2012-04-10

Automated Fingertip Detection

Joseph G. Butler
Brigham Young University - Provo

Automated Fingertip Detection

Joseph Butler

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Parris Egbert, Chair
Bryan Morse
Eric Ringger

Department of Computer Science

Brigham Young University

June 2012

ABSTRACT

Automated Fingertip Detection

Joseph Butler
Department of Computer Science, BYU
Master of Science

One of the oldest biometrics that has been used to uniquely identify a person is their fingerprint. Recent developments in research on fingerprint collection have made it possible to collect fingerprint data from a stand-off digital image. Each of the techniques developed so far have relied on either a very controlled capture environment to ensure only a single fingertip is collected or manual cropping of the image down to the fingertip. The main body of the research focuses on extracting the fingerprint itself. If fingerprint collection via digital image is ever to be fielded in the real world on such devices as smart phones or tablets it will be necessary for the software to automatically detect a single or multiple fingertips in an image and isolate them for extracting the fingerprint. We introduce an automatic fingertip detection algorithm that couples image processing techniques with a machine learning capability to successfully identify varying numbers of fingertips in digital images. Our algorithm proves that while it is difficult to remove all constraints from the capture environment it is achievable with the method we have developed and we can achieve a recall of 69.77% at a precision of 78.95%. This gives us the important capability to detect varying numbers of fingertips in an image and provide a crucial piece in what could be a complete automated fingerprint recognition system.

Keywords: Fingertip detection, fingerprint capture

ACKNOWLEDGMENTS

Thanks go to Dr. Egbert for his patience and support throughout the work on this project. Additional thanks to the other committee members, Dr. Morse and Dr. Ringger for their time in reviewing my work. Finally, thanks to my wife and kids for their support and sacrificing their time with me to allow me to complete this work.

Table of Contents

List of Figures

Chapter 1

Introduction

The field of biometrics is one that has greatly benefited from the advancement of technology. Without technology certain biometric modalities would not even be possible and some of the older, more traditional modalities, such as fingerprint matching or facial recognition would not be as widely used. The application of technology to the modality of facial recognition has afforded us the capability to now automatically detect faces in images and identify these faces as the people to whom they belong. Both detection and recognition are necessary parts of the facial recognition modality [Zhao et al., 2003]. While facial detection is the easier problem of the two, automated facial recognition would not be possible without automated facial detection. The fingerprint modality has a similar need of a two-part solution to automatically detect and recognize fingerprints in digital images.

Traditionally fingerprint data was collected from an individual by the means of applying ink to the fingers of the individual and then pressing the fingers onto paper. As technology has advanced, more sophisticated fingerprint sensors have been developed to help increase the accuracy of the fingerprint data collected and make collection easier. As this process has matured, the use of fingerprints has increased and has become much more common. One can now use fingerprint sensors to open the locks on doors or provide access to personal computers. One problem with incorporating fingerprint sensors into devices is the additional cost to manufacturers. One of the cutting edge developments in fingerprint acquisition that helps alleviate this cost is the ability to extract a fingerprint from a stand-off digital image of a finger. This development turns a simple digital camera into an inexpensive and increasingly

higher-definition fingerprint sensor. Many personal computing devices such as tablets, laptops and even smart phones are attempting to provide the most capability in the smallest form factor. The ability then to use a digital camera, with which many of these devices are already equipped, as a multi-use device that can also act as the fingerprint sensor is an attractive proposition.

The current ability of capturing fingerprints from digital images has its limitations. These include the need to have either a controlled capture environment where the lighting and focus distance are uniform for all images captured or the need to manually crop the image collected down to the individual fingertip of interest. These restrictions, while they do not take away from the significant achievement of extracting the fingerprint data from the image, do leave the system unfinished. Just as in a facial recognition system, the ability to match a fingerprint to its owner is not enough. A complete system must also be able to automatically detect the location of the fingertip in the image so that it can then be matched to its owner.

Some methods have been developed in an effort to pre-process a digital image of a finger and prepare it for fingerprint extraction. However, the methods developed to date all rely on either placing constraints in the capture environment to control some of the variables such as lighting, background, or focal distance or they rely on the manual step of cropping the image. We present an automated fingertip detection algorithm that is free of the constraints of a controlled environment and free of the need for manual cropping. Through the combination of skin color pixel masking, high frequency texture masking, friction ridge orientation estimation and Support Vector Machine (SVM) classification our algorithm decides which regions of the image are classified as fingertips. This provides fingerprint capture and storage that is much more robust than earlier techniques.

