Shape and Color Features
for the
Recognition of Roadside Objects

A Senior Thesis Presented to
The Faculty of the School of Arts and Sciences
Brandeis University

Undergraduate Program in Computer Science
Antonella Di Lillo and James A. Storer, Advisors

In partial fulfillment of the requirements for the degree of
Bachelor of Science

by
Nicholas Monath
May 2013
Acknowledgements

“Treat people as if they were what they ought to be and you help them become what they are capable of becoming.” - Johann Wolfgang von Goethe.

This work would not have been possible without my two incredible advisors, Antonella Di Lillo and Jim Storer, who embody Goethe’s quotation. Words cannot express my gratitude to them for their help and support throughout this project. They were there for me through thick and thin. I thank them for putting in countless hours every week despite their busy schedule. I also would like to express my appreciation for their mentorship and their belief in me. The positive reinforcement I received while working with Antonella and Jim has fueled my interest in Computer Science and inspired me to continue in the field. They are remarkable professors and enriched my experience at Brandeis.

I would also like to thank Kevin Thomas, David Lasher (and Freddie) and Jordan Pardo for their help and contributions to this project. Their assistance was invaluable to the project.

On a personal level, I would like to thank my friends and family, especially my mother and father, Blake and Tom, for all their kindness and support.
Abstract

In this thesis, we contribute a supervised system for the recognition of roadside objects which are of interest to a navigational aid for individuals who are visually impaired. These objects of interest have known geo-location (latitude and longitude) and so the recognition of such objects along with GPS measurements would allow the navigational aid to calculate a more precise geo-location of its user than GPS alone. The proposed recognition system is a fully generalizable system and can be used to recognize other objects with only a change of training data. Additionally, we contribute a segmentation method, which is used in conjunction with the recognition system. We provide classification experiments and results using our method and show that our approach performs competitively with techniques from the literature.
Contents

1 Introduction 5

2 Survey of Previous Works 9
   2.1 Assistive Technology .......................... 9
   2.2 Projective Geometry .......................... 12
   2.3 Object Recognition .......................... 13

3 Background 16
   3.1 Fourier-Polar-Fourier Transform ............... 16
   3.2 SIFT .......................................... 18
   3.3 Fisher’s Linear Discriminant Analysis .......... 22
   3.4 Vector Quantization .......................... 23
   3.5 Density Based Spacial Clustering of Applications with Noise .... 26
   3.6 Supervised Learning .......................... 29

4 A Preliminary Case Study 30
   4.1 Proposed Approach ........................... 31
   4.2 Experiments and Results – Silhouette Database .... 36
   4.3 Experiments and Results – User Acquired Data .... 37

5 An Overview of the Guidance System 39

6 Proposed Object Recognition Approach 42
   6.1 Feature Extraction ............................ 43
   6.2 A Supervised System for Classification .......... 47
   6.3 Classification Experiments and Results Using Perfect Segmentation .... 48
   6.4 Automatic Segmentation Method ................ 51
   6.5 Classification Experiments and Results Using Automatically Segmented Data .......... 56
   6.6 A New Object Recognition Approach:  ............. 58
      The Voting Method ........................... 58
   6.7 Classification Using SIFT ...................... 61
CONTENTS

7 Location Informed Object Recognition
   7.1 Previous Approach
   7.2 Proposed Approach
   7.3 Experiments and Results

8 Future Work and Conclusions

A Sample Dataset Imagess
Global Positioning System (GPS) locators have become common place in smartphones and similar devices. While these devices (along with standalone GPS units) can provide excellent navigational assistance to those driving, the devices cannot provide precise assistance on a finer scale due to the error in the geo-location (latitude and longitude) measurements. A system that can provide accurate navigational assistance on a small scale, such as a step scale for humans or the inch scale for robots, has numerous applications. Having such a system would assist an autonomous agent in an urban environment. Agents could be autonomous mailmen, autonomous meter-maids, ground-level surveillance drones, etc. Additionally, such a guidance system would be invaluable in assistive technologies for individuals who are visually impaired.

Over 21.5 million American adults, about seven percent of the population, are visually impaired according to the American Foundation for the Blind [2]. Significant research into aids for individuals who are visually impaired has been done and smartphone applications offering assistance have been created. These applications help users with tasks such as reading restaurant menus, counting dollar bills, and sending text messages. There is, of course, always a need for new assistive technologies. In
particular, there is the need for an application to provide navigational assistance, especially in urban areas. Of the 21.5 million adults with visual impairment, all but 4.5 million live in a metropolitan area. Guidance systems have been (and continue to be) developed by universities and companies and have proven successful in certain environments. However, none of these technologies provide a suitable replacement for the traditional assistive guides such as the white cane, nor do they provide assistance navigating an unknown city street [22].

Developing such a guidance system requires extensive research over a broad range of topics, such as decision making and planning, computer vision, hardware systems for guidance, and human computer interaction. In this thesis, I propose several techniques in computer vision that could be utilized by such a guidance system. These techniques include the recognition of objects common to urban environments, such as fire hydrants, postboxes, fire alarm boxes, and street signs. The techniques presented are designed to work as a “black box”, which could easily be incorporated into a guidance system. The guidance system for which these techniques are proposed is one that provides assistance to the user by maintaining an accurate geo-location of the user. In essence, with a precise geo-location of the user, the system could use information from map sources to provide assistance. The guidance system is a “vision-first” system; one that is based on image and video processing rather than laser scanners, gyroscopes or pedometers. In this way, the system could be implemented with little hardware. The system would require a smart-phone with a built-in camera and GPS device. The GPS locator would provide an estimate of the user’s location. Typically, a cell-phone quality GPS is accurate within 10-30 meters. The camera would consistently take photos of the user’s surroundings. These photos would be queried against Google Maps Streetview data of the same area and projective geometry would be used to determine the precise geo-location of the user. The recognition of objects such as
CHAPTER 1. INTRODUCTION

fire hydrants and postboxes can help provide an even more accurate location of the user. Since the latitude and longitude of such objects is maintained in city plans, the location of the objects could be used to refine the user’s location provided the guidance system could recognize and locate these objects. Of course, the system is not quite this simple; many other factors must be considered. For example, measures to alert the user of hazards would need to be implemented.

The recognition of these objects also provides benefits to a guidance system beyond the refinement of location as it would allow a user of the system to be alerted that one of these objects is nearby. For example, a user may wish to mail a letter while en route to a destination. It would be helpful if the guidance system could detect a postbox and direct the user to it. This feature could be beneficial not just to the guidance system described above, but also to guidance systems in general.

The recent rise in the computation power of mobile devices such as smart-phones and tablets coupled with their economical value has made such devices a prime choice for the platform of assistive technologies for people with disabilities. With processor speeds at between 1.2 and 1.4 GHz and 1 GB of RAM, the iPhone 5 and Samsung Galaxy SIII have the same computation power as a personal computer from just eight years ago [29] [30]. If further computation power is needed, data can be transmitted over Wi-Fi or a 4G network and processed on an industrial grade server. This power is offered at a fraction of the cost of designing and building a product from scratch to run an assistive program for individuals with disabilities. The built-in high resolution camera of a smartphone makes the device capable of running software to assist people who are visually impaired.

In this work, we contribute a recognition system for five objects of interest to a guidance system: fire alarm call boxes, fire hydrants, postboxes, stop signs, and street signs. The objects were chosen because the geo-location of each maintained in
city plans. We also contribute a segmentation method, which is a part of the object recognition system. The object recognition system uses the segmentation method to extract the different shapes contained in an image and uses a supervised system of two dimensional shape classification to determine if any of the extracted shapes are an object of interest. The research for this system is divided into two parts, a preliminary case study with stronger assumptions and the generalization of that system, which are presented in Chapters 4 and 6 respectively. The rest of the work is organized as follows: a survey of prior work, a background of methods used in this thesis, a preliminary case study on the recognition of fire hydrants, an overview of the guidance system, the proposed object recognition system, initial work on location informed object recognition, and future work and conclusions. Experiments and results are included in appropriate sections.
Survey of Previous Works

2.1 Assistive Technology

Technologies aimed to assist an individual who is visually impaired in achieving greater mobility have been developed both commercially and at academic institutions. These technologies are meant to allow for safer maneuvering within hazardous or unfamiliar environments. The aids may inform users of obstacles or potential hazards, such as an uneven sidewalk or road closure, alert the user to notable features in the surrounding environment such as a doorway or store front, or assist the user with directions to a destination. Despite the advances in modern technology, the most popular mobility aid among individuals who are visually impaired is the white cane or long cane. The white cane is swept side-to-side when walking, allowing a user to make sure they can safely continue to walk forward. The cane, of course, is also useful in negotiating unfamiliar hallways in buildings as a user can sweep the cane against a wall to find turns in the corridor. Like the white cane, the second most popular mobility aid is not an assistive technology; it is the guide dog. Guide dogs are trained to assist individuals by warning of hazards and by negotiating obstacles. While these
“traditional” methods will more than likely maintain their popularity in the next few years, more and more technologically based aids are being developed and becoming part of the every day lives of individuals who are visually impaired.

