We are also concerned with

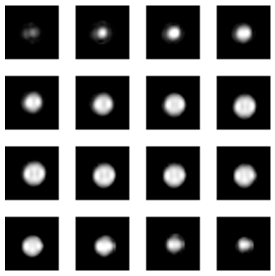


Figure 5.19: Training exemplars extracted from a single fiducial skin marker.

the abilitiy of the filter to accommodate within-class distortion, as the size and shape of the fiducial marker will vary through the image stack. Thus, we seek a composite correlation design that has been "corrected" for normalization.

For this work, a mean-subtracted MACH filter was constructed and applied to input stacks that were mean-subtracted locally over 32×32 pixel regions, which is the size of the training. Figure 5.19 shows the training exemplars used to produce the desired transfer function. The output image stack of correlation planes was then analyzed to determine the location of marker centroids. This was accomplished by first selecting and applying a global threshold (i.e., over the full image stack). Once the binary stack is produced, 4-neighbor connected components is employed to define each region with nonzero value. The mean x, y, and z coordinate of each region is then calculated, which denotes the possible centroid locations of fiducial markers. To reduce the number of false alarms, a region-growing algorithm is implemented at each potential centroid. If the volume of the grown region lies between predefined thresholds, the algorithm assumes that the centroid location corresponds to one of the fiducial skin markers. Figure 5.20 shows a slice from a 3-dimensional MRI image stack at three stages of this algorithm.

Preliminary results were obtained using the training set in Figure 5.19 and MRI image stacks from five individuals, which each contained between nine and eighteen fiducial markers. Adjusting the level of the global threshold applied to the correlation output either increases or decreases the number of regions computed via connected components. Obviously, if the threshold is set too high, peaks in the correlation output that correspond to locations of markers may be lost. Conversely, if the threshold is set too low, many regions in the correlation output must be analyzed as potential locations for markers, effectively increasing the discrimination task and false alarm rate.

Figure 5.21 shows the ROC curve produced for this data set after applying the algorithm discussed in this section. The plot in Figure 5.22 is included to show the global thresholds that were used as the operating points of the ROC curve. An operating threshold of 0.5 corresponds to a threshold of 50% of the global maximum applied to the full correlated output image stack. It is apparent that a relatively high probability of detection is obtained for a wide range of thresholds for an acceptable false alarm rate. A noticeable problem is that the algorithm does not attain a 100% probability of detection even after drastically lowering the global threshold. This obstacle is due to the poor contrast characteristic of slices from the beginning and end of the image stacks. Methods of preprocessing the MRI data prior to implementation of the correlation algorithm are being considered for future work.

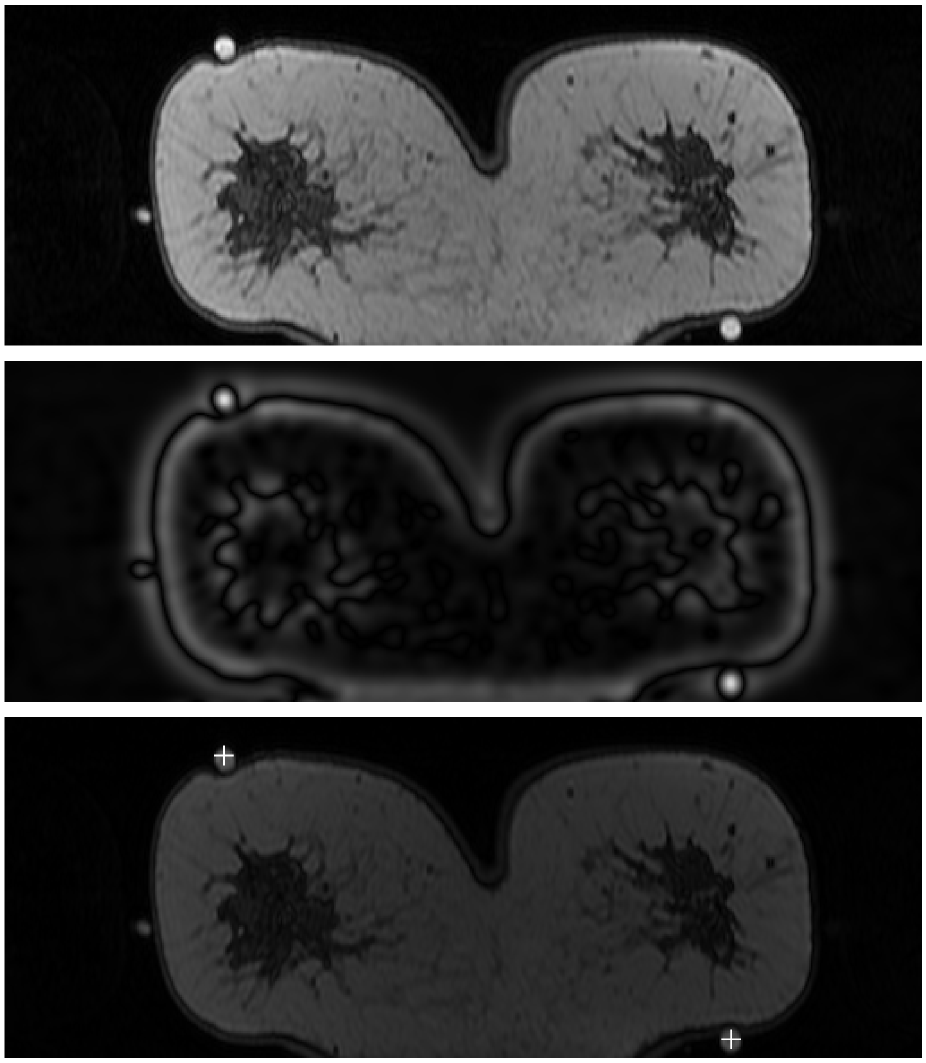


Figure 5.20: Slice 110 from the 3-dimensional MRI image stack (Top) before preprocessing to subtract the local mean, (Middle) after correlation with a mean-subtracted MACH filter, and (Bottom) post-processed to show the centroid locations of registration markers. Note that the leftmost marker was detected, but its centroid lies further through the image stack.

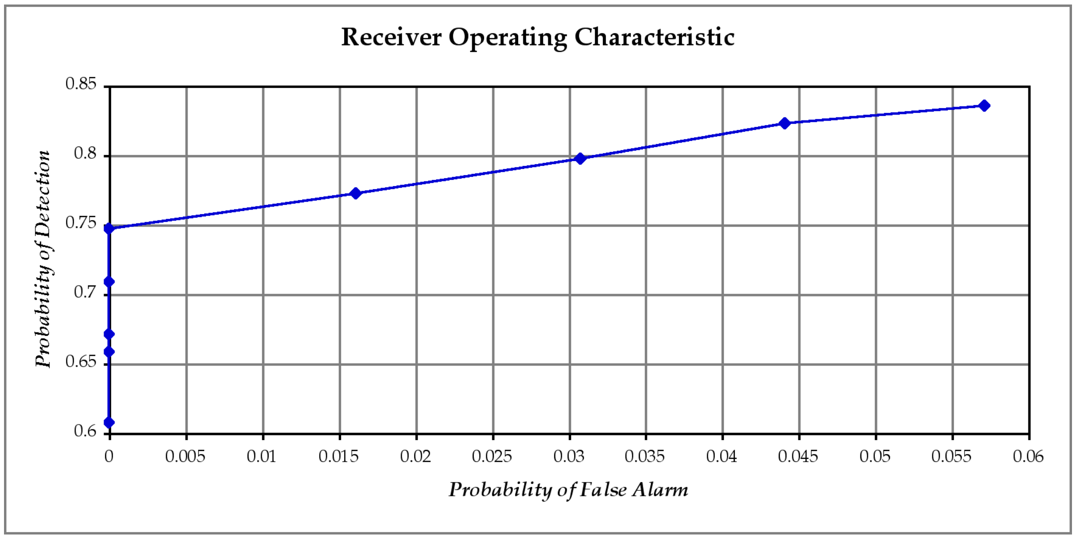


Figure 5.21: The ROC curve produced after applying CPR to classify fiducial skin markers from 3-dimensional MRI image stacks of five individuals.