Chapter 2

Related Work

In a color image of a finger one of the most recognizable features of the finger is the skin tone. Lee et al. [2005] developed a method of determining the skin colored pixels in an image by converting the RGB values into the YUV color space and comparing the chrominance values of each pixel to a distribution of skin color taken from several images of skin across the different races. The values were then used to generate a binary mask of the image. In addition to distinguishing the skin colored pixels Lee et al. [2005] use a Discrete Wavelet Transform to determine the areas of high frequency in the image with the reasoning being that the finger is going to be closer to the camera than the background, and should be in focus. In additional research Lee et al. [2006] developed an algorithm that calculates the recognizable portions of the image using three main factors of focus, quality and pose to determine which regions are recognizable.

Hiew et al. [2007, 2006] constructed a box in which they can control the lighting and focus distance of the image captured by having the subject place a single finger through a hole in the box. Given the constant background color, the isolation of the skin colored pixels is trivial. The authors then use adaptive thresholding to isolate the pixels that are in focus enough to be used in case the image is of poor enough quality that portions of the finger are not usable. The image resulting from the adaptive thresholding and normalization they perform is then cropped to remove all of the background and edges of the fingertip. A short-time Fourier transform is then used to enhance the image and produce more contrast between the friction ridges and valleys. Finally they developed a Gabor filter based feature

extractor to detect the core points of the fingerprint. Given the strictly controlled capture environment they require through the use of their capture apparatus, the method used in their research is not robust enough for our application.

Yu et al. [2009] developed an algorithm for identifying the fingertips in an image of a hand. Their approach was to first convert the image of the hand to a binary image and employ an edge detection algorithm to locate the edges of the hand. They then used their border tracing algorithm to find the peaks of the fingertips and the valleys between them. Using these peak and valley locations they were able to segment the image and crop out entire fingers. The authors employed the morphological operation known as opening to accent the noise in the first knuckle of the finger making it easier to detect. They then cropped the image at the first knuckle leaving only the fingertip. It was necessary in their work to restrict the capture environment by controlling the background against which the images were taken and requiring the fingers to be extended providing adequate space between each finger for their algorithm to identify both the peaks and valleys of the border of the hand. While the solution they provided did allow them to locate the fingertips in the image it was still necessary to control the capture environment leaving the problem of uncontrolled robust capture unsolved.

The friction ridges of the fingertip are its identifying feature and are crucial to distinguishing between the actual fingertip versus other possible high-texture skin colored regions of an image. Work done by Hong et al. [1998] developed an algorithm to estimate the orientation and frequency of friction ridges in an image of a fingerprint. Orientation estimation was done by taking the least mean square of the gradient value at each pixel within the individual blocks of the image. The result of this calculation gave them the direction orthogonal to the dominant direction of the Fourier spectrum of each block. The authors noted that noise in the image could lead to incorrect orientation estimation and introduced the solution of low-pass filtering the image prior to orientation estimation. Gray-scale images of fingerprints are more susceptible to noise in the image than are color images of fingers

making low-pass filtering unnecessary in our algorithm. Hong et al. also introduced an algorithm for calculating the frequency of the friction ridges. Using the orientation of the blocks in the image they compute a window of a set size and orient the window to match the orientation of the image. They then computed what they called the x-signature of the window, which would form a discrete sinusoidal-shape wave, the frequency of which was determined to be the frequency of the ridges. Working with images of fingerprints that were all at the resolution of 500 dpi the authors knew that the frequency of the ridges should fall within a certain range. If the frequency of a certain block fell outside of this range it was marked as invalid and the authors used interpolation between the neighboring blocks' frequencies to set the frequency value for the invalid block.

Wang and Wang [2004] introduced a method of enhancing a fingerprint image in its singular point region. The singular point region of the image is the region where the ridge curvature is higher than normal and where the direction of the ridges changes rapidly. To enhance the singular point region the authors first needed to identify this region of the image. To do so they employed a method developed by Jain et al. [1997]. This method divided the image into equal blocks of a set size and calculated the gradient average of each block. After the orientation of each block was calculated, Wang and Wang presented the algorithm for calculating the Poincare index value of each block. Through their work they determined that singular point regions would have Poincare values of either $-\frac{1}{2}$ or $\frac{1}{2}$. Once the singular point regions were detected they then designed a band-pass filter to enhance the region and suppress possible noise.