Since the mid-1980s, the University of California Santa Barbara has worked on a personal guidance system, which utilizes GPS information as well as data collected by sensors in a backpack worn by the user to provide navigational assistance. The system maintains in a database a representation of the path the user is attempting to navigate [16]. While the paths in the published experiments are located within the UCSB campus, one can imagine extrapolating the system to city streets. One of the most important contributions of this work is the study of how to interface the system with the user. One reason why the white cane and guide dog have remained the most popular mobility aids is that they provide assistance in a way that is not intrusive nor irritating. A careful balance of the number of alerts given to the user must be maintained. A system which alerts or directs a user too often would be annoying and could result in the user ignoring the system’s warnings, which could have unsafe consequences. However, a system must not alert the user too infrequently that safety is compromised. Furthermore, the method of alerting must be considered. In [16], an auditory alert system is compared to a haptic alert system, which uses vibration patterns to inform the user of navigational directions. In addition, using “on course” cues, which inform the user that they are moving in the correct direction towards their destination were compared to using “off course” cues, which inform a user when they are moving in an incorrect direction. The haptic system with “on course” cues was preferred by the users in the experiments done by the authors. The system was evaluated on the metrics of precision, safety, ease, and privacy.

Unlike the work at UCSB, Joel Hesch and Stergios Roumeliotis of the University of Minnesota contribute a guidance system that can detect obstacles. The guidance
aid consists of a foot-mounted pedometer and sensing equipment (a 2D laser and 3-axis gyroscope) attached to a white cane. This system is designed for navigation indoors; it exploits the perpendicular nature of walls in the hallways of buildings to compute the distances necessary to determine the location of the user. The authors believe that their tool can be a building block for other aids, which can make use of Internet-based building map databases and even be extended to the navigation of unknown buildings [13].

A similar approach was used at the University of Stuttgart, where a guidance aid known as TANIA (Tactile-Acoustical Navigation and Information Assistant) has been developed to provide step-guidance to the blind and deaf. A new TANIA prototype was created and tested in 2011. The system leverages digital maps of buildings using GPS and RFID technology to find the exact location of the user. The system has at best one-step accuracy. It also has the capability of presenting the user with information about the surrounding areas, such as railway schedules at train stations or menus at restaurants, presented aurally or on a Braille display [14] [15]. Several commercial technologies such as BrailleNote GPS and Trekker provide similar services.

A promising new project in the field of assistive technology was developed as a part of a 36-hour challenge at the Copenhagen Institute of Interaction Design. The project, which is known as Blind Maps, provides navigational assistance using a Braille display. The Braille display provides instructions on how the user should move in order to reach their destination. These instructions are presented as a path of dots, which change according to the person’s movement. While the system is not an all-inclusive guidance system (it does not have any hazard detection), its simple, intuitive and non-intrusive interface makes the system desirable. Users who tried Blind Maps pointed out how they preferred the Braille display to an auditory system. Auditory systems require the use of headphones, which users said can be distracting.
in an unsafe way. *Blind Maps* also has a route rating feature, where routes are rated in terms of negotiability, safety and travel time/distance. The system gathers user data and feedback about the routes and is constantly re-ranking routes to provide better assistance [5].

### 2.2 Projective Geometry

Recall that the guidance system proposed in the introduction worked as follows. A smartphone is attached to the user (or held by the user) with the phone continuously taking pictures of the user’s surroundings. The GPS built into the smartphone provides an estimate (within few meters) of the user’s location. The pictures taken by the camera are then queried against the Google Maps Streetview data to obtain a more accurate geo-location of the user. If a precise geo-location is determined for the user, then navigation instructions can be provided with great detail to the user. Additionally, if objects with known geo-location can be recognized in the images taken by the user, a more accurate geo-location of the user can be obtained. This is, in essence, a method known as *Simultaneous Localization and Mapping* or *SLAM*. Techniques used in the algorithm, SLAM, are used by several other guidance systems. SLAM, which is essential to the study of robotics, is an algorithm for an autonomous agent to create a map of an unknown environment while navigating it. Traditionally, SLAM works by using laser measurements to find the position of walls and other obstacles relative to the agent and to record information about ‘landmarks’ in the environment. A landmark refers to any object discovered by the agent multiple times in his navigation of the environment. The agent is almost certain of the location of the landmark, allowing the agent to almost exactly determine his position relative to the landmark [9]. Recently, much work has been done in the related problem of
CHAPTER 2. SURVEY OF PREVIOUS WORKS

V-SLAM, a version of SLAM, which utilizes computer vision instead of (or in addition to) laser measurements. Two works which are of particular interest are that of Andrew Davidson, in the use of a single camera in SLAM, and Xavier Pérez, in the mapping and navigation of robots. Davidson contributes a method in which an agile moving camera allows for the detection of features, which can then be used for the mapping of the environment [6]. Pérez not only provides a solution to the SLAM problem, but also provides a navigation model in which robots can be instructed in their navigation of the environment with simple English commands such as ‘turn left in 10 feet’ or ‘turn 45°’ [23]. It is important to keep in mind the difference between the requirements of the proposed guidance system and those of these related problems. The guidance system would provide absolute geo-location, not a map based on relative distances. Instead of discovering landmarks as the environment is explored, the user of our sytem has a set of landmarks with known location. While in some ways this is a subset of the SLAM problem, it is different in that it is meant to work in crowded, difficult environments, such as urban areas, where traditional SLAM techniques would fail.

2.3 Object Recognition

Object recognition is a much studied problem in computer vision. The goal of the problem is to be able to determine what objects appear in a given image. In some cases, such as our own, we are interested in determining where in the image the objects appear. Typically, the recognition of these objects includes classification. Despite years of research there are few successful general purpose object recognition tools. There has however been much success in object recognition tools built for the recognition of specific objects.

One difficulty in object recognition is the fact that an image might contain not
only a single object but also background noise, other objects in the foreground etc. The object recognition task can be simplified if the image is segmented into its different shapes/regions. For this reason, the problem of image segmentation is closely related to object recognition. Like object recognition, image segmentation is a difficult problem and after years of research no one technique stands out. Similarly, segmentation is usually more effective when additional information about the image is leveraged rather than working *a priori*.

However, image segmentation, like objection recognition, is a difficult problem in computer vision and one that remains without an optimal technique. Common methods for segmentation include a clustering approach, where a feature vector is calculated for each pixel and then pixels are clustered (typically with the *k*-means algorithm). This feature vector is typically based on either color as in [4] and [28] or texture as in [8] and [18]. Other methods utilize edge detection such as in [1]; sometimes edge detection is used in conjunction with region-growing techniques as in [33].

Object recognition techniques typically fall in three categories: *model-based* approaches, *shape-based* approaches, and *appearance-based* approaches. Model-based approaches develop a representation of an object from simple geometric shapes such as squares, circles, triangles, etc. in two dimensions or cubes, spheres, pyramids, etc. in three dimensions. Shape-based approaches describe an object by features representing its contours. Appearance-based approaches extract features of the object typically in two-dimensions. These features are either extracted at the *local* level (a small group of pixels or porition of the object) or the *global* level (the entire object) [24].

One of the most well known tools for general purpose object recognition is an appearance-based method known as SIFT (Scale Invariant Feature Transform). Given
CHAPTER 2. SURVEY OF PREVIOUS WORKS

an image of an object, $A$, and an image, $B$, believed to contain the object, SIFT performs object recognition by selecting key-points, extracting feature descriptors from those points in both $A$ and $B$ and then matching the features of $A$ with those of $B$. In this way, image segmentation is not necessary, but can be used to limit the search space of $B$. SIFT will be described in greater detail in the next chapter.

Other object recognition techniques are inspired by the human visual system such as [7] and [26]. In [7], objects are described using an invariant transformation in the frequency space; this work is is presented in the next chapter. In [26], features are extracted in a hierarchical manner, applying a Gabor filter to each scale space image of the hierarchy. The work claims the feature extraction mimics the ventral stream in the visual cortex, that it first obtains invariance to position and scale and later to the viewpoint. Perhaps in contrast, [12] presents a multi-component model for object recognition in which objects are grouped not only into different classes, but into different views or components on an intra-class level. The work achieves strong results on the PASCAL database using a contour detection system to isolate areas of interest in the image along with a segmentation method and a trained Support Vector Machine for classification. This is a brief overview of a few notable object recognition techniques; a more complete survey can be found in [24] and [3].
3

Background

This chapter presents several of the key components of the segmentation and classification methods utilized in the remaining chapters. These components include techniques in signal processing and feature extraction, statistical methods for variable analysis, and clustering methods.

3.1 Fourier-Polar-Fourier Transform

The Fourier-Polar-Fourier Transform (FPFT) of an image was shown to be a state-of-the-art feature descriptor for texture segmentation in [8]. The same descriptor was successful in the problem of shape classification and retrieval in [7], outperforming several of the top methods such as the Inner-distance method presented in [19] and the Generative Models of [31]. The descriptor transforms images into the frequency domain with a series of transformations to mimic the behavior of recognition tasks in the human visual system. The FPFT provides an abstract representation of an image that is rotation and shift invariant, making it an excellent choice as a descriptor for images. As in [7], we use the FPFT descriptor in the machine learning task of shape recognition. However, some modifications have been made to the learning algorithm,
which will be discussed later in the work.

Formally, the transform is defined as: Given an \( m \times n \) grayscale image \( I \) apply a discrete 2-dimensional Fourier transform to \( I \) to get \( I' \):

\[
I'(u, v) = \frac{1}{nm} \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} I(x, y) \exp \left( -2\pi i \left( \frac{ux}{m} + \frac{vy}{n} \right) \right)
\]  

(3.1)

\( I' \) is then converted into polar coordinates, using a Polar transform, resulting in \( I'' \):

\[
I''(r, \theta) = I'(x, y)
\]  

(3.2)

where

\[
\begin{align*}
r &= \log_a \left( \frac{\rho}{\rho_0} \right) \\
\theta &= \frac{S}{2\pi} * \eta \\
\rho &= \sqrt{x^2 + y^2} \\
\eta &= \arctan \left( \frac{y}{x} \right) \\
a &= \exp \left( \frac{\ln \rho_{\text{max}}}{R} \right)
\end{align*}
\]

where we discretize the polar space into rings and sectors such that values of \( r \) are between 0 and \( R \), and values of \( \theta \) are between 0 and \( S \). \( \rho_0 \) is the radius of the inner most ring and \( \rho_{\text{max}} \) is the radius of the visual field—the largest radius. Finally, we apply a 1-dimensional Fourier transform to the rows of \( I'' \) to get the final image \( I^* \):

\[
I^*(u, y) = \frac{1}{n} \sum_{x=0}^{n-1} I''(x, y) \exp \left( -2\pi i \left( \frac{ux}{m} \right) \right)
\]  

(3.3)

for every column \( y \) from 1 to \( m \).