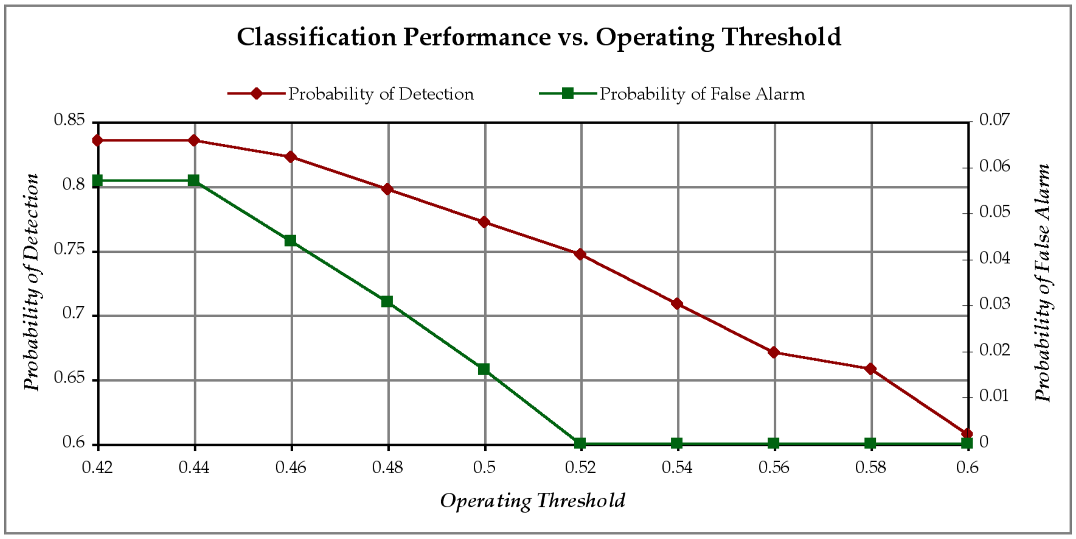


Figure 5.22: The global thresholds applied to the correlation output to control the operating point on the ROC curve.

5.3 LSCF Demonstration

This section contains two examples of LSCF application, each illustrating an important concept in the use of linear subspaces for correlation filtering. Using synthetically-generated characters, the first example shows how spatial translation and in-plane rotation affect the correlation plane produced using the Corefaces and LSCF algorithms. Figure 5.23 shows the training exemplars used to derive the $c = 6$ PC subspaces, which reconstruct input-phase spectra, and the single LDA subspace, which reconstructs distinct versions of magnitude spectra. Each of the six classes contains five exemplars that differ only in rotation angle. The corresponding mean-subtracted training images are shown in Figure 5.24. An interesting result is provided in Figure 5.25, which is not part of the LSCF algorithm, but is included for discussion. The exemplars have been projected onto an LDA subspace derived from the full set of training images, and subsequently reconstructed in the spatial domain. Note that each character is partially reconstructed (shown in white), but appears to contain a negative combination of the other classes (shown in black).

Figures 5.26-5.28 show similar sets of images for the test set containing spatially-translated characters that have been rotated by different amounts than the training set. Perhaps the most important result of this example is shown in Figure 5.28, which shows how spatial translation affects LDA-subspace projection; partial reconstruction of the characters is no longer visible. A translation of even one pixel may produce drastic differences in the direction of the corresponding lexicographically-ordered vectors.

We now proceed as discussed in Section 4.3: the phase and magnitude spectra are projected onto the PC subspace and LDA subspace, respectively. Figure 5.29 shows the magnitude spectra of each test image, which exhibit identical rotations. Using case 2 LDA to minimize within-class variability while maximizing class separability, five discriminant functions are produced and displayed as images of magnitude spectra in Figure 5.30. Figure 5.31 shows the next logical step in this progression: the magnitude spectra of the test images after projection onto the LDA subspace and subsequent reconstruction in the frequency domain. Notice the visual difference between the reconstructed-magnitude spectra of class '*a*' and '*o*'. The use of this particular linear subspace has removed magnitude components from class '*a*' and '*o*' that serve little purpose in discrimination tasks. Also note the reduction of within-class distortion in

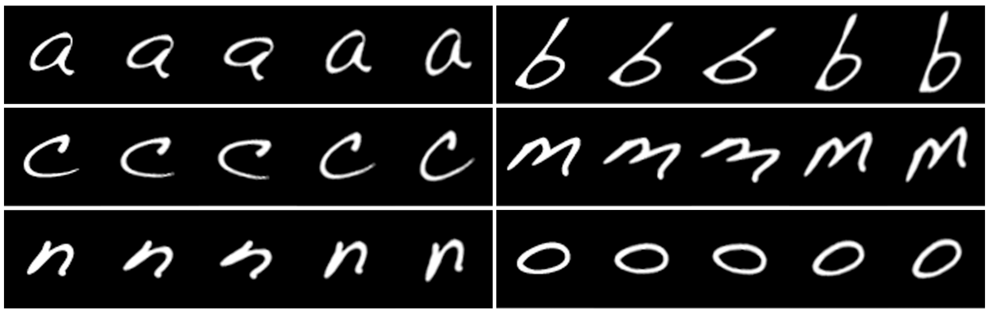


Figure 5.23: Synthetic characters example – training images for the $c = 6$ classes. For each class of characters, each of the 5 exemplars is rotated by a multiple of 10 degrees ($[0, -1, -2, +1, +2]$ as shown).

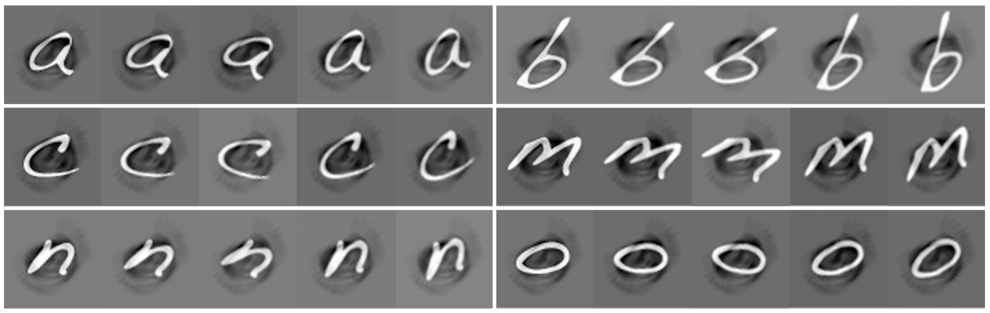


Figure 5.24: Synthetic characters example – corresponding training images for each class after subtracting the mean training image of the full set of 30 exemplars

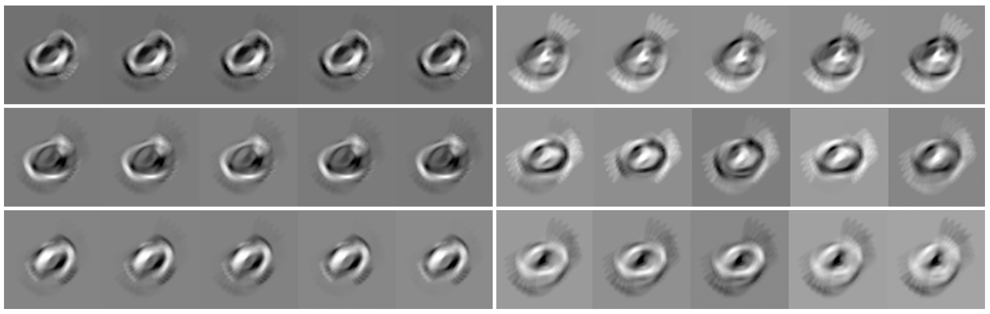


Figure 5.25: Synthetic characters example – corresponding training images after projection onto a spatially-derived LDA subspace and subsequent reconstruction to an image representation in the spatial domain. Note that this figure is for illustration purposes only; it is not part of the LSCF algorithm.

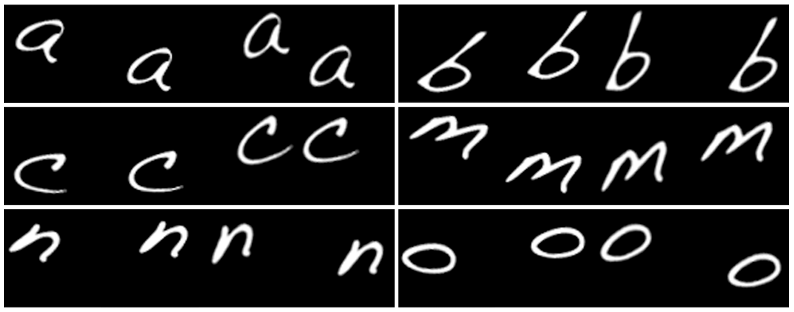


Figure 5.26: Synthetic characters example – test images for the $c = 6$ classes. For all classes of characters, each of the 4 images is rotated by a multiple of 5 degrees ($[-3, -1, +3, +1]$ as shown) and spatially translated.

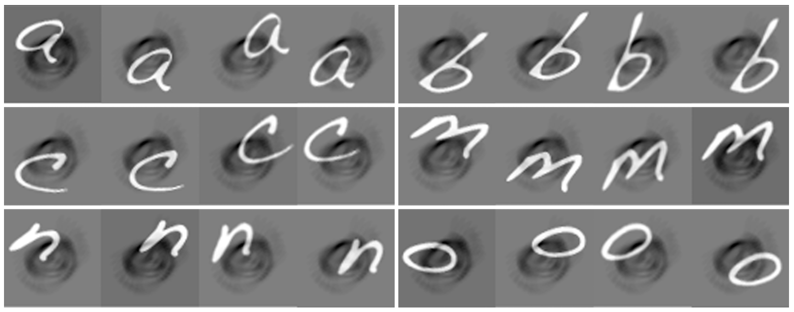


Figure 5.27: Synthetic characters example – corresponding test images for each class after subtracting the mean training image of the full set of 30 exemplars.