Van and Le [2009] developed an algorithm that could consistently detect the reference point in fingerprint images for all types of fingerprints. The algorithm they developed employs the gradient average of sub-blocks within the image. After calculating the gradient average for each block the algorithm then employs a method of adaptive neighborhood smoothing to smooth the orientation values calculated and reduce the effects of noise that could have been introduced by scars or creases in the fingertip. The orientation is then used to calculate

what they term the *convex orientation consistency field*, allowing them to locate the reference point in the image. This orientation consistency field is a representation of the relationship between the orientation of each block and the orientations of the neighboring blocks. This is similar to the Poincare index value used by Wang and Wang [2004] in that in both cases the authors are trying to use this relationship of the orientation between neighboring blocks. Van and Le also compute the convex curvature of each block using a unique algorithm that they developed. Through combining the orientation consistency field and convex curvature of each block the authors are able to locate the reference point of the fingertip. The images used in this project were not digital images of fingers but instead actual fingerprint images from a fingerprint database.

Working with fingerprint images obtained from touch sensors, Qi and Xie [2008] developed a method to detect and remove background information from the image. The authors split the image into blocks and calculate the average gradient magnitude and the direction variance of each block. They use this information to give a block quality score to each block. After setting a threshold for suitable block quality they classify all of the blocks in the image as either background or foreground, then remove all of the blocks classified as background. The algorithm they produced from their research will quickly and efficiently remove background blocks from images of fingerprints obtained from touch sensors and give an overall image quality score. This preprocessing they performed was proposed as a solution for increasing the effectiveness of a fingerprint recognition system by enhancing the value of the image sent to the system.

Another solution to the problem of removing unwanted background information from images of fingerprints obtained from touch sensors was presented by Weixin et al. [2009]. They build upon the active contouring algorithm which uses a snake or an energy-minimizing spline. The snake is guided by external constraint energies and influenced by image energies to pull it to salient features of the image such as the edges of the fingerprint. To increase the performance of the active contouring the authors first normalize the image to smooth

out the gray values along the ridges and valleys. They then divide the image into blocks and calculate the gradient average for each block. They discretize the orientation values into eight different possibilities and then compute a directional histogram with the discretized orientations of all the blocks in the image. The active contouring algorithm is then fed orientation-variance information and histogram-peak differences for neighborhoods of the image. This method works well for fingerprint images that are collected via touch sensor since the only high frequency areas in the image were in the foreground and the contrast between the edge of the fingerprint and the background was well defined. Attempting to apply this method to segmenting out a fingertip from a digital color image of a hand in a robust capture environment such as we are presenting would present too difficult of a challenge given that the background might very well contain information that would confuse the active contouring algorithm.

Song et al. [2004] developed a new system of touch-less fingerprint recognition. Their system involved both a hardware apparatus for capturing the images as well as image-processing techniques to enhance the image and prepare it for fingerprint extraction. The hardware apparatus used a digital camera to provide touch-less capture. The camera was built into a platform onto which the subject would place their finger. The tip of the finger where the fingerprint was being captured from was not touching the sensor but the joint just below the fingertip was resting on the platform to allow a controlled focal distance in all of the images captured. The apparatus they constructed was also able to somewhat control the lighting and the background. A blue light was placed in the apparatus that shone on the fingertip during capture. This blue light helped reduce the effects of the other lighting in the image. To further help reduce the effects of other lighting on the image a blue filter was added to the camera. As well as controlling the lighting, the apparatus also controlled the background by using an opening only large enough for the camera to capture the fingertip itself without any background. Once the image was captured the image was converted to gray-scale, then Gaussian smoothing was performed and the image was then split into blocks.

The gradient average for each block was computed and used to construct block-specific Gabor filters to further smooth out noise and enhance the contrast between the ridges and valleys.