From this resulting image, \( I^* \), we obtain the feature descriptor or signature (later denoted \( \text{SIG}(I) \)) of image \( I \). The signature is defined as the image \( I^* \) reshaped column-wise into a \( 1 \times nm \) vector. While we will not provide mathematical proof of
the rotation and shift invariance of the FPFT, Figures 3.1 and 3.2 demonstrate those properties.

Figure 3.1: FPFT Transform. From left to right: initial image, image after application of 2D Fourier transform, image after application of Polar transform, image after application of 1D Fourier transform

Figure 3.2: FPFT Transform applied to a rotated, scaled and shifted version of the image above. From left to right: initial image, image after application of 2D Fourier transform, image after application of Polar transform, image after application of 1D Fourier transform

### 3.2 Scale-Invariant Feature Transform for Key Point Localization and Feature Description

The Scale-Invariant Feature Transform (SIFT) is an algorithm for both the detection of key points in an image and the extraction and matching of features in images. SIFT was considered state-of-the-art when it was first published in 1999. Since then it remains one of the most powerful object recognition techniques. SIFT utilizes a consistent algorithm for locating key points as well as a robust feature descriptor for
those key points to perform image matching and object recognition at a high level.

A difficulty in object recognition is to determine \textit{a priori} the region of an image which contains an object. Of course, an extreme upper bound, the entire image, may be used, but this will most likely yield unsatisfactory results. SIFT addresses this problem by extracting features from a set of key points in the image. The number of keypoints is on the order of one-thousand for a reasonable size image and the keypoints are chosen in a consistent way. By “consistent”, we mean the key points of an object in one image are similar in relative location to the key points of the same object in a different image. Experiments in [21] proved this property.

The key points selected in the SIFT algorithm are the maxima and minima of a difference of Gaussian function applied to the scale space of the image. The transformation of image $I$ to the image in scale space, $I'$, is defined in terms of a scale parameter, $t$, as:

$$I'(x, y) = \frac{\exp \frac{-x^2+y^2}{4t}}{\sqrt{2\pi t}} I(x, y)$$ \hspace{1cm} (3.4)

The transformation is applied recursively with several values of $t$, resampling the image between each step. Maxima and minima are found by comparing each pixel to its eight neighbors. If a pixel is a maxima or minima compared to its neighbors for each value of $t$, then it is considered a key point [21].

From each key point, a feature descriptor is extracted. A circular window is created around the key point with radius proportional to the scale value of the key point. For each pixel value in that window, we calculate it’s magnitude, $m$, and orientation, $\theta$, value by the equations:

$$m(x, y) = \sqrt{(I'(x, y + 1) - I'(x, y - 1))^2 + (I'(x + 1, y) - I'(x - 1, y))^2}$$ \hspace{1cm} (3.5)
\[\theta(x, y) = \arctan \left( \frac{(I'(x, y + 1) - I'(x, y - 1))}{(I'(x + 1, y) - I'(x - 1, y))} \right) \tag{3.6}\]

where \(I'\) is the Gaussian smoothed image with the scale \(t\) of the key point. We assign the window 1 of 36 possible overall orientation values by creating a 36-bin histogram of \(\theta\) values for each pixel in the window. The overall orientation is the largest bin in the histogram. If there is a tie, a feature descriptor is extracted using each of the largest of bin values as the overall orientation. Next, the \(m\) and \(\theta\) values are normalized to the overall orientation of the window and we divide the window into 16 non-overlapping blocks arranged in a \(4 \times 4\) grid. We find the orientation of each block in the same way we found the overall orientation of the window and normalize the \(m\) and \(\theta\) to this orientation. With these values we create an 8-bin histogram of the orientation of the block. We combine the 8-bin histograms of each of the blocks together into a \(16 \times 8 = 128\) element feature descriptor for the key-point. The descriptor is normalized to unit length and thresholded for illumination invariance [24] [20].

The combination of key point localization and feature descriptors provides the model for object recognition using SIFT. Given an image \(A\) containing an object of interest and an image \(B\), in which we hope to find the object in \(A\), we apply the SIFT algorithm to each image, resulting in a set of key points \(F_A\), a set of descriptors for those key points \(D_A\) for image \(A\), and a set \(F_B\) and \(D_B\) for image \(B\). We match the descriptors \(D_A\) to descriptors in \(D_B\) using a slightly modified nearest neighbor method. Traditionally, nearest-neighbor matching means that \(d_a \in D_A\) is matched to the closest element of \(D_B\), say \(d_b^*\). However, in this case \(d_a\) is only matched to \(d_b^*\) if the distance between \(d_a\) and \(d_b^*\) multiplied by a threshold value is less than the distance of \(d_a\) to any other \(d_b \in D_B\). The distance between descriptors is measured by Euclidean distance. The matched key points can be (if necessary) clustered using the
Hough Transform as suggested by [20]. We also use the clustering algorithm DBSCAN later in this work and achieve optimal results. Figure 3.3 shows SIFT point matches between two images.

Figure 3.3: SIFT descriptors from the image on the top are matched to those of the image on the bottom. The matches are then clustered by their coordinate position in the image for emphasis.
3.3 Fisher’s Linear Discriminant Analysis

Linear discriminant analysis is a technique used to discover the set of variables which best defines a class of features. Given a set of $k$-dimensional feature vectors $X = \{x| x = \langle x_1, x_2, ..., x_k \rangle\}$, each with a class label in the set $Y$, let $X_y$ represent the feature vectors of class $y \in Y$. The purpose of linear discriminant analysis is to determine a linear combination of the variables $x_1, x_2, ..., x_k$, which best distinguishes each set $X_y$ from the feature vectors of every other class. Fisher’s linear discriminant analysis differs from linear discriminant analysis in that it does not make the assumptions that the mean of each $X_y$ is normally distributed and that covariance of each class is equal. We can reduce the dimensionality of $X$ by removing the least important variables from the resulting linear combination. Such a dimensionality reduction will create a more robust definition of features, allowing for a more powerful classifier.

Let $\mathbf{w}$ be the linear combination of the variables $x_1, x_2, ..., x_k$ which best distinguishes each set $X_y$ from the feature vectors of every other class. The metric Fisher presents to represent the separation of classes is:

$$\frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_W \mathbf{w}}$$

(3.7)

where

$$S_B = \sum_{y \in Y} (\mu_y - \bar{x})(\mu_y - \bar{x})^T$$

(3.8)

$$S_W = \sum_{y \in Y} \sum_{\xi \in X_y} (\xi - \mu_y)(\xi - \mu_y)^T$$

(3.9)
\( \mu_y = \text{mean}(X_y) \) \hspace{1cm} (3.10)

\( \bar{x} = \text{mean}(X) \) \hspace{1cm} (3.11)

\( S_B \) is a matrix representing the distance of the feature vectors between the different classes and \( S_W \) is a matrix representing the distance of the feature vectors within a certain class. The optimal linear combination of variables is \( w^* \) such that:

\[
w^* = \arg \max_w d(w) = \frac{w^T S_B w}{w^T S_W w}
\] \hspace{1cm} (3.12)

Notice that the function \( d(w) \) is large when the distance of feature vectors between classes is large and the distance within a class is small. This is precisely what we would want for our distribution of data [32].

While there are several ways to use Fisher’s LDA for dimensionality reduction, we do the following to remove the less important components. Given an user-specified value, \( \text{cut} \), we remove a variable, \( x_i \) if the coefficient of \( x_i \) in \( w, w_i \), is less than \( \text{cut} \times \text{sum}(w) \). Additionally, we use the Fisher coefficients of each variable to create a weighted distance function used in shape classification.

### 3.4 Vector Quantization

Vector Quantization is a clustering technique, which is defined as a mapping between a dataset of \( m \)-dimensional vectors, \( \mathcal{R}^m \), and a codebook: \( Q : \mathcal{R}^m \rightarrow \mathcal{C} \). The codebook, \( \mathcal{C} \), is a set of \( k \) centroids, each of which is \( m \)-dimensional. The vectors of the dataset, \( \mathcal{R}^m \), are each mapped to the “closest” centroid in the codebook. Various metrics,
including Euclidean, Manhattan, and Chebychev distance can be used to determine the what makes one centroid “closer” to a vector in $\mathbb{R}^m$ than another [11].

The objective of vector quantization is to determine a codebook, which minimizes the error of the mapping. To do so vectors are broken into $k$ clusters and a single centroid is obtained from each group. There are several methods to perform the clustering. In our work, we employ the well known Lloyd’s algorithm [11], with the slight modification proposed by [17].

Lloyd’s algorithm begins by partitioning the input data into $k$ initial sets. Typically, the partitioning is done at random or with a simple heuristic. A centroid is extracted from each set; the centroid is the average of all the vectors in the set. Next, the data is repartitioned based on the new centroids. Each data point is assigned to the nearest centroid in terms of the distance metric. The result is a new clustering of the data. From these new clusters we calculate a new set of centroids. We repeat the computation of centroids and repartitioning of data until the re-clustering of data is identical to the previous clustering or until the average distortion of the codebook of the current step is within $\epsilon$ of the average distortion of the previous step [11]. The value $\epsilon$ is a very small positive number and distortion is a user defined metric.