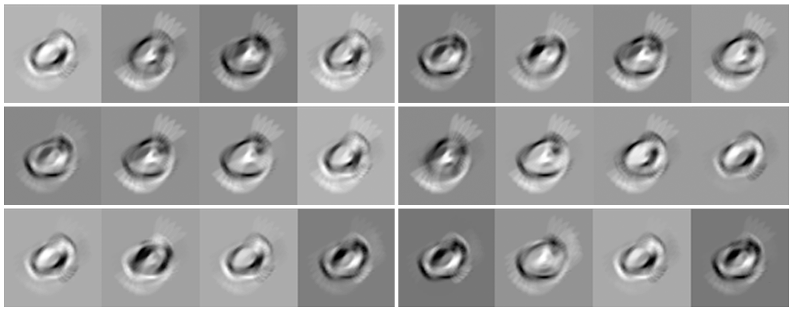


Figure 5.28: Synthetic characters example – corresponding test images after projection onto a spatially-derived LDA subspace and subsequent reconstruction to an image representation in the spatial domain. Note that this figure is for illustration purposes only; it is not part of the LSCF algorithm.

class ‘*a*’ and ‘*o*’. Each magnitude spectrum now appears almost identical to the other magnitude spectra belonging to the same class. It should be emphasized that these visual observations may not always be apparent despite adequate LDA performance; if only two classes exist, one discriminant vector is used to separate the classes. Projecting the two classes onto this vector yields reconstructed spectra that differ by a (very useful) scale factor. It is this scale factor that is used for classification.

The phase spectra of the test images, which contain spatial-translation information, are shown as spatial-domain images in Figure 5.32. Unlike LDA, which produces one vector subspace to discriminate all classes, PCA is employed to construct a vector subspace that “best” represents each of the classes. The five principal directions describing the variance in the phase spectra of class ‘*a*’ are shown in Figure 5.33. Figure 5.34 shows the phase of the test characters after projection onto the subspace in Figure 5.33 and reconstruction in the frequency domain. Because each phase is a linear combination of the PCA vectors, the phase images in Figure 5.34 share similar shape in the form of the character ’*a*’. When phase canceling is performed via correlation, the phase of test characters from class ‘*a*’ will cancel well, which results in peak formation. Also, while the test images contain translated versions of the reference pattern, the images produced from the reconstructed phase spectra show centered characters. The assumption is that the phase of the reference pattern will be approximately reconstructed, and the linear-phase term specifying translation will be removed by the subspace projection. Thus, phase canceling between the original and reconstructed-test images from the true class should leave only the linear-phase term to specify the location of the reference pattern.

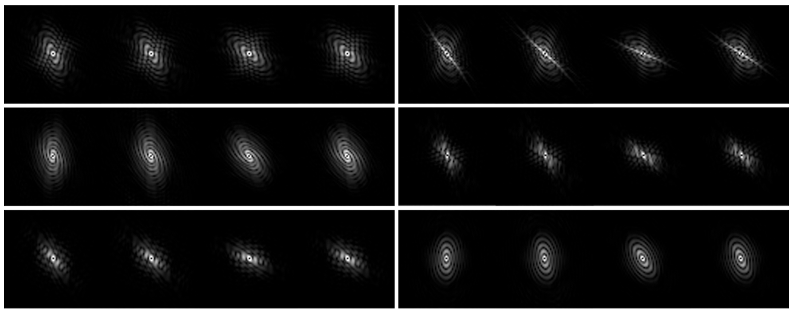


Figure 5.29: Synthetic characters example – corresponding magnitude spectra of the test images. The angle of rotation is identical between the spatial domain image and frequency domain magnitude representations. Translation information is absent in the magnitude spectra.

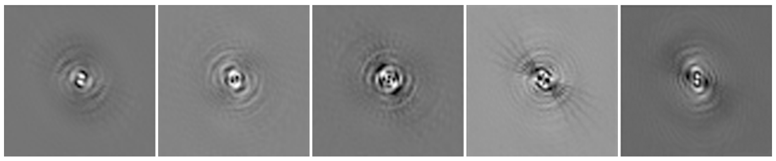


Figure 5.30: Synthetic characters example – LDA subspace derived from the magnitude spectra of the full set of training exemplars. The 5 lexicographically-ordered vectors (shown here as images) are the discriminant functions, specified by LDA, that maximize between-class separation while minimizing within-class variability.

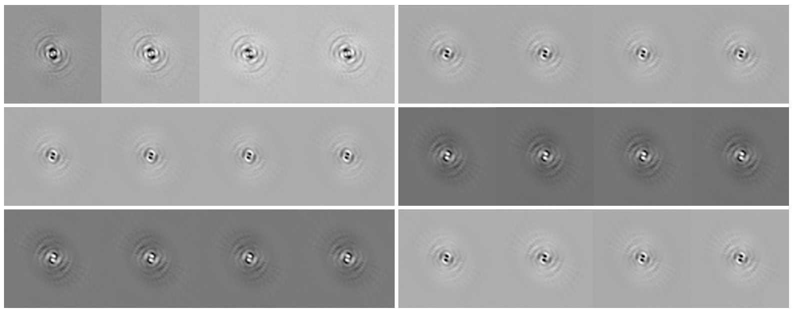


Figure 5.31: Synthetic characters example – corresponding magnitude spectra of the test images after projection onto the LDA subspace and subsequent reconstruction to a magnitude representation in the Frequency domian.

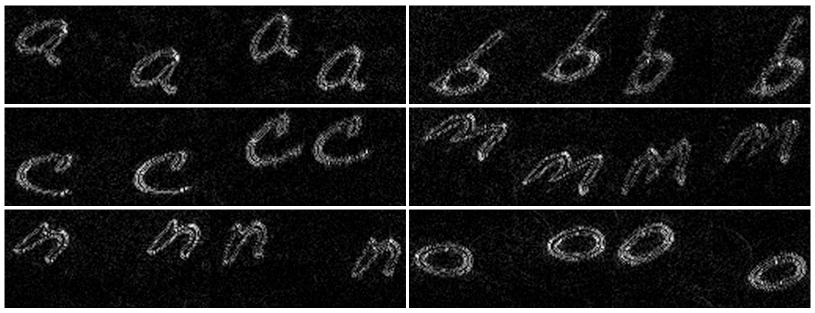


Figure 5.32: Synthetic characters example – corresponding phase information of the test images (shown as spatial domain images).

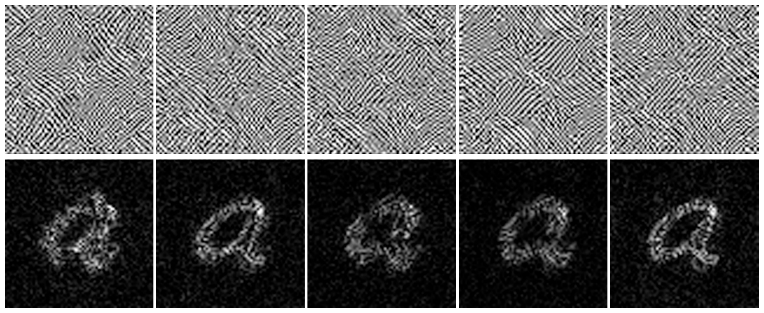


Figure 5.33: Synthetic characters example – PC subspace derived from the phase spectra of the first class of training exemplars. The 5 lexicographically-ordered vectors, shown here as phase spectra (TOP) and spatial domain images, are the directions that maximize the variability and provide optimal reconstruction of class 1.

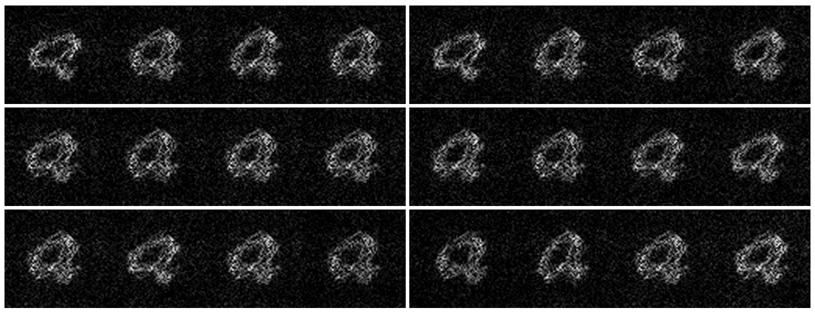


Figure 5.34: Synthetic characters example – corresponding phase information of the test images (shown as spatial domain images) after projection onto the PC subspace for the first class and subsequent reconstruction to a phase representation in the Frequency domain.

Figure 5.35 shows images of the correlation planes produced for each test image when the phase of each class was reconstructed from the true-class PC subspace. This is a direct implementation of the Corefaces algorithm [43]. While three of the six classes show peak formation at the location of the character, distortion of the test images in the other three classes prevented adequate phase canceling. In addition, false alarms occur even when all test targets from a particular class produce peaks. Figure 5.36 shows that when the phase of the test images from class '*a*' is reconstructed from each PC subspace, some of the peaks are misclassified as characters '*b*' and '*o*'.