All of the work done to allow the collection of fingerprint data via digital images has been restricted to images of single fingertips or images that were collected in a controlled environment. Thus the need to make capture more realistic in an uncontrolled environment via a simple digital camera without the need for any additional sensors or capture apparatuses still exists. To solve this problem it will be necessary to automatically detect fingertips in an image where multiple fingertips could be present and where the collection environment is uncontrolled. This paper describes our solution to that problem.

Chapter 3

Automated Fingertip Detection Solution

Uncontrolled capture of images with varying numbers of fingers in the image, varying lighting situations, varied backgrounds and focal distances is a difficult problem. Each control that we eliminate from the capture process must be accounted for. Our solution to this problem uses various methods developed in the previous work as well as our own novel techniques.

The first step in the process is to determine what portion of the image contains the hand and ignore the rest. In a color image we can assume that the likelihood of skin tones in the background is going to be low. This assumption led us to adopt the method of skin color masking as used by Lee et al. [2005]. While the likelihood of skin tones in the background may be low we did not want to rely on it being impossible. In order to allow for possible skin color masks that fail to occlude all of the background we look to another significant feature of the image, texture.

If an image is going to be usable then the hand, and more specifically the fingertips, will need to be in focus. For the hand to be in focus and fingerprint detail to be discernible, the focal distance will have to be relatively short, thus making anything behind the hand out of focus. We utilize this feature of texture to create a texture mask similar to the method used by Lee et al. [2005]. Through the combination of both the color mask and the texture mask we are able to reduce the remainder of the calculation to the portion of the image that has been determined to include the fingers.

We then utilize the most distinguishing feature of the fingertip, the friction ridges, as the basis for further processing. Many of the previous works employ a method of orientation estimation to calculate the orientation of the friction ridges within each sub-block of the image. Our solution employs the method of gradient average to estimate the ridge orientation. We enhance orientation estimation in our solution through the use of an iterative process of orientation estimation and ridge frequency estimation. In a method similar to Hong et al. [1998] we use the orientation value to help us calculate ridge frequency. Hong et al. calculate a window around each block that is rotated to match the orientation value of that block and attempt to find a sinusoidal-shaped wave that fits the alternating intensity values of the ridges and valleys. The frequency of the wave gives them the ridge frequency near that block. To enhance the accuracy of our ridge frequency estimation we only calculate the ridge frequency of the blocks in which none of the pixels are masked. Iterating through all of the unmasked blocks we first calculate the threshold between the ridges and valleys based on the average intensity values of the ridges and valleys. We then interpolate across each block perpendicular to the orientation value for that block. As we interpolate across the block we count the width of the ridges by the number of pixels that are higher than the average threshold. This ridge width calculation is then used to adjust the size of the blocks and re-calculate the orientation information.

As shown in previous work [Van and Le, 2009] and [Wang and Wang, 2004], the orientation estimation value can be used to determine a relationship between neighboring blocks. Our solution utilizes the Poincare index value as presented by Wang and Wang to represent this relationship. However, instead of using the Poincare index as the deciding factor, we use this information as input to a SVM to perform classification. The possibility of more than just fingertips in the images we use means we can't rely on simply the Poincare value. We need something more to help us decide which regions of the image were indeed fingertips.

The final step to our solution is to use all of the information gathered so far to decide which regions of the image are fingertips and to isolate these regions for input into a fingerprint extraction system. To do this we developed our connected-neighbor algorithm, which finds the core regions of the fingertips based on the number of connected blocks which have been classified by the SVM as core blocks. These regions are determined to be the core of the fingertip and a dynamic region around each core is set based on the average ridge width. Each fingertip region is then saved as a single image and is ready for fingertip extraction. Each step in our solution will be presented in detail in the following chapters.

Our research builds upon the previous work by adapting and enhancing the methods used to identify skin color and high texture information. We enhance a commonly used approach to estimate friction ridge orientation and add our unique method of calculating ridge width to provide a more accurate orientation estimation. We take the information given us by the calculation of the Poincare value and build upon it through the use of the SVM to allow a decision to automatically be made by the software. This automation is made all the more unique by removing the controls in the capture environment. While the end goal of all of the previous work was to extract the fingerprint from a single fingertip in the image, our goal is to provide a solution to the unsolved problem of automating the detection of varying numbers of fingertips that may be present in an image. Just as facial recognition would not be what it is today without automated facial detection, we are confident that our novel solution to the problem of automated fingertip detection will move fingerprint recognition via digital image further along its path of development.