The authors Ioannis Katsavounidis, C.-C. Jay Kuo, and Ben Zhang of [17] provide a new technique for the initialization of the codebook in Lloyd’s algorithm. We have used their method in this work. The process for initializing the codebook begins by calculating the norm of all the vectors in the training set. The metric used for the norm is the square root of the inner product of a vector times by itself, that is, for the column vector $x$, the norm of $x$ is $\|x\| = \sqrt{x^T x}$, where $T$ is the transpose operator. The vector with the maximum norm is selected as the first element in the codebook. The rest of the elements in the codebook are chosen one at a time as the vector with the largest distance from the elements of the codebook. This modification accelerates
the convergence of the algorithm and reduces the error of the mapping [17].

**Algorithm 1** Lloyd’s Algorithm

```
procedure Cluster(X, k)  \(\triangleright\) Input: A dataset, X, of \(m\)-dimensional vectors and a number of centroids, \(k\)  \(\triangleright\) Output: An index and codebook containing the class of each vector in \(X\) and the corresponding centroids respectively

Let \(C\) be a codebook which will contain \(k\) centroids
Let \(Idx\) store the index into \(C\) for each value of \(X\)

Step 1. Initialize \(C\) using the method described in [17]
Step 2. Assign each vector in \(X\) to its nearest centroid in \(C\), create \(Idx\) accordingly
Step 3. Recompute the centroids of each cluster, update \(C\) accordingly
Step 4. Assign each vector in \(X\) to its nearest centroid in \(C\), update \(Idx\) accordingly

Repeat Steps 3 and 4 until re-clustering of data is identical to the previous clustering or until the average distortion of \(C\) in the current step is within \(\epsilon\) of the average distortion of \(C\) in the previous step, where \(\epsilon > 0\).

return \(Idx\) and \(C\)
end procedure
```

Figures 3.4 and 3.5 illustrate vectors in the space \(\mathbb{R}^2\) clustered using Lloyds algorithm with \(k = 4\) centroids.
Density Based Spacial Clustering of Applications with Noise

Density Based Spacial Clustering of Applications with Noise or DBSCAN is a clustering technique developed by Martin Ester, Hans-Peter Kriegel, Jorg Sander and Xiaowei Xu [10]. One of the most significant advantages DBSCAN has over vector quantization is that DBSCAN does not require a predefined, fixed-size codebook, instead the size of the codebook is determined by the estimated density distribution of the data. The only parameter DBSCAN requires the user specify is the minimum number of elements in a cluster, which we will refer to as MinPts.

In contrast to the definition of a cluster in the vector quantization algorithm, a cluster in the DBSCAN algorithm is defined as a set of points of at least a certain cardinality and density. Formally a cluster, \( C \) is defined as a non-empty subset of the
dataset, $D$, satisfying:

$\forall p, q : \text{if } p \in C \text{ and } q \text{ is density-reachable from } p, \text{ then } q \in C$

$\forall p, q \in C : p \text{ is density-connected to } q.$

The vector, $p$, is density-reachable from $q$ in the dataset $D$ if $\text{DIST}(p, q) \leq \epsilon$ and $|X| \geq \text{MinPts}$, where the set $X = \{x \mid x \in D, \text{DIST}(x, q) \leq \epsilon \}$. Additionally, a vector $p$ is density-reachable from a vector $q$ if there exists a chain of vectors $p_1, p_2, \ldots, p_n$, $p_1 = q, p_n = p$ such that $p_{i+1}$ is density reachable from $p_i$. A vector $p$ is density-connected to a point $q$ if there is a vector $r$ such that $p$ and $q$ are density-reachable from $r$. Note that the neighborhood radius parameter, $\epsilon$, may either be specified by the user or calculated analytically.

The algorithm computes clusters by beginning with a vector $p$ in the dataset and calculating the set $X_p = \{x_p \mid x_p \text{ is density-reachable from } p \}$. If this set $X_p$ satisfies the properties of a cluster, then we appropriately label the vectors in the set with a cluster number and continue. If not, we mark $p$ as an outlier and select the next vector in the database. We continue in this way until every vector is labeled. Two clusters, $C_1$ and $C_2$ may be merged if the distance, $\text{DIST}(C_1, C_2) = \min_{p \in C_1, q \in C_2} \text{DIST}(p, q)$ is less than $\epsilon$ [10].
Algorithm 2 DBSCAN Algorithm

procedure Cluster($X$, $MinPts$, $\epsilon$ (Optional))

▷ Input: A dataset, $X$, of $k$-dimensional vectors and a minimum number of points for each cluster and the optional parameter, $\epsilon$. If $\epsilon$ is not given, it is estimated by the function below

▷ Output: An index giving a cluster ID to each vector in $X$

Let $Idx$ be a list containing the cluster ID for each vector in $X$ or $-1$ if the vector is an outlier, i.e. in no cluster

Let $C$ be a counter keeping track of the current number of clusters

for each unmarked vector, $v$, in $X$ do

Mark $v$

Let $N_v$ be the set of vectors within $\epsilon$ of $v$

if $|N_v| < MinPts$ then

Set $v$’s value in $Idx$ to $-1$

else

$C \leftarrow C + 1$

Set $v$’s value in $Idx$ to $C$

for each vector, $u$, in $N_v$ do

if $u$ is marked then

Mark $u$

Let $N_u$ be the set of vectors within $\epsilon$ of $u$

if $|N_u| \geq MinPts$ then

$N_v \leftarrow N_v \cup N_u$

end if

end if

if $u$ does not belong to a cluster then

Set $u$’s value in $Idx$ to $C$

end if

end for

end if

end for

end procedure
CHAPTER 3. BACKGROUND

3.6 Supervised Learning

The problem of supervised learning is defined as the approximation of an unknown function $f : X \rightarrow Y$, given a set of $N$ input-output pairs, $(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N)$. We call the approximation of the function $f$, the hypothesis, $h$. The set, $X = \{x_1, x_2, \ldots, x_N\}$, is known as the training set. The set $Y = \{y_1, y_2, \ldots, y_N\}$ is a set of labels. If $Y$ is a finite set, we call this problem of supervised learning classification; otherwise, if $Y$ is infinite, we call the problem regression. The goal is to be able to correctly classify an element $x_{n+1} \notin X$ as $h(x_{n+1}) \approx f(x_{n+1}) = y_{n+1}$ [25]. In our model, $X$ is a set of images that have been labeled with an object type in $Y$. Clearly, then $Y$ is a finite set and we are presented a classification problem. By determining an approximation to $f$ with $h$, we will have a function which given any image, will correctly classify the object in the image.
Recognition of Fire Hydrants: A Preliminary Case Study

The work of this thesis began as a final project for the class, Introduction to Scientific Computing. The goal of the project was to create a system to recognize whether or not a given image contained a fire hydrant, with the assumption that the fire hydrant would be colored red. The idea behind the project was enhancing the guidance of an autonomous car, in contrast to the motivation of this thesis. Autonomous, or self-driving, cars are expected in the consumer market in the near future. Recently, the technology industry has become intertwined with the automotive industry. Cars in the consumer market today can park themselves, alert drivers if they become too close to another vehicle and even connect to the Internet. A system which recognizes fire hydrants would allow an autonomous car to determine if a prospective parking space is next to a hydrant and therefore illegal. Despite the variation in motive between the work of this chapter and that of this thesis, the techniques described here contribute effectively as a constituent part of the proposed object recognition tool in Chapter 6.
CHAPTER 4. A PRELIMINARY CASE STUDY

4.1 Proposed Approach

The recognition of red fire hydrants is done with a supervised system for two-dimensional shape classification. The training data for the classifier is a set of images containing entirely black silhouetted shapes centered against a white background; the class label of each sample is the shape it contains. Figure 4.1 shows example silhouetted shapes. From each training sample, a feature vector is extracted. Row reduction is used to lessen the dimension of the feature vectors and vector quantization is used to determine representative vectors or centroids for each class of image. These representative elements are then used as a nearest neighbor classifier. The test samples must also be black silhouetted shapes against a white background so that they are handled properly by the classifier. A feature vector is extracted from the test sample and row reduction is applied as in the training stage. This feature vector is then classified by the nearest neighbor classifier labeling the test sample with its shape. As we wish to determine if a user acquired real world image, for example a city street, contains a red fire hydrant, we perform preprocessing on the test images to extract the silhouette of the fire hydrants. An image segmentation technique, which detects the red portions of an image is used to extract the hydrant.

This approach (not including the segmentation technique) was originally proposed in [7] as a classification and retrieval technique for two-dimensional shapes. The retrieval method consists of a training stage identical to the training stage of the classification method and a testing stage in which a testing sample is classified, say as class $c$, and the images with the top-$k$ closest dimensionally reduced feature vectors in the training data to the centroids of class $c$ are retrieved.
CHAPTER 4. A PRELIMINARY CASE STUDY

4.1.1 Training Stage

The training data for this method is drawn from a set of 20 silhouetted fire hydrant shapes against a white background and the MPEG-7 CE Shape database, which contains 70 different classes of shapes each with 20 samples in the same silhouetted format. The training data consists entirely of $n \times n$ images. The shapes in the database fall into a number of categories; many of which would not be found on an urban street. The shapes include: cars, horseshoes, hammers, turtles, etc. Additionally, the database contains non-sensical shapes such as spirals or fractals. See Figure 4.1 for examples of the training data. All together, the hydrant silhouettes and the MPEG-7 database consist of 1420 images; there are 71 classes of shapes, each with 20 samples.