The LSCF algorithm was also applied to the data set of synthetic characters. While the "inverse filtering" method detailed in Section 4.3.2 may provide sufficient correlation peaks for classification, the LDA-magnitude term was used differently in this chapter. The magnitude from each test image was projected onto the LDA subspace (Figure 5.30) and reconstructed in the frequency domain. This process was also performed for the training, and an mean magnitude reconstruction was obtained for each class of training. Rather than use the mean magnitude reconstructions as inverse filters, they were projected onto the reconstructed magnitude of the test image, yielding some value p. The scalar quantity $q = 1 - |1 - p|$ provides a measure of similarity for the reconstructed-magnitude spectra. For magnitude spectra projected onto the mean reconstructed-magnitude spectra of the true class, the values p and q will be close to unity. Conversely, projection onto the false class yields a low value for p, causing q to decrease. This value q provides added class discrimination to reduce false alarms. For this example, the peak-to-sidelobe ratio of the LSCF algorithm is just the product of q and the PSR of the Corefaces algorithm.

Figure 5.37 shows the confusion matrices produced by the Corefaces and LSCF algorithms for the test images of synthetic characters. The overall recognition rate has improved from 42% to 88% (14 false alarms to 3 false alarms). PSR values used to classify the test characters are provided in Figure 5.38 and Figure 5.39 for Corefaces and LSCF, respectively. Both plots show PSR values using the true-class and false-class subspace(s); false-class PSRs are characterized by the mean and standard error. The strong overlap between the true-class and false-class PSRs produced using Corefaces results in a low overall recognition rate. A noticeable increase in separation between true-class and false-class PSRs is achieved using the LSCF algorithm, which indicates better class discrimination and more accurate classification.

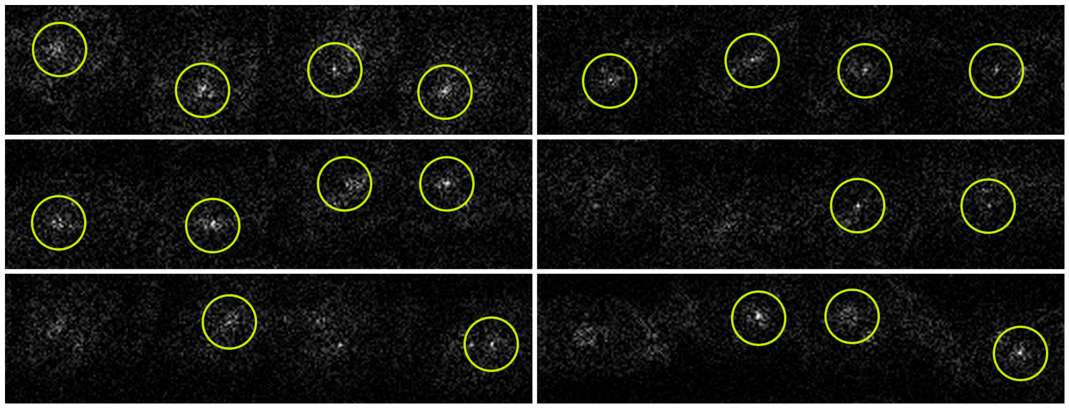


Figure 5.35: Synthetic characters example – resulting correlation planes (shown as images) for each of the 6 classes of test images when the phase was reconstructed with the true-class PC subspace. Peaks circled in yellow correspond to detection.

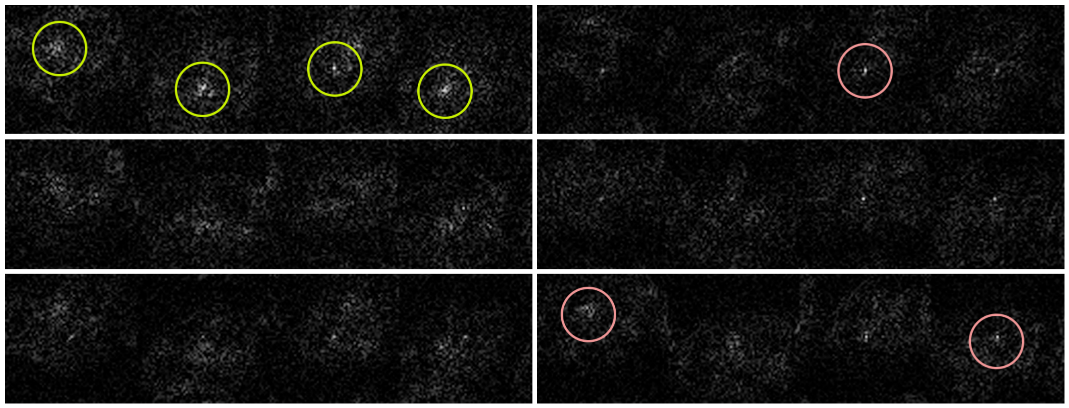


Figure 5.36: Synthetic characters example – resulting correlation planes (shown as images) for test images from the first class when the phase was reconstructed with each of the 6 PC subspaces. Peaks circled in yellow correspond to the true-class peak, and peaks circled in pink correspond to false alarms.

	Synthetic Characters				*Corefaces*		
	1	2	3	4	5	6	*Images*
1	25%	25%	0%	0%	0%	50%	4
2	0%	50%	0%	50%	0%	0%	4
3	0%	25%	50%	0%	0%	25%	4
4	25%	25%	0%	50%	0%	0%	4
5	0%	50%	0%	0%	25%	25%	4
6	0%	50%	0%	0%	0%	50%	4
Overall							42%

	Synthetic Characters				*LSCF*		
	1	2	3	4	5	6	*Images*
1	75%	0%	0%	0%	0%	25%	4
2	0%	100%	0%	0%	0%	0%	4
3	0%	25%	75%	0%	0%	0%	4
4	0%	0%	0%	100%	0%	0%	4
5	0%	0%	0%	25%	75%	0%	4
6	0%	0%	0%	0%	0%	100%	4
Overall							88%

Figure 5.37: Synthetic characters example – confusion matrices comparing the Corefaces and LSCF algorithms for the test set of characters. The calculated recognition rate over the set of $c = 6$ classes was 42% and 88% for the respective correlation techniques.

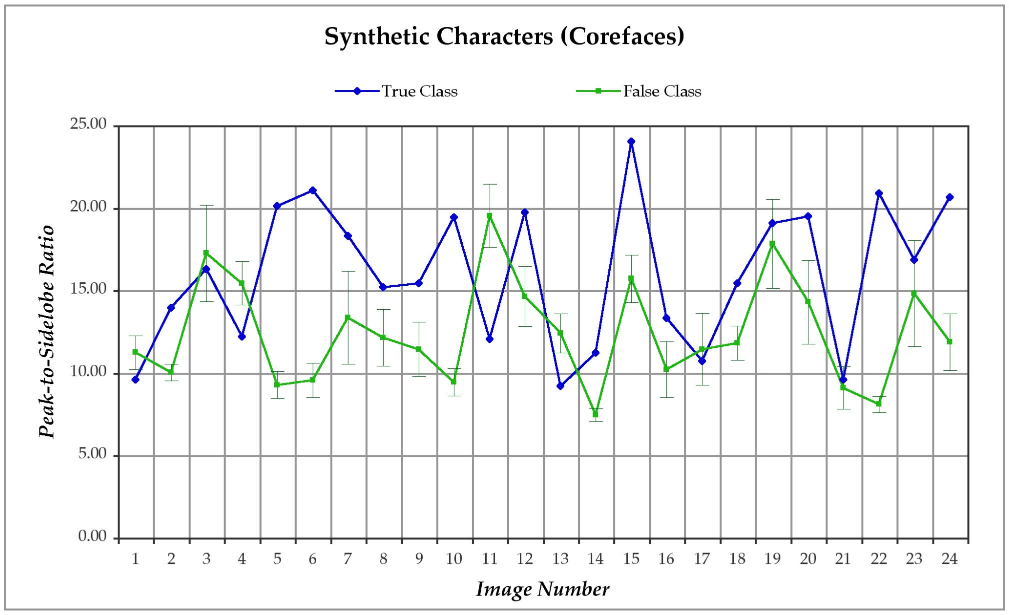


Figure 5.38: Synthetic characters example – comparison of the true-class PSR and mean false-class PSR values for all 24 test images using the Corefaces algorithm.