Chapter 4

Color and Texture Masking

To allow a much more robust method of collecting fingerprint data via digital images we have developed the automatic fingertip detection algorithm. This algorithm builds on previous work and adds the decision-making capability of the support vector classification. This provides the freedom from constraint that the previous methods have had to operate under such as controlled capture environments and only capturing a single finger at a time. The first step in our algorithm is to produce two masks that will be used in the fingerprint detection phase. The first is a color mask which allows us to determine portions of the image that are candidate skin regions. The second is a texture mask that will be used to determine actual fingerprints. These masks allow us to isolate the regions of the image that are both skin color and high frequency texture areas.

Our skin color determination algorithm is based on the method used by Lee et al. [2005] but we have extended it to produce more accurate results. Instead of converting to a normalized RGB color space we convert the image to $Y'UV$ color space. The Y' refers to the intensity value of each pixel or, in other words, the gray-scale value of the pixel. The U and V are different chrominance values that represent the color in the image. To convert from RGB to $Y'UV$ we use

$$Y' = W_R R + W_G G + W_B B \tag{4.1}$$

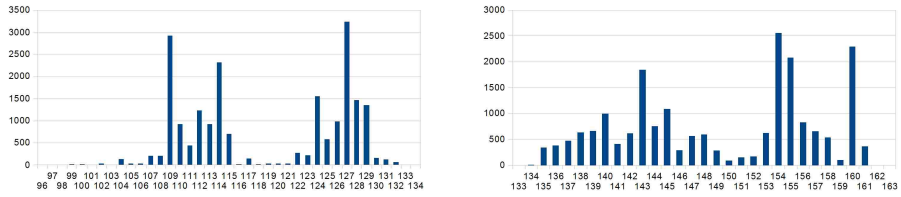
$$U = U_{\max} \frac{B - Y'}{1 - W_B} \tag{4.2}$$

$$V = V_{\max} \frac{R - Y'}{1 - W_R} \tag{4.3}$$

where

$$\begin{aligned} W_R &= 0.299 \\ W_B &= 0.114 \\ W_G &= 1 - W_R - W_B \\ U_{\max} &= 0.436 \\ V_{\max} &= 0.615 \end{aligned} \tag{4.4}$$

Once converted, the pixels are compared to a skin-color distribution. To build the skin-color distribution we gathered skin-patch images from each of the different races using various image search websites. Each skin-patch image was collected from the palm or fingers to provide a tighter distribution. We converted the skin-patch images to the $Y'UV$ color space and created a histogram of the U and V values. The Y' value is not used since it only represents the intensity or gray-scale value at each pixel. These histograms are shown in Figure 4.1.



(a) U distribution histogram (b) V distribution histogram

Figure 4.1: The distribution of the U and V values taken from the skin color patches in our collection

The skin-colored pixels in our collection were distributed in a bimodal Gaussian curve. Recognizing this we knew we could employ the optimal thresholding technique to discover the threshold between the two curves. To do so we first need to calculate the probability that a given U or V value would fall into each curve. Once we have determined the probabilities we can then use the following to find the optimal threshold T between the two distributions:

$$AT^2 + BT + C = 0 \tag{4.5}$$

where

$$A = \sigma_1^2 - \sigma_2^2 \tag{4.6}$$

$$B = 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2) \tag{4.7}$$

$$C = \sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + 2\sigma_1^2\sigma_2^2 ln(\sigma_2 P_1/\sigma_1 P_2) \tag{4.8}$$

and μ_1 and σ_1 are the mean and standard deviation of the lower curve, μ_2 and σ_2 are the mean and standard deviation of the upper curve and P_1 and P_2 are the probabilities that each U and V value will fall into the lower and upper curves respectively. In solving for T in Eq. 4.5 there is a possibility of obtaining two possible solutions between the means of both distributions. If such is the case then two thresholds would need to be used. In our case there was only one solution for T between the means of both distributions leaving us with the optimal threshold.