For each training sample, an $n \times n$ image $I$ containing a shape silhouette, a $1 \times n^2$ dimension feature vector is extracted using the FPFT descriptor (see Section 3.1). This feature vector is a rotation and shift invariant representation of the image $I$. After the feature vector of each sample has been calculated, Fisher’s linear discriminant analysis is used to reduce the dimensionality of the feature vectors. Finally, representative vectors or centroids are calculated for each class of shape by applying vector quantization to the reduced feature vectors of each class. The number of centroids per class is a parameter specified by the user. These sets of centroids are used in a nearest neighbor classifier in the testing stage. Euclidean distance is the metric for vector distance in both the quantization and classification.

Figure 4.1: Sample images from the MPEG-7 Database and fire hydrant silhouette
CHAPTER 4. A PRELIMINARY CASE STUDY

Algorithm 3 Training Algorithm

procedure TRAIN(X, Y) \> Input: The set of training images X with class labels Y \> Output: A set of feature vectors containing k centroids for each class of image

FeatureVectors \leftarrow \text{A set which will contain the feature vectors of the training images}
Codebook \leftarrow \text{A set which will contain k centroids for each class of image}

for each \( x \) in \( X \) do

FeatureVectors.\text{Append}(\text{SIG}(x)) \> \text{See Section 3.1 on the FPFT for the def. of SIG}

end for

FeatureVectors \leftarrow \text{ROWREDUCTION(FeatureVectors)} \> \text{Using Fisher’s LDA}

Let \( FV_y \) represent the feature vectors of images with class label \( y \)

for each class label \( y \in Y \) do

\{\( FV_{y,\text{Index}} \quad FV_{y,\text{Codebook}} \} \leftarrow \text{VQ}(FV_y, k)

Codebook.\text{APPEND}(FV_{y,\text{Codebook}})

end for

end procedure

4.1.2 Testing Stage - Segmentation

In determining if a user acquired image contains a red fire hydrant, the red regions of the image are extracted using the segmentation technique presented in this section. Each of the red regions is silhouetted and centered against a white background. The extraction and silhouetting process converts the user acquired image into a series of binary shapes. These silhouetted shapes are then classified using the classifier described in Section 4.1.3 to determine if the image contained a fire hydrant. Figure 4.2 shows a test image and the extracted red region, which is a fire hydrant.

The segmentation method leverages on the color of the fire hydrant. The method uses a logical expression of the color values of an image to determine which pixels are red. Then a region growing technique is used to extract the \( n \) largest red regions from the image. Given an input image \( I \), convert \( I \) into the YCbCr colorspace. Then, convert the Cr component back into the RGB colorspace to obtain \( I' \). Next, following
binary (logical) image is created:

\[ I'' = (I'(\cdot, \cdot, R) < 10) \lor (I(\cdot, \cdot, G) > (20 + I(\cdot, \cdot, B))) \]  \hspace{1cm} (4.1) 

where the colon notation refers to a pixel-by-pixel traversal of the images and \( R \), \( G \), and \( B \) refer to their respective color planes. Expression 4.1 gives pixels in \( I'' \) a value of 0, the color black, if the pixel was red in the original image, and a value of 1, the color white, otherwise. A region growing technique extracts the black regions from \( I'' \). Traversing the image black-pixel-by-black-pixel, surround the pixel with a small window. If the window contains a certain percentage of black points, we try to expand the window. The expansion is done by adding on rows and columns (in all four directions), which have a certain percentage of black points, while maintaining the overall density of the black points of the entire window. When the frame stops growing, the region is extracted (i.e. the pixels in the region are set to white in \( I'' \)) and the next region begins growing. When all regions have been extracted, the \( n \) largest regions are selected. The shapes, which make up these regions, are smoothed using a median filter and any holes in the shape are filled.
CHAPTER 4. A PRELIMINARY CASE STUDY

Figure 4.2: An image and the extracted region of a fire hydrant using this segmentation method

4.1.3 Testing Stage - Classification

Given a testing sample, a $n \times n$ image $S$, of unknown class, a label will be calculated for $S$ using a nearest neighbor classifier built from the centroids of the training data. $S$ is assumed to be a silhouetted shape centered against a white background, just like the training data. $S$ may be a region extracted by the segmentation or a manually created silhouetted shape. A $1 \times n^2$ dimensional feature vector is extracted from the image using the FPFT descriptor just as was done in the training. The variables removed by Fisher’s linear discriminant analysis in the training stage are remove from the feature vector of $S$. The reduced sample $S$ is given the class label of the closest centroid in terms of Euclidean distance.
CHAPTER 4. A PRELIMINARY CASE STUDY

Algorithm 4 Classification Algorithm

\begin{algorithm}
\textbf{procedure} Classify(x) \triangleright Input: An image of unknown class, $x$ \triangleright Output: The class of the image $x$

$\text{Codebook} \leftarrow$ the set of $k$ centroids of each class of image, created during the training stage of the algorithm

$\text{signature} \leftarrow \text{SIG}(x)$

Reduce the dimensionality of signature in the same way as was done in the training stage

$\text{closest} \leftarrow \arg \min_{c \in \text{Codebook}} \text{DIST} (\text{signature}, c)$

\textbf{return} the class of the centroid closest

\textbf{end procedure}
\end{algorithm}

4.1.4 Testing Stage - Retrieval

The authors of [7] presented the classification method along with a content-based retrieval method. The retrieval method uses the same training stage as described in Section 4.1.1 and does the following to determine the most relevant training samples to a given test sample, an $n \times n$ image, $S$. Using the FPFT descriptor, a $1 \times n^2$ feature vector is calculated for $S$. This feature vector is reduced by removing the variables, which were removed by Fisher’s linear discriminant analysis in the training stage. $S$ is given a class label, $c$, using the method in Section 4.1.3. The top-$k$ retrieval results are found by calculating the $k$ closest feature vectors to the centroids of class $c$.

4.2 Experiments and Results – Silhouette Database

One classification experiment was performed using only the MPEG-7 database with the added silhouettes in shape of the fire hydrant. In the first experiment, the system is trained on the 1420 images (the 20 samples of the 70 classes in the database plus the 20 fire hydrant silhouettes) and is tested on the same images. The system was able to classify 95.00\% of the shapes correctly. The same experiment was performed in
[7] with the MPEG-7 database without the fire hydrant shape; 94.93% of the shapes were correctly classified.

In addition to the classification experiment, a retrieval experiment was performed. In this experiment, the system is trained on all 1420 and then queried, retrieving the top 40 closest matches to the test sample. The retrieval is scored by the recall—how many of the correct class of object are retrieved in the top 40 matches. The system scores 94.51% on the dataset of 1420 images. The same experiment was performed by the authors of [7] on the MPEG-7 database without the fire hydrant shape and scored 92.25%.

In both the classification and the retrieval experiment, the fire hydrant class of shape received a percent score. In the classification case, this means every shape was correctly matched as a fire hydrant. In the retrieval case, this means that of the top 40 images retrieved for each hydrant shape, 20 of those 40 images were fire hydrants.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Score without Hydrant</th>
<th>Score with Hydrant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification</td>
<td>94.93%</td>
<td>95.00%</td>
</tr>
<tr>
<td>Retrieval</td>
<td>92.25%</td>
<td>94.51%</td>
</tr>
</tbody>
</table>

### Table 4.1: Silhouette Data Results

#### 4.3 Experiments and Results – User Acquired Data

The second group of experiments were performed on user acquired real world data. We collected a dataset of 15 images containing red fire hydrants. The images contain hydrants at a variety of distances from the lens. Each image is segmented using the method explained in this chapter and the extracted regions are then classified by the system. The system was able to correctly identify 11 out of the 15 hydrants (73.33%).
CHAPTER 4. A PRELIMINARY CASE STUDY

Figure 4.3: The test samples of the user acquired dataset

Table 4.2: Classification Results

<table>
<thead>
<tr>
<th>Image Number</th>
<th>Result</th>
<th>Image Number</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Matched</td>
<td>9</td>
<td>Not Matched</td>
</tr>
<tr>
<td>2</td>
<td>Not Matched</td>
<td>10</td>
<td>Matched</td>
</tr>
<tr>
<td>3</td>
<td>Matched</td>
<td>11</td>
<td>Not Matched</td>
</tr>
<tr>
<td>4</td>
<td>Matched</td>
<td>12</td>
<td>Matched</td>
</tr>
<tr>
<td>5</td>
<td>Matched</td>
<td>13</td>
<td>Matched</td>
</tr>
<tr>
<td>6</td>
<td>Matched</td>
<td>14</td>
<td>Matched</td>
</tr>
<tr>
<td>7</td>
<td>Not Matched</td>
<td>15</td>
<td>Matched</td>
</tr>
<tr>
<td>8</td>
<td>Matched</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An Overview of the Guidance System

The object recognition system presented in this work was motivated by the guidance system for individuals who are visually impaired described in the introduction. A key component to that guidance system is determining a precise geo-location for the user and tracking that location. Figure 5.1 shows the motion of a user on a street in Waltham, MA; the red ‘L’ or ‘Landmark’ markers represent the locations at which the system recognized an object of interest. Recall that the guidance system assumes that a smart-phone (or comparable device) is attached to the user and at each time step a photograph is taken by the device. This photograph is processed along with the GPS data of the device and queried against the Google Maps Streetview database. By matching similar regions between the user’s photograph and the Streetview data, we can obtain a geo-location for the user with greater precision than the GPS estimate. Additionally, the image is searched for objects of interest, which have known geo-locations. The objects’ position in the image, along with the objects’ geo-location and the user’s estimated geo-location, is used to determine the location from which the image was taken.