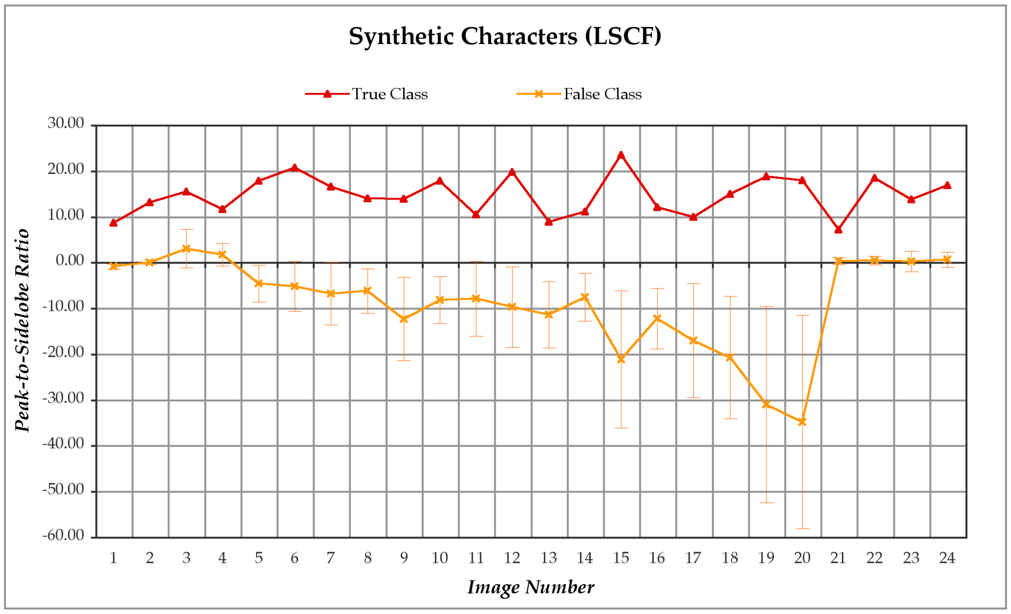


Figure 5.39: Synthetic characters example – comparison of the true-class PSR and mean false-class PSR values for all 24 test images using the LSCF algorithm.

A second example of LSCF application is included to demonstrate the effectiveness of the algorithm on images of faces. The images are taken from the Yale-B face database [15], and contain variation in pose under constant illumination. Figure 5.40 shows faces of four individuals ($c = 4$ classes) at four different poses used as training. Images of the same four individuals at different poses are included in this example to test the ability of each correlation algorithm to tolerate out-of-plane rotation (Figure 5.42). The mean-subtracted images of the training and test sets are shown in Figure 5.41 and Figure 5.43, respectively.

Figure 5.44 shows images of correlation planes generated in similar fashion to those presented in the previous example. Results from the application of the Corefaces algorithm show that none of the classes of faces produced a peak for each test image. Observing the correlation peaks from the top-left individual, we find that false alarms occur for test images reconstructed from two false-class PC subspaces (Figure 5.45). Moreover, the face that did not produce a peak for the true-class reconstruction was classified with a high PSR value for the wrong individual.

The classification results for the Corefaces and LSCF algorithm are illustrated by the confusion matrices in Figure 5.46. Again, the overall recognition rate improves using the added class discrimination provided by the LDA-magnitude component of the LSCF. Most noticeable is the improved classification accuracy for test face images of the bottom-left individual. Figure 5.47 and Figure 5.48 show the PSR values produced using the Corefaces and LSCF algorithms, respectively. The increased separation between true-class and mean false-class PSRs using the LSCF algorithm is responsible for raising overall recognition rate.

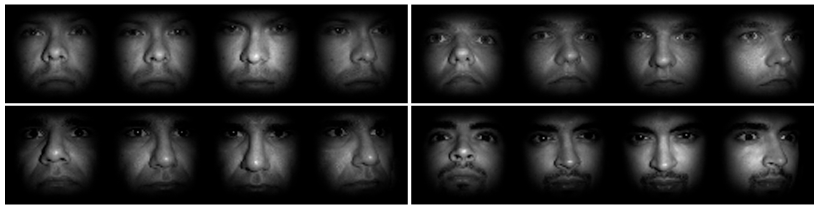


Figure 5.40: Yale-B face example – training images for the $c = 4$ classes. For each class of faces, the pose varies between each of the 4 faces.

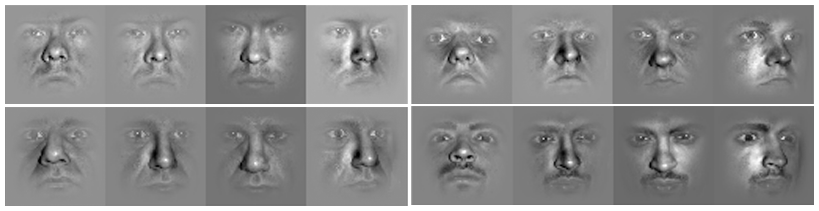


Figure 5.41: Yale-B face example – corresponding training images for each class after subtracting the mean training image of the full set of 16 exemplars.

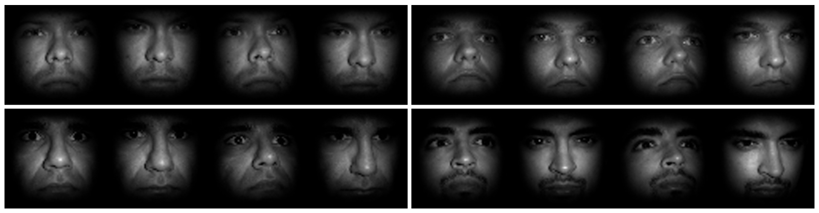


Figure 5.42: Yale-B face example – test images for the $c = 4$ classes. For each class of faces, the pose varies between each of the 4 faces, and is different than the training.

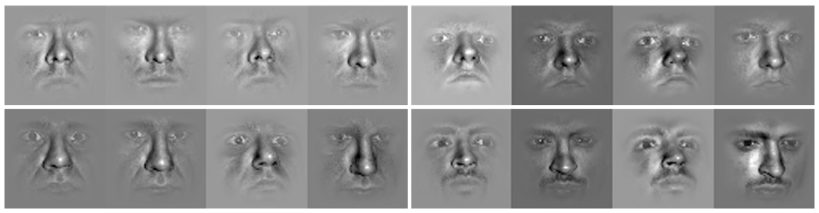


Figure 5.43: Yale-B face example – corresponding test images for each class after subtracting the mean training image of the full set of 16 exemplars.

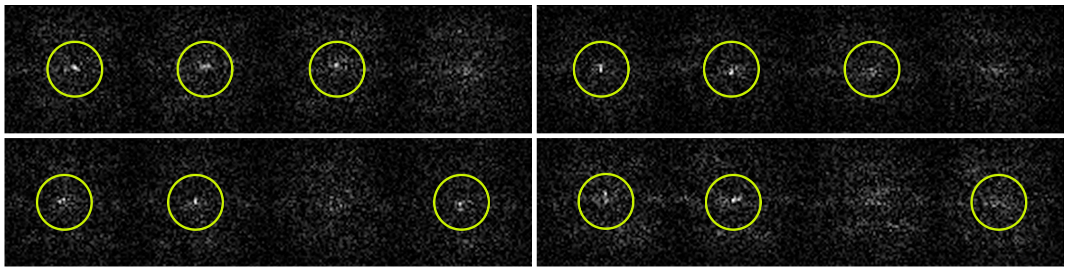


Figure 5.44: Yale-B face example – resulting correlation planes (shown as images) for each of the 4 classes of test images when the phase was reconstructed with the true-class PC subspace. Peaks circled in yellow correspond to detection.

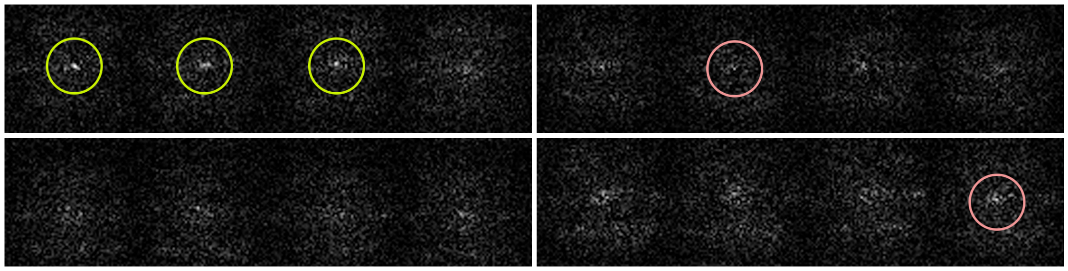


Figure 5.45: Yale-B face example – resulting correlation planes (shown as images) for test images from the first class when the phase was reconstructed with each of the 6 PC subspaces. Peaks circled in yellow correspond to the true-class peak, and peaks circled in pink correspond to false alarms.

	Yale-B		*Corefaces*		
	1	2	3	4	*Images*
1	50%	25%	0%	25%	4
2	25%	75%	0%	0%	4
3	0%	0%	75%	25%	4
4	0%	75%	25%	0%	4
Overall					50%

	Yale-B		*LSCF*		
	1	2	3	4	*Images*
1	75%	25%	0%	0%	4
2	25%	75%	0%	0%	4
3	0%	0%	100%	0%	4
4	0%	0%	25%	75%	4
Overall					81%

Figure 5.46: Yale-B face example – confusion matrices comparing the Corefaces and LSCF algorithms for the test set of faces. The calculated recognition rate over the set of $c = 4$ classes was 50% and 81% for the respective correlation techniques.