Using this optimal thresholding technique we are able to first classify a pixel value as falling into either the upper or lower distribution. Then through empirical testing we determined that the upper curve of both the U and V distributions are more likely to contain the true skin color values whereas the lower curves are more likely to contain the color values that are impostor skin colored values. Impostor skin colors can come from items in the image such as jewelry being worn that contain certain shades of gold, orange, red or yellow. Our testing showed that the U and V values within three standard deviations of the mean of the upper curve and within one standard deviation of the lower curve were generally the actual skin colored pixels. Given that it is possible to have skin tones in the background, the color mask alone is not sufficient for removing all unwanted information from the image.

An image taken of a hand and focused on the fingers will have a short enough depth of field that the fingers will be in focus but the background will be out of focus. This means

that we only want to center our attention on the area of the image that is in focus. This will help us remove any unwanted background information that may have passed the color mask but is not actually part of the fingers or hand. To allow us to concentrate only on the regions of the image that are in focus we determine the high frequency areas of the image through the use of a Discrete Wavelet Transform. Specifically we use a Haar wavelet to extract the texture information from the image. Running the image through a two-dimensional Haar wavelet transform will divide the image into four subregions. The lower right subregion of the image will contain the lowest frequency data, the lower left will contain mostly the horizontal edges, the upper right will contain mostly the vertical edges and the upper left subregion will contain the highest frequency information from the image. To ensure that we are only getting high frequency information for the mask we perform two iterations of the Haar transformation on our input image. From the subregion of the resulting image that contains the highest frequency change, a texture mask is generated and scaled back to the original size of the image.

The resulting color and texture masks are then both passed through a dilation and erosion process to smooth the edges and close any holes. The masks are then combined with more weight given to the texture mask to account for the possible background color that passes as skin color. This combined mask is then applied to the image.

Chapter 5

Orientation Estimation and Poincare Index

Once we have isolated the usable regions of the image we then need to determine what is fingertip and what is not. To do this we will leverage the friction ridges of the finger. One of the most common and effective methods for estimating the orientation of the friction ridges is by calculating the gradient average of a sub-block of the image as done in [Wang and Wang, 2004], [Van and Le, 2009] and [Weixin et al., 2009]. While other methods of estimating the orientation of each block were presented by [Hong et al., 1998] and [Qi and Xie, 2008], we elected to use the gradient average method as it has not only proven itself to be effective but we also wanted to extend it using our two step process. The first step of our process is to divide the image into sub-blocks of default size (16 pixels) and calculate the gradient average for each block with the following equation:

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{\sum_{i=1}^{W}\sum_{j=1}^{W} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{W}\sum_{j=1}^{W}(G_x^2(i,j) - G_y^2(i,J))}\right) \tag{5.1}$$

where W is the size of the block, G_x is the gradient in the x direction at the location i, j and G_y is the gradient in the y direction at the location i, j [Wang and Wang, 2004]. Looping over the pixels in a block of size W we calculate the gradient at each pixel and the resulting θ is the gradient average of the entire block or the direction of greatest change within the block.

The friction ridge orientation is perpendicular to the gradient average of each block as seen in Figure 5.1. This first calculation of the orientation estimation with the blocks set to a default size may have been adequate in cases where the input images in which the fingers were all very similar in size. We found however, that it is not as accurate when the sizes of the fingers in the image can vary greatly such as in our uncontrolled capture environment.

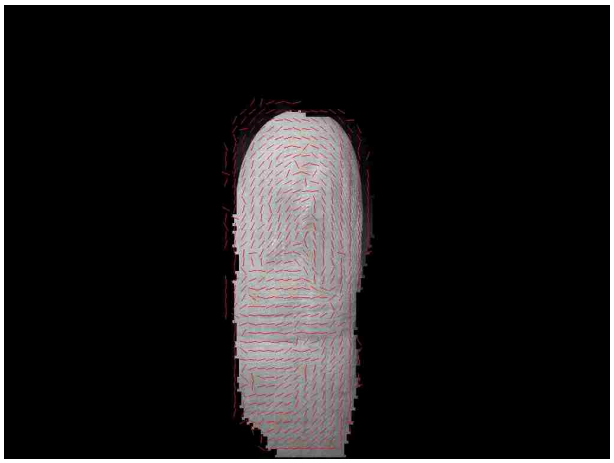


Figure 5.1: Friction ridge orientations. The red lines on the image indicate the friction ridge orientation which is perpendicular to the gradient average as calculated by our algorithm.