To determine if an image contains one of the objects of interest we propose a
CHAPTER 5. AN OVERVIEW OF THE GUIDANCE SYSTEM

recognition system that is an extension of the system described in Chapter 4. While the experimental results of the preliminary object recognition tool demonstrated that the system did a good job matching silhouetted shapes of fire hydrants, the system had the limitation of the segmentation method, which would only extract red regions from an image so as to locate a red fire hydrant. One of the first steps of generalizing the preliminary work is to develop a segmentation method, which will extract any of the objects of interest from an image regardless of color. This segmentation method along with a modified classification approach is presented in Chapter 6.

Object recognition is, of course, a difficult problem and it is more than likely that the querying of the Streetview database and the recognition of landmarks will not provide a precise geo-location at every time step. A photograph taken of the city street might be too occluded by cars and people to be queried against Streetview data or searched for landmarks. Nevertheless, the system needs to maintain a geo-location for the user. While the location in this case could simply be estimated by the GPS of the phone, perhaps a more accurate geo-location can be extrapolated using probabilistic inference about the user’s past locations and environment.
Figure 5.1: An aerial map with the plotted path of the user tracked by the guidance system. Recognized landmarks are labeled in the image.
Proposed Object Recognition Approach

Building upon the preliminary work in object recognition presented in Chapter 4, we present, in this chapter, an improved system which can recognize five roadside objects and could be utilized by a mobility aid to help individuals who are visually impaired navigate in urban areas. Figure 6.1 shows a sample of the objects of interest. As before, two-dimensional shape classification is at the heart of the recognition system. In this chapter, we investigate several methods of feature extraction, perform experiments and present results using a supervised classification system. A different approach that combines some of the methods of feature extraction is then proposed. Finally, the performance of our approach is compared to the performance of a system which classifies using SIFT.

The chapter is organized as follows: a presentation of the supervised system; a description of feature extraction techniques; experiments and results using the perfectly segmented data; the automatic segmentation method; experiments and results using the automatically segmented data; a new approach for object recognition; and a comparison to classification using SIFT.
CHAPTER 6. PROPOSED OBJECT RECOGNITION APPROACH

Figure 6.1: The objects of interest: fire alarm call boxes, postboxes, fire hydrants, stop signs, and street signs

6.1 Feature Extraction

In this section, we present the different feature extraction methods that we have used. We then present results comparing the methods in Section 6.3 and 6.5.

6.1.1 Shape-FPFT

The first method of feature extraction, *Shape-FPFT*, is almost identical to the method of feature extraction used in Chapter 4. The Fourier-Polar-Fourier Transform is applied to a grayscale image and the image is reshaped column-wise from a two-dimensional matrix into a one-dimensional vector. Figure 6.2 shows a training sample after each stage of the transform and a portion of the resulting feature vector plotted with the transformed pixel values on the y-axis and the position in the feature vector on the x-axis.
CHAPTER 6. PROPOSED OBJECT RECOGNITION APPROACH

6.1.2 Edge-FPFT

The Edge-FPFT feature descriptor is defined as follows. Given an image \( I \), calculate the edge outline of the image \( E \) using the Sobel operator. Apply the FPFT to \( E \) and reshape the image matrix into a one-dimensional vector. The resulting vector is the feature vector for \( I \). Figure 6.3 shows the training sample from Figure 6.2 after edge detection, each stage of the transform, and a plot of a portion of the resulting feature vector.

Figure 6.3: A training sample after edge detection and after each stage of the FPFT transform and a portion resulting 1-D feature vector

6.1.3 Shape-FPFT & Edge-FPFT Combined

The combination of the Shape-FPFT and the Edge-FPFT feature vectors is defined as the concatenation of the two. The combination should give a more descriptive representation of the image, which should allow for a more powerful classifier.
6.1.4 Shape-FPFT & Edge-FPFT with Color

There are numerous ways to represent the color of an image. Yet it is difficult to create a mapping between the human perception of color and the numerical color values of an image. For this reason, a statistical approach is taken for the extraction of the Color feature from the image. An image $I$ is converted from the RGB colorspace to the YCbCr colorspace. Then a three-dimensional histogram is built from the Cb and Cr components of any pixel that was not white in the RGB image. The histogram has 27 bins in both the Cb and the Cr direction. This two-dimensional matrix is transformed into a one dimensional vector and normalized by multiplying by one over the sum of the histogram. Figure 6.4 shows a training sample, the sample in the YCbCr colorspace, the corresponding two dimensional histogram and one-dimensional vector.

A classifier, which uses the Color feature vector alone to recognize an image is typically not successful as multiple objects have similar colors. However, the Color feature vector can be used in addition to the other descriptors to create a more robust representation of the image. Given an image $I$, we define its Shape-FPFT & Edge-FPFT with Color feature vector as the concatenation of its Shape-FPFT feature vector with its Edge-FPFT and Color feature vectors.
Figure 6.4: A training sample, the sample in the YCbCr Colorspace, the histogram of the Cb and Cr components of the non-white pixels of the RGB image, and the one-dimensional vector.
6.2 A Supervised System for Classification

As with any classification system, there is a training stage and testing stage. The training samples are images containing a color shape against a white background. For each sample, a $k$-element feature vector is extracted. After the feature vector of each sample has been calculated, Fisher’s linear discriminant analysis is used to reduce the dimensionality of the feature vectors. Finally, representative vectors or centroids are calculated for each class of shape by applying vector quantization to the reduced feature vectors of each class. These sets of centroids are used in a nearest neighbor classifier in the testing stage. A Euclidean distance function is used. The Euclidean distance between two vectors, $\mathbf{x} = \{x_1, x_2, \ldots x_n\}$, $\mathbf{y} = \{y_1, y_2, \ldots y_n\}$ is defined as:

$$
\text{DIST}(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}
$$

(6.1)

The weighted Euclidean distance between two vectors is defined, given a set of weights $W = \{w_1, w_2, \ldots w_n\}$ as:

$$
\text{DIST}(\mathbf{x}, \mathbf{y}) = \sqrt{w_1(x_1 - y_1)^2 + w_2(x_2 - y_2)^2 + \ldots + w_n(x_n - y_n)^2}
$$

(6.2)

In our work, we experiment using the Fisher coefficients as the weights, $W$, of the Euclidean distance function.

The testing stage involves providing a class label to an image of unknown class. Given a testing sample, an image $S$, a nearest neighbor classifier built from the centroids of the training data gives $S$ a label. A $k$-dimensional feature vector is extracted from $S$. The sample $S$ is given the class label of the closest centroid to its feature vector in terms of the Euclidean distance.
6.3 Classification Experiments and Results Using Perfect Segmentation

To determine the effectiveness of the aforementioned feature descriptors, we compare the performance of the supervised classification system with each of our descriptors. We obtained a dataset of 200 images, 40 images of five classes of objects: fire alarm call boxes, postboxes, fire hydrants, stop signs, and street signs. The images in the database were taken around Waltham, MA using an iPhone 5-megapixel camera. The use of the cell-phone camera was meant to simulate the images that might be taken by the guidance system. The images in the dataset contain samples of the objects of interest taken from a variety of views. Figure 6.5 shows example images from our dataset. Appendix A contains a larger subset of the dataset.

Each image of the database was segmented by hand; that is for each image in the database, the object of interest is extracted using the image editing software GIMP and stored as a single shape against a white background. Twenty samples of each class of object (100 images in total) are selected as the training set for the classifier.

In this experiment, we use the classifier to determine the shape of the remaining 100 samples in our dataset. Note that the hand segmented version of the sample is classified not the original image. Through some initial experimentation, we discovered that results can be improved if a Gaussian filter is used to slightly smooth the image. We also normalized the samples so that a minimum square bounding box is taken of the shape and this bounding box is scaled (in nearly every case scaled down) to a 300 × 300 image. This normalizes the ratio of object to white space in the data.

Table 6.1 presents the results of the classification using each type of feature descriptor.
Table 6.1: Classification Results

<table>
<thead>
<tr>
<th>Feature Vector</th>
<th>Percent Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect Segmentation</td>
<td></td>
</tr>
<tr>
<td>Shape-FPFT</td>
<td>94</td>
</tr>
<tr>
<td>Edge-FPFT</td>
<td>92</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT</td>
<td>94</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT with Color</td>
<td>95</td>
</tr>
</tbody>
</table>

Table 6.1 shows that all of the above methods of feature extraction perform at a high level on the perfectly segmented data. Next, we present how the classifiers perform on automatically segmented data using the proposed segmentation described in the next section.
CHAPTER 6. PROPOSED OBJECT RECOGNITION APPROACH

Figure 6.5: Sample Images From Our Dataset
6.4 Automatic Segmentation Method

While much work has been done in the field of segmentation, we decided to contribute a method of our own given the limited scope of the problem. The segmentation method we propose is an automatic method based on color. The algorithm works by first taking an image $I$ and running vector quantization on the pixel values of $I$ to determine a color codebook. Nearest-neighbor classification is performed on each pixel, assigning it the nearest value in color codebook. Finally, the image is separated into its different regions by finding the connected shapes of each color in the codebook.

The first stage of the algorithm is to generate the color codebook. Several transformations to the image $I$ are applied before using vector quantization. First, $I$ is smoothed with a Gaussian filter; then $I$ is converted from the RGB colorspace to the YCbCr colorspace. The luminance is normalized by setting each pixel’s $Y$ value to 128. This is done because an object of a single color will often be made up of pixels with nearly identical $Cb$ and $Cr$ values, but a range of $Y$ values. Normalizing the luminance therefore makes the object more uniform in color, which will be necessary for the object extraction stage of the algorithm. The image is then converted back into the RGB colorspace. The color codebook is found by performing vector quantization on the pixel values of $I$. The number of colors in the codebook, $k$, is a user specified parameter.