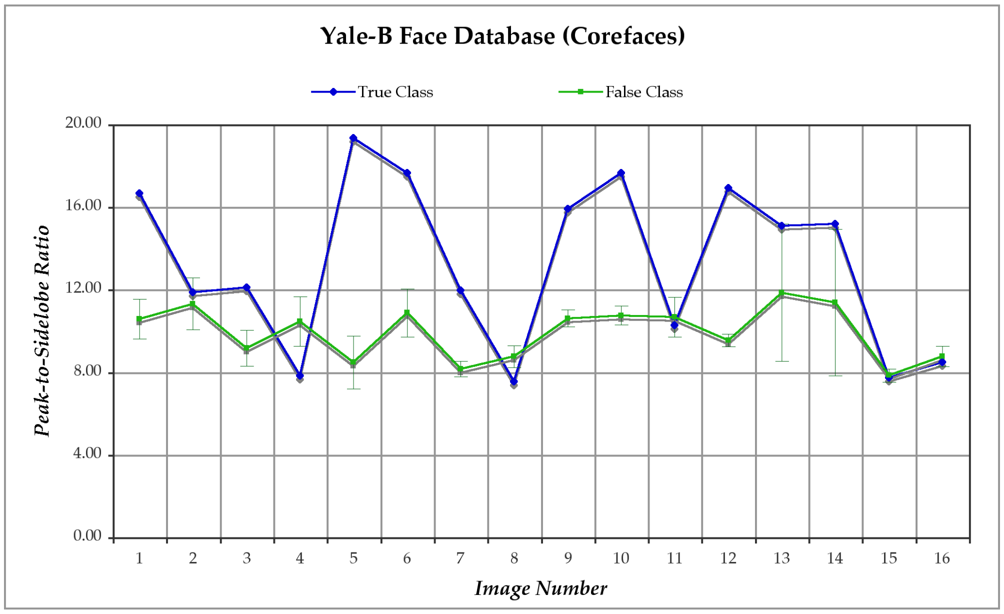


Figure 5.47: Yale-B face example – comparison of the true-class PSR and mean false-class PSR values for all 16 test images using the Corefaces algorithm.

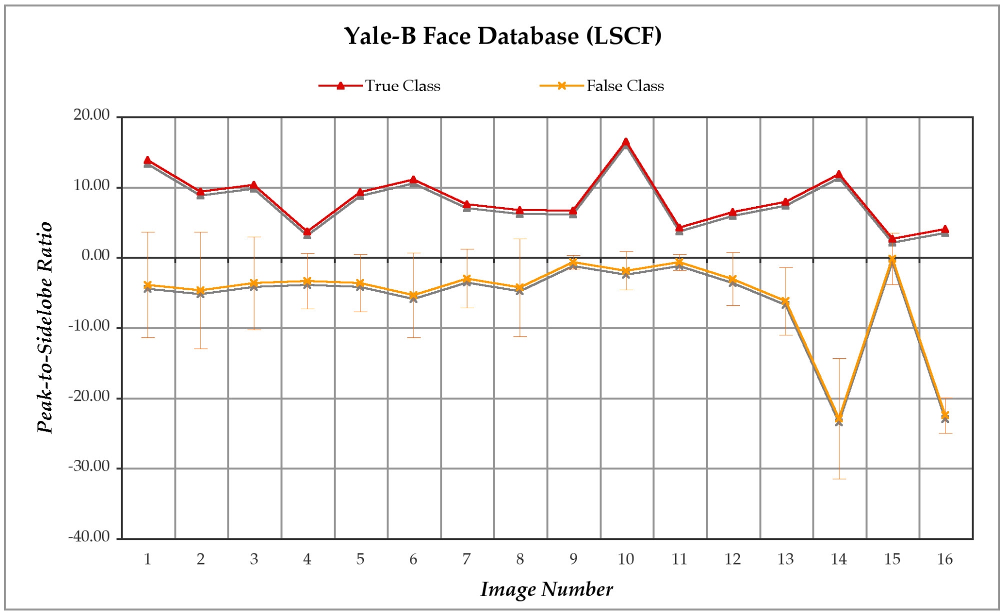


Figure 5.48: Yale B face example – comparison of the true-class PSR and mean false-class PSR values for all 16 test images using the LSCF algorithm.

–Not everything that can be counted counts, and not everything that counts can be counted.

Albert Einstein (1879-1955)

6 Conclusions

We conclude this body of work by briefly summarizing the completion of the stated objectives, reviewing the relevant contributions, and discussing possibilities for future work related to this topic.

The correlation pattern recognition system created during the course of this work has been analyzed and quantitatively evaluated using manually extracted test sets of character from the Archimedes Palimpsest imagery. Though relatively high recognition rates were achieved for both clean and obscured characters, there are regions of the manuscript in which the damage is too severe to make a decision based solely on the spatial properties of the images. The existing partial transcription served as the foundation for constructing conditional probability tables for the integration of a probabilistic network into the recognition system. This advancement permitted scholars to incorporate their knowledge of the language and context into the decision-making process by applying evidence to the model. Thus, the output probabilities generated by the system are produced using three sources of information: the palimpsest imagery, the partial transcription, and the contextual knowledge of the user. The overall

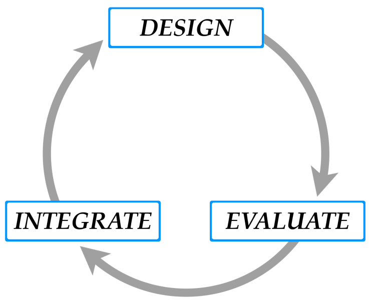


Figure 6.1: Cyclic workflow for future work using the current CPR system.

character recognition system has been qualitatively evaluated by scholars using difficult regions of text, and their feedback has been both positive and encouraging.

The flexibility of spatial correlation-based character recognition has enabled use of the CPR system on additional applications. Using a training set consisting of a limited number of exemplars, the CPR system was used to confirm the publication date of a fifteenth-century Hebrew colophon. The system has also demonstrated success in the task of automating the detection of fiducial markers in 3-dimensional MRI breast imaging. The results from these applications are included to provide example usage of the CPR system in future projects.

To complete one revolution in the cyclic workflow proposed in this work (Figure 6.1), a new method of correlation filtering was developed to exploit the benefits of linear discriminant analysis while maintaining shift invariance. The LSCF method proposed in this work was applied to synthetic characters that varied in rotation angle and face images from the Yale-B face database [15] that varied in pose. Results from these experiments verify that this filter design benefits from from high within-class distortion tolerance and discrimination ability.

Possibilities for future work exists in each major area discussed in this text: application of the CPR system, the probabilistic network, and the LSCF algorithm. The flexibility of the CPR system is apparent, and the GUI is available for use on additional applications. Its design permits simple updates to the filter library, encouraging development of new filters to be integrated into the system. The current probabilistic network can be greatly improved by accepting partial words as inputs, which would permit inference over character strings in the Archimedes word dictionary. Use of a multiply-connected network, which could require an approximate inferencing algorithm, will also increase the efficiency of the design. The LSCF shows promises for applications with multiple classes sharing similar spatial characteristics, but changes in illumination between the input and training may corrupt the correlation plane. Use of linear subspaces in correlation filter design opens the field of correlation pattern recognition to a new class of filtering techniques.

This is just one story from the Archimedes project in which a series of technological advancements, built upon the mathematical foundations of Archimedes' existing work, was used to assist in transcribing additional information from the famous 10^{th}-century manuscript. The irony here is twofold; these mathematical foundations not only made possible the recovery of historically significant findings from the mind of Archimedes himself, but also provided a framework for the development of completely new algorithms, such as the LSCF detailed in this work. Thus, the work encapsulated by this dissertation has demonstrated how algorithms derived using concepts from Geometry, Calculus, and Combinatorics were used to look backwards to the past and move forward to the future.

–To teach is to learn twice.

Joseph Joubert

A Mathematical Foundations

A.1 Generalized Eigenvalue Problems

Consider the linear equation $\mathbf{Ax} = \mathbf{b}$. The matrix operator $\mathbf{A}$ will change the direction of almost any input vector $\mathbf{x}$. *Eigenvectors* $\mathbf{e}$ are those vectors that point in the same direction as $\mathbf{Ae}$. Multiplying an eigenvector by the matrix $\mathbf{A}$ yields an output vector $\mathbf{b}$ that is just a scaled version of the original eigenvector $\lambda\mathbf{e}$. The scale factor λ is the *eigenvalue* associated with the particular eigenvector $\mathbf{e}$. Thus, we arrive at the standard equation for eigenvalue problems:

$$\mathbf{Ae} = \lambda\mathbf{e} \tag{A.1}$$

Eq. A.1 can be written as $\mathbf{AE} = \mathbf{E\Lambda}$ for a set of $\mathbf{e}$'s and λ's; the columns of the matrix $\mathbf{E}$ are the eigenvectors, and the elements along the main diagonal of the diagonal matrix $\mathbf{\Lambda}$ are the corresponding eigenvalues.

A *generalized eigenvalue problem* exists when two matrices, $\mathbf{A}$ and $\mathbf{B}$, are present in the eigenproblem:

$$\mathbf{Ae} = \lambda \mathbf{Be} \tag{A.2}$$

If **B** is nonsingular, the generalized eigenvalue problem can be expressed as a standard eigenvalue problem $(\mathbf{B}^{-1}\mathbf{A})\mathbf{e} = \lambda \mathbf{e}$. In the event that **B** is the identity matrix **I**, Eq. A.2 returns to the standard form in Eq. A.1.