To enhance the process we determined that the size of the block used should vary depending on the size of the finger which is the second step in our process. We determine the size of the finger by measuring the average width of the friction ridges.

After having obtained the orientation of each block of default size, we then loop through the unmasked blocks of the image and find the upper and lower intensity values in each block. The upper intensity value represents the ridges in the image and the lower represents the valleys. The threshold of each block is half the difference between the two values. The average of all the thresholds will be used to calculate the ridge width. After determining the threshold, we then loop through the image and extrapolate across each block perpendicular to the block's orientation. As we extrapolate we determine the ridge width based on the number of continuous pixels that are above the average intensity threshold. To counteract the possibility of noise such as creases or scars we use the average ridge width of all the blocks. Once the average ridge width is set we resize the sub-blocks of the image to contain two ridges and valleys per block and re-estimate the orientation.

After the final orientation estimation is complete we can now look at the relationship each block has with its neighbors. This relationship is important because it helps us determine when the orientation values for the neighboring blocks are all similar, such as we would see in the edges of the fingertip or the regions of the finger below the first knuckle. More

importantly, this relationship also helps us determine when the orientation values differ in specific ways, such as at the singular points of the image. To provide information on this relationship we calculate the Poincare index values for each block. Using the orientation value from each block and its eight connected neighboring blocks we calculate the Poincare index as follows:

$$Poincare(i,j) = \left(\frac{1}{2\pi}\right) \sum_{k=0}^{N-1} \Delta(k) \tag{5.2}$$

$$\Delta(k) = \begin{cases} \delta(k) & |\delta(k)| < \frac{\pi}{2} \\ \pi + \delta(k) & \delta(k) < \frac{-\pi}{2} \\ \pi - \delta(k) & otherwise \end{cases} \tag{5.3}$$

$$\delta(k) = \theta(X(k'), Y(k')) - \theta(X(k), Y(k)) \tag{5.4}$$

$$k' = (k+1) mod(N) \tag{5.5}$$

where $\theta(i,j)$ is the orientation for the block at i,j, $X(k), Y(k)$ are the coordinates of the blocks that are in a closed loop around block i,j, and the total number of blocks in the loop is N [Wang and Wang, 2004]. The resulting Poincare value for each block gives us a measure of the center block's orientation and how it compares to the orientations of its neighbors. If all of the neighboring orientations are quite similar, such as is the case at the outer regions of the fingertip or below the knuckle of the finger, then the Poincare index value will reflect this. On the other hand, if there is a significant difference in the orientation between the neighbors, like we would see at the singular point regions of the fingerprint, this will also be reflected in the Poincare index value. Wang and Wang reported in their research that Poincare index values of $\frac{1}{2}$ and $-\frac{1}{2}$ represent the key features, known as the core and delta, of the singular point region. Our solution does not use the Poincare index value as the sole

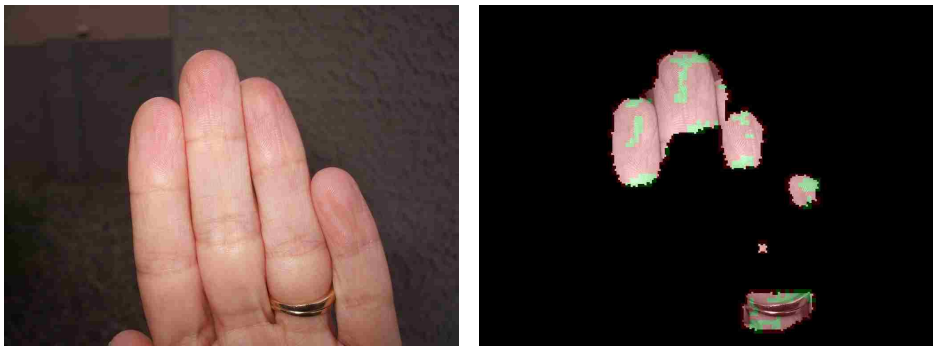
deciding factor on the location of the fingertip in the image but we utilize the information it gives us about the relationships between neighboring blocks as input to the next step in the process, support vector classification.