Next, we create a new image $I'$ by classifying the color value of each pixel of $I$ to its nearest neighbor in the color codebook. The image $I'$ now contains just $k$ different colors, as opposed to $I$ which typically has several thousand. This reduction in color is done so we can extract from the image the objects of the each color. A mode filter is applied to $I'$ to remove specklingly of colors that might have occurred during the pixel classification. To extract the objects of each color from the image, the pixel
coordinates of the image are divided into a set for each color in the codebook; we have sets \( P_c = \{(x, y)|I'(x, y) = c\} \) for each \( c \in \text{color codebook} \). Additionally, we apply an edge detection method to the image to obtain an image \( E \). The pixels in \( E \) which are considered edge points have value 1 and all other pixels have value of 0. From the points, \( P_c \), we construct a graph with a vertex for every \((x, y)\) in \( P_c \) and edges between any two vertices, \((x, y)\) and \((x', y')\) such that \((x - x')^2 + (y - y')^2 = 1\) and \( E(x', y') = 0 \). To find the connected components of the graph, we use breadth first search. These connected components are the different shapes of a single color in the image \( I \). A minimum number of pixels for a connected component is set to eliminate small regions, which might be considered noise. Finally, the background color of the image, which is the most common color value, is discarded, that is no regions of that color are extracted. Next we show the results of applying the segmentation method to our dataset.
Figure 6.6: Top to Bottom, Left to Right: Original Image, Gaussian Filter applied to image, converted to YCbCr, luminance removed, and converted back to RGB, indexed image, indexed image after mode filter, edge outline of the image, final image segmentation.
Algorithm 5 Unsupervised Segmentation Algorithm

procedure UNSUPERVISEDSEGMENTATION($I, numColors, MinPts$)

▷ Input: An image $I$, the number of colors to be used in the segmentation and the minimum number of points in an extracted region  
▷ Output: A set of extracted regions from the image $I$

$I ← \text{SMOOTH}(I)$
$I' ← \text{CONVERTToYCbCr}(I)$
$I'(;:,1) ← 128$
$I' ← \text{CONVERTToRGB}(I)$
$E ← \text{binary image such that a pixel has value 1 if it considered an edge point by an edge detection method and a value of 0 otherwise}$

$\text{Index, Codebook} ← \text{VQ}(I', numColors)$
$I'' ← \text{Codebook}(\text{Index})$  
▷ Nearest Neighbor Classification

$backgroundcolor ← \text{the most common color in } I''$

for each $c$ in $\text{Codebook}, c \neq backgroundcolor$ do

$P_c = \{(x,y) | I''(x,y) == c\}$

Construct a graph $G(V, E)$ such that $V = \{(x,y) | (x,y) \in P_C\}$

and $E = \{[(x,y), (x',y')] | (x-x')^2 + (y-y')^2 = 1 \land E(x',y') = 0, (x,y), (x',y') \in P_C\}$

Use Breadth First Search to find the $\text{ConnectedComponents}$ of $G$

for each $cmp$ in $\text{ConnectedComponents}$ do

if $c \neq \text{Mode}(I'')$ and $|cmp| > MinPts$ then

return $cmp$

end if

end for

end for

end procedure

6.4.1 Experiments and Results of the Segmentation Method

In this thesis, we are more concerned with how well the segmentation method extracts the object of interest than with how well it partitions the image. It is at times difficult to judge exactly how an image should be segmented; consider an American Flag, it's a matter of opinion whether or not the “correct” segmentation would be the shape
as a whole or the stars and stripes individually. Therefore, scoring the segmentation of an image can be a difficult task. This being said, we score our segmentation solely on how well the object of interest is extracted from the image, measuring how many of the pixels of the object are correctly extracted and how many additionally pixels are incorrectly extracted.

To determine the effectiveness of the segmentation method, we apply it to the 100 images used as the testing set in the classification experiment. We compare the shape extracted by our automatic segmentation to the ground truth. Note that an incorrectly extracted pixel is typically a pixel picked up by the segmentation method that is not part of the object of interest. On average, the method extracts shapes with 80.80% accuracy and only 410.59 pixels, 0.0029% of the image, are incorrectly extracted.

<table>
<thead>
<tr>
<th>Object Class</th>
<th>Avg % Correctly Extracted</th>
<th>Avg Num Pixels Incorrectly Extracted (% of the Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fire Alarm Call Box</td>
<td>78.30</td>
<td>255.9 (0.0018%)</td>
</tr>
<tr>
<td>Post Box</td>
<td>84.17</td>
<td>199.7 (0.0027%)</td>
</tr>
<tr>
<td>Fire Hydrant</td>
<td>63.00</td>
<td>368.5 (0.0014%)</td>
</tr>
<tr>
<td>Stop sign</td>
<td>87.96</td>
<td>85.65 (0.0062%)</td>
</tr>
<tr>
<td>Street sign</td>
<td>87.16</td>
<td>1143.2 (0.0079%)</td>
</tr>
</tbody>
</table>

The segmentation method does a good job extracting the object of interest from an image. The method extracts on the order of ten to fifteen regions per image. Despite this relatively small number, we have no way of knowing which regions contain an object of interest and so each segmented region needs to be run through the classifier to determine its shape.
6.5 Classification Experiments and Results Using Automatically Segmented Data

Next, we will investigate how the classifier using the feature descriptors from Section 6.2 perform on the dataset of segmented images. For each of the 100 testing samples, we use the proposed automatic segmentation method to extract the object of interest. The classifiers are scored on how well they classify the 100 segments containing the object of interest.

<table>
<thead>
<tr>
<th>Feature Vector</th>
<th>% Correct Perfect Segmentation</th>
<th>% Correct Automatic Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape-FPFT</td>
<td>94</td>
<td>81</td>
</tr>
<tr>
<td>Edge-FPFT</td>
<td>92</td>
<td>76</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT</td>
<td>94</td>
<td>82</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT with Color</td>
<td>95</td>
<td>81</td>
</tr>
</tbody>
</table>

Table 6.3: Classification Results

Table 6.3 shows the results of this experiment using the different feature descriptors. The descriptors have similar performance classifying the automatically segmented data with around 80% accuracy. To improve the performance of the system on the automatically segmented data, we propose a different recognition approach based on the observation that some features may contribute more than others to the recognition of the object.
Figure 6.7: Sample test images with corresponding objects of interest extracted by the automatic method
6.6 A New Object Recognition Approach:

The Voting Method

Rather than combining the Shape-FPFT, Edge-FPFT and Color features of an image into a single feature vector, we propose a system that extracts the three features individually, building three separate classifiers. Each of these classifiers provides a class label for a given test sample and a simple decision algorithm is used to determine the final classification.

It is important to note the difference in the model of using three separate feature vectors and using a feature vector of the three combined. Using three separate feature vectors disassociates the shape, edge and color characteristics of the class of image. While this could draw criticism, it does provide a more lenient definition of a class of objects. This definition can provide for a more powerful classifier for the automatically segmented data. The automatically segmented data often contains just a portion of the object of interest or the object of interest amongst some background noise, in these cases comparing the shape, edge and color features separately can be an effective method of classification. We now provide descriptions of the training and testing stages of the algorithm. Figures 6.8 and 6.9 provide a flow-chart representation of the two stages of the algorithm.

6.6.1 Training

Like the approach described in Section 6.1, the proposed approach is a supervised system. In brief, the training stage builds three separate classifiers for the Shape-FPFT, Edge-FPFT and Color features of an image.

From each training sample, three features are extracted—the Shape-FPFT, Edge-
CHAPTER 6. PROPOSED OBJECT RECOGNITION APPROACH

Figure 6.8: The training stage of the new object recognition approach (the Voting Method)

FPFT, and Color features. The features are not combined, but rather stored in three separate groups. After the features have been extracted from each sample, Fisher’s linear discriminant analysis is used to reduce the dimensionality of the Shape-FPFT and Edge-FPFT vectors. The dimensionality of the Color feature is not reduced as it has relatively low dimensionality and its construction was such that none of its variables are unnecessary. Note that the Fisher cut used to reduce the dimension of the Shape-FPFT vectors need not be the same as the cut used to reduce the dimension of the Edge-FPFT vectors. The features are grouped by their type and their class. Vector quantization is applied to each group to compute representative vectors. Note that the number of centroids from each class need not be the same across the three types of features. These codebooks are used to perform nearest neighbor classification. Euclidean distance (or weighted Euclidean distance) is used as the distance metric for the Shape-FPFT and Edge-FPFT vectors. Manhattan or cityblock distance is used as the metric for the color features. The Manhattan distance between two vectors, \( \mathbf{x} = \{x_1, x_2, ..., x_n\}, \mathbf{y} = \{y_1, y_2, ..., y_n\} \) is defined as:

\[
\text{DIST}(\mathbf{x}, \mathbf{y}) = |x_1 - y_1| + |x_2 - y_2| + ... + |x_n - y_n| \tag{6.3}
\]
6.6.2 Testing

Given a test sample, we extract three feature vectors, just as in the training stage. The dimensionality of the Shape-FPFT and Edge-FPFT vectors are reduced in the same way as in the training. Then the nearest centroid to each of the three feature vectors is determined. Three classifications of the image are proposed by the class of the closest centroids. Let’s refer to these classifications as votes. For this reason, we refer to this method as the Voting Method. If there is a majority of votes for one class, the image is given that class label. If there is no majority, that is each of the three votes is for a different class, a tie-breaker is performed. The second closest centroid to the Shape-FPFT vector of the test sample is determined; the class of this centroid is added as a vote. Now if there is a majority, the image is given the class label of the majority; if not the classifier returns a null class indicating that the image could not be classified.