A.1.1 Simultaneous Diagonalization

Simultaneous diagonalization is the process of diagonalizing two symmetric matrices, **A** and **B**, of a generalized eigenvalue problem by finding a nonsingular transformation matrix **T**. This matrix is determined by first diagonalizing the matrix **B** such that:

$$\mathbf{V}^T\mathbf{BV} = \mathbf{\Lambda}_B \tag{A.3}$$

where **V** is a matrix containing the eigenvectors of **B**, and $\mathbf{\Lambda}_B$ is a diagonal matrix of corresponding eigenvalues. It follows that the matrix $\mathbf{Y} = \mathbf{V}\mathbf{\Lambda}_B^{-\frac{1}{2}}$ diagonalizes **B** to unity, i.e., the identity matrix **I**:

$$\mathbf{Y}^T\mathbf{BY} = \mathbf{I} \tag{A.4}$$

These scaled eigenvectors of **B** (the columns of **Y**), provide a change of basis such that in the new coordinate system, the concentration ellipse specified by the matrix **B** is a sphere. This fact allows the basis to be rotated to diagonalize the matrix **A** while retaining the diagonalized matrix **B**. Thus, the goal now is to find the matrix **U** which diagonalizes $\mathbf{Y}^T\mathbf{AY}$, that is:

$$\mathbf{U}^T\mathbf{Y}^T\mathbf{AYU} = \mathbf{\Lambda}_\mathbf{A} \tag{A.5}$$

where **U** is the matrix of eigenvectors of $\mathbf{Y}^T\mathbf{AY}$ and $\mathbf{\Lambda}_\mathbf{A}$ is a diagonal matrix of corresponding eigenvalues. In other words, we wish to diagonalize **A** in the coordinate system specified by the eigenvectors of **B**. This concept is illustrated in Figure A.1.

The overall transformation matrix then, which simultaneously diagonalizes **A** and **B** is:

$$\mathbf{T} = \mathbf{YU} \tag{A.6}$$

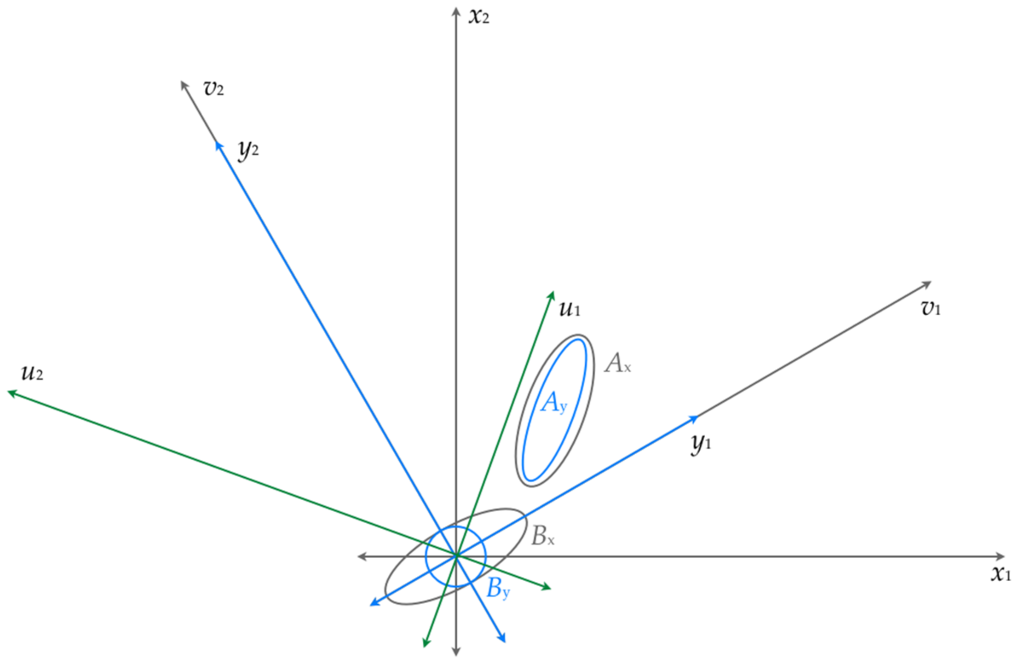


Figure A.1: Simultaneous diagonalization in 2-dimensions – A_x and B_x are the concentration ellipses for the symmetric matrices **A** and **B** in the original basis, which is specified by the variables x_1 and x_2. Variables v_1 and v_2, which point in the directions of the eigenvectors of **B**, specify a change of basis that diagonalizes **B**. Normalizing each direction by its corresponding eigenvalue yields variables y_1 and y_2, and effectively changes the concentration ellipse of **B** to a sphere. The variables u_1 and u_2 diagonalize both matrices **A** and **B** in the scaled coordinate system.

We can verify this by simple substitution:

$$\mathbf{T}^T\mathbf{B}\mathbf{T} = \mathbf{U}^T\mathbf{Y}^T\mathbf{B}\mathbf{Y}\mathbf{U} = \mathbf{U}^T\mathbf{I}\mathbf{U} = \mathbf{I} \tag{A.7a}$$

$$\mathbf{T}^T\mathbf{A}\mathbf{T} = \mathbf{U}^T\mathbf{Y}^T\mathbf{A}\mathbf{Y}\mathbf{U} = \mathbf{U}^T\mathbf{\Lambda}\mathbf{U} = \mathbf{\Lambda}_\mathbf{A} \tag{A.7b}$$

It should be noted that the full transformation matrix $\mathbf{T}$ is a linear combination of the column vectors of $\mathbf{Y}$ with weights specified by the elements of $\mathbf{U}$. $\mathbf{T}$ is rarely an orthogonal matrix. It follows that the full transformation should not be considered a pure rotation [47]. This is a direct result of the scaling applied along the eigenvector directions in the matrix $\mathbf{Y}$. Simultaneous diagonalization can however, be interpreted as a linear transformation that provides a new coordinate system such that the ellipsoids specified by matrices $\mathbf{A}$ and $\mathbf{B}$ are aligned (Figure A.1).

From Eqs. A.7a and A.7b, we find that $\mathbf{T}^T\mathbf{A}\mathbf{T} = \mathbf{T}^T\mathbf{B}\mathbf{T}\mathbf{\Lambda}_\mathbf{A}$. Multiplying both sides on the left by $(\mathbf{T}^T)^{-1}$, we find that $\mathbf{T}$ and $\mathbf{\Lambda}_\mathbf{A}$ contain the eigenvectors and eigenvalues for the generalized eigenvalue problem $\mathbf{A}\mathbf{T} = \mathbf{B}\mathbf{T}\mathbf{\Lambda}_\mathbf{A}$. The process of simultaneously diagonalizing two symmetric matrices is particularly useful when the matrix $\mathbf{B}$ is singular. In this case, only eigenvectors with nonzero eigenvectors of $\mathbf{B}$ are retained (which is referred to as *reduced rank simultaneous diagonalization*) [37]. This method is used to solve the Direct LDA algorithm commonly used in current face recognition applications [54].

A.1.2 Rayleigh's Quotient

The ratio between two quadratic criteria, specified by real symmetric matrices $\mathbf{A}$ and $\mathbf{B}$, can be written:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{A}\mathbf{w}}{\mathbf{w}^T\mathbf{B}\mathbf{w}} \tag{A.8}$$

where $\mathbf{w}$ is typically chosen to be the vector that maximizes the function. This expression is known as the *Rayleigh quotient*, and maximizing the function $J(\mathbf{w})$ with respect to $\mathbf{w}$ yields a generalized eigenvalue problem. The gradient $\bigtriangledown_{\mathbf{w}} J(\mathbf{w})$ of Eq. A.8 is:

$$\begin{aligned}
\bigtriangledown_{\mathbf{w}} J(\mathbf{w}) &= \bigtriangledown_{\mathbf{w}} \left(\frac{\mathbf{w}^T \mathbf{A} \mathbf{w}}{\mathbf{w}^T \mathbf{B} \mathbf{w}} \right) && \text{(A.9a)} \\
&= \frac{2(\mathbf{w}^T \mathbf{B} \mathbf{w}) \mathbf{A} \mathbf{w} - 2(\mathbf{w}^T \mathbf{A} \mathbf{w}) \mathbf{B} \mathbf{w}}{(\mathbf{w}^T \mathbf{B} \mathbf{w})^2} && \text{(A.9b)} \\
&= \frac{2\mathbf{A}\mathbf{w} - 2J(\mathbf{w})\mathbf{B}\mathbf{w}}{(\mathbf{w}^T \mathbf{B} \mathbf{w})} && \text{(A.9c)}
\end{aligned}$$

Setting Eq. A.9c to zero yields:

$$\mathbf{A}\mathbf{w} = J(\mathbf{w})\mathbf{B}\mathbf{w} \tag{A.10}$$

which is in the form of a generalized eigenvalue problem. Thus, the vector $\mathbf{w}$ (or matrix of vectors $\mathbf{W}$) that maximizes the Rayleigh quotient is the eigenvector (or matrix of eigenvectors) of the generalized eigenvalue problem.