6.6.3 Experiments and Results

Using the Voting Method to classify the 100 samples in our testing dataset, we saw a dramatic improvement in the performance of the system on the automatically segmented data; the classifier works with 90% accuracy. Table shows the results of the classification experiment.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Percent Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perfect Segmentation</td>
<td>95</td>
</tr>
<tr>
<td>Automatic Segmentation</td>
<td>90</td>
</tr>
</tbody>
</table>

Of the ten automatically segmented images, which were not correctly classified by the system, three could not be classified and seven were misclassified. While we would hope for less misclassifications, this is an improvement over a strict nearest neighbor approach which assigns every object a class.

6.7 Classification Using SIFT

In this section, we compare our classifier to a classifier using SIFT. Given a training sample, SIFT is applied to obtain a set of descriptors. The descriptors of the samples are grouped by class. An image of unknown class is classified by first applying SIFT to that image, obtaining a set of descriptors. The test image’s descriptors are matched with the set of descriptors for each class of image, obtained in the training stage. The image is label with the class of the set of training descriptors with the largest number of matches. Note that the matches are calculated according to the method proposed by [20]; that is two SIFT descriptors $a$ and $b$ from the sets of descriptors $A$ and $B$. 
respectively are matched if and only if the distance between $a$ and $b$ multiplied by a threshold value is less than the distance between $a$ and every other element in $B$. Euclidean distance is used to measure the distance between descriptors.

Table 6.5 shows results for both our classifiers and the classifier using SIFT. It can be seen that the Voting Method outperforms our other classifiers and performs competitively with the classifier using SIFT.

<table>
<thead>
<tr>
<th>Method</th>
<th>Percent Correct Perfect Segmentation</th>
<th>Percent Correct Automatic Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voting Method</td>
<td>95</td>
<td>90</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT</td>
<td>94</td>
<td>82</td>
</tr>
<tr>
<td>Shape-FPFT &amp; Edge-FPFT with Color</td>
<td>95</td>
<td>81</td>
</tr>
<tr>
<td>SIFT</td>
<td>89</td>
<td>83</td>
</tr>
</tbody>
</table>
The object recognition done in Chapter 6 did not leverage on the data that would be available if the recognition were done as a part of the navigational guidance system, but rather generically recognized shapes. The geo-location from which a photograph was taken could provide a great deal of information to the object recognition system, in particular, which objects of interest are in sight.

The problem of location informed object recognition is defined as follows: given a photograph taken on a street at a specific geo-location estimated by GPS, \((lat, long)\), in a given direction, \(d\). We are interested in locating, in the image, an object of interest known to be viewable at \((lat, long)\) in direction \(d\). We make the assumption that a database is maintained of all objects of interest. Additionally, we make the assumption that if more than one view of the object is stored in the database, the optimal view for the object recognition can be chosen using \((lat, long)\) and \(d\).
CHAPTER 7. LOCATION INFORMED OBJECT RECOGNITION

7.1 Previous Approach

A related problem is presented in [27]. The approach presented by the author utilizes SIFT along with a clustering algorithm to find the region of interest. The approach assumes we are given the object or region of interest in an image $A$ and the query image $B$ in which we hope to find $A$. First, $A$ is scaled so that it is the size we would expect to find it in $B$. SIFT is applied to both images resulting in sets of keypoints and descriptors for each, $F_A, D_A$ and $F_B, D_B$. The closest descriptor in $D_B$ for each descriptor in $D_A$ is computed. For the top $N$ closest descriptor matches, we compute a rectangle of the same dimension of $A$ centered at the corresponding keypoint in $F_B$. The region which most likely is the region of interest, $A$, is the rectangle centered at a keypoint containing the largest number of matched SIFT descriptors in $B$.

7.2 Proposed Approach

We propose a similar approach that attempts to remove the limitations on rectangular shape and scale dependence. Given an image $A$ containing an object of interest and the image $B$ which we suspect contains the object of interest, apply SIFT to each image to obtain a set of keypoints and descriptors for each, $F_A, D_A$ and $F_B, D_B$. After finding matches for the descriptors of $D_A$ in $D_B$, we take the corresponding keypoints in $F_B$ of these matches and cluster these points based on their $(x, y)$-location using the algorithm DBSCAN. The largest cluster of keypoints is proposed to be the object of interest in $B$. 
7.3 Experiments and Results

To compare the two methods, an initial experiment was performed on a small dataset of 15 images taken in Waltham, MA. For each of the 15 images, a database image of the object of interest is stored. We determine the pixel location in each of the 15 images of the center of the object of interest. Then each method is used to find the object of interest and the center of the object is calculated by the methods. The proposed approach outperformed the previous approach, finding a more accurate center 10 out of the 15 times. On average the proposed approach had an error of 411.95 pixels and the previous approach had an error of 604.88 pixels. For the ten images in which the proposed approach outperformed the previous approach, the proposed approach outperformed the previous approach on average by 315.09 pixels. For the other five images, the previous approach outperformed the proposed approach on average by just 51.36 pixels.
Figure 7.1: A sample database image containing an object of interest

Figure 7.2: Image in which we will locate the object in the database image
CHAPTER 7. LOCATION INFORMED OBJECT RECOGNITION

Figure 7.3: Database object located using \textbf{previous method}. Rectangle denotes location in which the object is believed to be.

Figure 7.4: Database object located using the \textbf{proposed method}. Pink points represent where the object is believed to be.
The research presented in this thesis is a start of a larger project focused on both object recognition and the guidance system for individuals who are visually impaired. This section presents some of the work that can be done as extension of this thesis.

The results of the object recognition show that the system’s success depends largely on the segmentation of the image. If the segmentation method extracts the object of interest well, the system is able to classify objects with a high level of accuracy. However, as the effectiveness of the segmentation decreases so does the accuracy of the system. Developing a more effective segmentation method would be a good first step in future research for the project.

More experiments with our object recognition system must also be done. So far we have only compared our method to SIFT and we are planning to have a more extensive comparison with other object recognition systems from the literature. An experiment applying some of the state of the art methods in object recognition (such as those mentioned in Chapter 2) to our dataset would give us a better idea of the effectiveness of our system and the difficulty of our dataset. Similarly, an experiment applying our object recognition system to a well known, challenging dataset such
as the Visual Object Classes (VOC) Challenge should be done. It would also be interesting to replace our segmentation method with methods from the literature while maintaining the same shape classification techniques and compare the results of this modified system to the results of the original system.

In addition to these experiments, there are several modifications to the feature extraction and shape classification method that we would like to pursue. Other methods of feature extraction should be considered, in particular the use of the wavelet transform or the scale space. Our feature descriptors only provide a certain level of scale-invariance and these transforms might provide for a more descriptive representation of images.

In terms of the location-based matching, further investigation will need to be done to determine if a better method and model should be used. A system ideally would precisely find the object of interest in the user’s image. This precise location would of course allow for the calculation of a more precise geo-location of the user who took the photograph. As discussed earlier in the work, it might be the case that a precise location cannot be determined from one of the user’s photos. Additionally, it might be the case that a considerable amount of time passes without the system being able to determine the location of the user. In this case, a probabilistic model can be used to determine where the person moved for the purpose of predicting where the next photo is taken. The prediction of the location of where the next photo is necessary so that the image can be processed correctly—i.e. compared to the correct images in the Google Streetview database and searched for the the correct objects of interest.

Finally, we would want to test how well the matching performs over time. The guidance system would not just have one photograph of the object, but rather a series of images. It would be interesting to see if having several views of the same object can be leveraged to improve the recognition performance. We would need to consider
whether or not the model would need to change given this modified problem. We may also want to consider using techniques in motion estimation used in video compression to track the object between the series of images.

In this thesis, we contribute a recognition system of five objects of interest to a navigational guidance system for individuals who are visually impaired. The five objects—fire alarm call boxes, postboxes, fire hydrants, stop signs and street signs—were chosen because the geo-location (latitude and longitude) of each is known. The guidance system for which this work was designed for is a ‘vision-first’ system. In terms of hardware, it consists of smartphone with a built-in camera and GPS device. The phone is either attached to the user or held by the user and continuously takes pictures of the user’s surroundings. The phone’s GPS locator is used to estimate the user’s location and the photographs of the surroundings are used to determine a precise geo-location. An image taken by the system is processed in two ways. First, it is queried against relevant images in the Google Maps Streetview database; relevant images are those in the database taken within a certain radius of the user’s estimated location. A matching algorithm is used to find matching points between the user’s image and the database image, and projective geometry is used to refine the user’s geo-location based on the matches. Next, the image is searched for the five objects of interest. If one of the objects is located in the image, the object’s position in the image and the object’s geo-location is used to refine the position of the user.

The main contribution of this thesis is a recognition system for roadside objects. The method leverages on the combination of three feature vectors, Shape-FPFT, Edge-FPFT and Color Histogram of the image. The object recognition system performed with a high level of accuracy on our dataset. It was able to correctly classify the
five objects of interest 95% of the time when the data was perfectly segmented and 90% of the time when the data was automatically segmented. Our system performs competitively with SIFT, a well known method for object recognition. Finally, the object recognition system we contribute is fully generalizable. It recognizes shapes \textit{a priori} and was not tuned to leverage on characteristics of the roadside objects. The system can be used to recognize objects other than the ones in this work with only a change of training data.
APPENDIX A. SAMPLE DATASET IMAGES

Sample Dataset Images
APPENDIX A. SAMPLE DATASET IMAGES
APPENDIX A. SAMPLE DATASET IMAGES
Bibliography


California State University, Northridge Center on Disabilities’ 24th Annual International Technology and Persons with Disabilities Conference (), pp. 1–3.