A.2 Lagrange Optimization

Lagrange optimization is the process of determining the extrema of a quadratic function subject to a set of linear constraints [48]. Assume that we are given an expression in the quadratic form $\mathbf{x}^T \mathbf{A} \mathbf{x}$, and we seek the real vector $\mathbf{x}$ that maximizes this function while simultaneously satisfying the linear constraint $||\mathbf{x}|| = 1$. The Lagrangian function is in the form:

$$L(\mathbf{x}; \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x}) \tag{A.11}$$

where $g(\mathbf{x}) = 0$. By substitution, we obtain:

$$L(\mathbf{x}; \lambda) = \mathbf{x}^T \mathbf{A} \mathbf{x} - \lambda(\mathbf{x}^T \mathbf{x} - 1) \tag{A.12}$$

where λ is the Lagrange multiplier. Setting the derivative of $L(\mathbf{x}; \lambda)$ to zero converts the constrained optimization problem to an unconstrained problem [9].

$$\frac{dL(\mathbf{x}; \lambda)}{d\mathbf{x}} = \frac{df(\mathbf{x})}{d\mathbf{x}} - \lambda \frac{dg(\mathbf{x})}{d\mathbf{x}} = 0 \tag{A.13}$$

Using the given quadratic form and linear constraint, we arrive at:

$$0 = 2\mathbf{A}\mathbf{x} - 2\lambda\mathbf{x} \tag{A.14}$$

which yields the standard eigenvalue problem:

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \tag{A.15}$$

Thus, the dominant eigenvector of **A** is the solution for **x**.

Bibliography

[1] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[2] D. Casasent. Unified synthetic discriminant function computational formulation. *Applied Optics*, 23(10):1620–1627, 1984.

[3] D. Casasent and W. Chang. Correlation synthetic discriminant functions. *Applied Optics*, 25(14):2343–2350, 1986.

[4] K. R. Castleman. *Digital Image Processing*. Prentice Hall, New Jersey, 1996.

[5] Q. Chen, M. Defrise, and F. Deconinck. Symmetric Phase-Only Matched Filtering of Fourier-Mellin Transforms for Image Registration and Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(12), 1994.

[6] E. Cohen. Gallico's Identity Exposed: Revealing an Erased Colophon from a Renaissance Prayer Book. *Proc. SPIE*, 5676:157–166, 2005.

[7] R. C. Congalton and K. Green. *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices*. Lewis Publishers, New York, 1999.

[8] R. S. Congalton, R. G. Oderwald, and R. A. Mead. Assessing landsat classification accuracy using discrete multivariate analysis statistical techniques. *Photogrammetric Engineering and Remote Sensing*, 49(12):1671–1678, 1983.

[9] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley and Sons, Inc., New York, 2001.

[10] R. L. Easton and W. Noel. The multispectral imaging of the Archimedes Palimpsest. *Gazette du Livre Medieval*, 45:39–49, 2004.

[11] T. Fawcett. An Introduction to ROC Analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[12] R. A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7:179–188, 1936.

[13] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, 1990.

[14] J. D. Gaskill. *Linear Systems, Fourier Transforms, and Optics*. John Wiley and Sons, New York, 1978.

[15] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[16] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, New Jersey, 2001.

[17] O. Gualdron and H. H. Arsenault. Locally nonlinear matched filtering. *Optics Letters*, 18(22):1949–1951, 1993.

[18] M. Handford. *Where's Waldo?* Little Brown and Co, United States, 1987.

[19] C. F. Hester and D. Casasent. Multivariant technique for multiclass pattern recognition. *Applied Optics*, 19(11):1758–1761, 1980.

[20] J. L. Horner and P. D. Gianino. Phase-only matched filtering. *Applied Optics*, 23:812–816, 1984.

[21] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[22] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:498–520, 1933.

[23] R. Huang, Q. Liu, H. Lu, and S. Ma. Solving the Small Sample Size Problem of LDA. *ICPR'02*, 3, 2002.

[24] F. V. Jensen. *An Introduction to Bayesian Networks*. Springer-Verlag, New York, 1996.

[25] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis Fifth Edition*. Prentice Hall, New Jersey, 2002.

[26] J. P. Knapp and Jr. R. L. Easton. Optical correlation using bitonal magnitude-only spatial light modulators. *Practical Holography XIV*, 3956A:233–240, 2000.

[27] B. V. K. Vijaya Kumar. Minimum-variance synthetic discriminant functions. *J. Opt. Soc. Am. A*, 3(10), 1986.

[28] B. V. K. Vijaya Kumar. Tutorial survey of composite filter designs for optical correlators. *Appl. Opt.*, 31(23), 1992.

[29] B. V. K. Vijaya Kumar and L. Hassebrook. Performance Measures for Correlation Filters. *Appl. Opt.*, 25:2997–3006, 1990.

[30] B. V. K. Vijaya Kumar and A. Mahalanobis. Recent Advancements in Composite Correlation Filter Design. *Asian Journal of Physics*, 8(3), 1999.

[31] B. V. K. Vijaya Kumar, A. Mahalanobis, and R. D. Juday. *Correlation Pattern Recognition*. Cambridge University Press, New York, 2005.

[32] B. V. K. Vijaya Kumar, M. Savvides, C. Xie, K. Venkataramani, J. Thornton, and A. Mahalanobis. Biometric verification with correlation filters. *Applied Optics*, 43(2), 2004.

[33] Y. Li, Z. Wang, and H. Zeng. Correlation Filter: An Accurate Approach to Detect and Locate Contrast Character Strings in Complex Table Environments. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 26(12), 2004.

[34] A. Mahalanobis, B. V. K. Vijaya Kumar, and D. Casasent. Minimum average correlation energy filters. *Appl. Opt.*, 26(17):3633–3640, 1987.

[35] A. Mahalanobis, B. V. K. Vijaya Kumar, S. Song, S. R. F. Sims, and J. F. Epperson. Unconstrained Correlation Filters. *Appl. Opt.*, 33(17):3751–3759, 1994.

[36] M. Maragoudakis, E. Kavallieratou, and N. Fakotakis. Improving handwritten character segmentation by incorporating Bayesian knowledge with support vector machines. *IEEE Proc. Of Int. Conf. Audio Speech and Signal Processing ICASSP2002*, Orlando-Florida, 2002.

[37] T. Melzer. *Generalized Canonical Correlation Analysis for Object Recognition*. PhD thesis, Vienna University of Technology, Institute of Automation, 2002.

[38] I. Milho, A. Fred, J. Albano, N. Baptista, and P. Sena. An Auxiliary System for Medical Diagnosis based on Bayesian Belief Networks. *Proc. 11th Portuguese Conference on Pattern Recognition*, RECPAD, 2000.

[39] R. Netz. *The Works of Archimedes: Volume I, The Two Books On The Sphere And The Cylinder*. Cambridge University Press, United Kingdom, 2004.

[40] R. Netz and W. Noel. *The Archimedes Codex*. Weidenfeld and Nicolson, London, 2007.

[41] J. A. Parker. *Image Reconstruction in Radiology*. CRC Press, Boston, 1990.

[42] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach (Second Edition)*. Prentice Hall, New Jersey, 2002.

[43] M. Savvides, B. V. K. Vijaya Kumar, and P. K. Khosla. Corefaces - Robust Shift Invariant PCA based Correlation Filter for Illumination Tolerant Face Recognition. *Proc. IEEE*, 1063-6919, 2004.

[44] J. R. Schott. *Remote Sensing: The Image Chain Approach*. Oxford University Press, New York, 2007.

[45] R. Singh and B. V. K. Vijaya Kumar. Performance of the extended maximum average correlation height (EMACH) filter and the polynomial distance classifier correlation filter (PDCCF) for multi-class sar detection and classification. *Proc. SPIE*, 4727, 2004.

[46] Rajan K. Singh. Advanced correlation filters for multi-class synthetic aperture radar detection and classification. Master's thesis, Carnegie Mellon University, 2002.

[47] G. Strang. *Linear Algebra and its Applications*. Academic Press, Inc., Orlando, Florida, 1980.

[48] G. Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, Massachusetts, 1986.

[49] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 1991.

[50] D. J. Walvoord and R. L. Easton. Variations of matched filtering for reduced noise amplification. *Proc. SPIE*, 5298:59–69, 2005.

[51] D. J. Walvoord and R. L. Easton. Adding contextual information to improve character recognition on the Archimedes Palimpsest. *Proc. SPIE*, 6500, 2007.

[52] D. J. Walvoord and R. L. Easton. Matched filter approximations for suppressing noise amplification. *Proc. IEEE Western New York Image Processing Workshop*, September 2004.

[53] D. J. Walvoord, R. L. Easton, K. T. Knox, and M. Heimbueger. Enhancement and character recognition of the erased colophon of a fifteenth century Hebrew prayer book. *Proc. SPIE*, 5676:157–166, 2004.

[54] Hua Yu and Jie Yang. A Direct LDA Algorithm for High-Dimensional Data – with Application to Face Recognition. *Pattern Recognition*, 34:2067–2070, 2001.

[55] N. L. Zhang and D. Poole. Exploiting Causal Independence in Bayesian Network Inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.

Index