Chapter 6

Support Vector Classification, Connected Neighbors and Automated Cropping

With all of the information that we have gathered on the image so far we can now make a decision on where the fingertip is in the image. By analyzing the information we have calculated in the Poincare index value for a block and its neighbors we can attempt to classify blocks into different classes. For our solution we chose to divide the blocks into two classes, core and non-core. The core blocks of the image are the blocks in which the core features of the fingerprint, such as the loop, arch, whirl or delta, are located. Dividing the blocks into these two classes makes sense because the core of a fingerprint is the most distinguishing characteristic of the fingerprint and it is a pattern unique to the fingertip that will not be found among the other friction ridges on the finger or hand. The tool that we chose to perform our classification is the support vector machine (SVM) as developed by Chang et al. [2011]. The SVM is a powerful classification tool that is also simple to understand conceptually. Given a set of data that could fall into two different classes, the SVM will attempt to find a hyper-plane that would best define the boundary between the two classes.

Support vector classification requires that the SVM be trained with examples of the classes into which the data will be classified. We gather training data for the SVM by selecting example images and visually determining the core blocks and non-core blocks. From each block we construct a feature vector which consists of the Poincare value for that block and its eight connected neighbors. These feature vectors are then given a class number and the data is formatted and scaled for the SVM training function. After training the SVM the



(a) Image with four fingers (b) Core blocks identified in image

Figure 6.1: Core block classification: (a) Test image from which we expect to detect four fingertips. (b) The resulting image with both the color and texture masks applied and the blocks classified as core blocks highlighted in green and those classified as non-core highlighted in red.

best fitting plane was found that defined the boundary between our training set data. This best fitting plane is known as the *hard boundary* between the training set data. To allow us to effectively perform classification it is also necessary to define the soft boundary between the data. The *soft boundary* is defined as $C > 0$ were C is the magnitude of the vector orthogonal to the best fitting plane. The SVM library provided by Chang et al. includes a tool to perform cross-validation of the C parameter used for the model. Upon performing this cross validation the tool gave us the value of $C = 0.5$. This parameter helps define the margin between the class boundary and the feature vectors in either class. This value of C provides the most accurate classification using a linear kernel. The kernel determines what method will be used to define the class boundary, in our case a linear plane.

Once all of the blocks in the image have been classified as core and non-core we need to be able to discern between erroneous classifications and the true core regions of the image. To do this we developed the connected neighbors search. Through examination of the resulting classifications we found that there were several seemingly anomalous blocks that were determined to be core blocks as seen in Figure 6.1b. Through empirical results we noted that true core regions of the image would have at least four or more neighboring blocks classified as core blocks. Using a recursive counting algorithm we are able to count

the number of connected neighbors each core block has. Those blocks with a count above the threshold are determined to be the fingertip regions of the image.

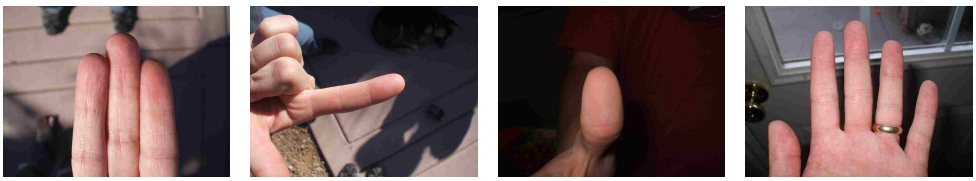
As stated, the purpose of the fingertip detection algorithm is to automate the process of detecting and cropping out varying numbers of fingertips in an image. In order to make this a more robust solution we also developed the capability to automatically determine the orientation of the overall image and rotate it so that the fingertips were at the top of the image. To do so we determine the center of mass of the color mask and rotate that to be the bottom of the image. This simplistic solution to rotation is sufficient since the center of mass of the color mask will either include the palm of the hand or the majority of the finger below the tip. This auto rotation is important in that it rotates the test data to the same orientation as the training data therefore providing more accurate classification.

The auto rotation of the image also makes the cropping of the fingertip regions easier to perform since a singular rectangular proportion can be used for cropping. Using the connected neighbors search we are able to automatically locate the core regions of the fingertip. After the core region is located a rectangular portion of the image around that region is saved off as the single fingertip image ready to be used to extract the fingerprint data. The blocks within the previously detected fingertip regions are skipped in the remaining connected neighbors search to reduce the possibility of erroneously detecting multiple core regions on a single finger.

Chapter 7

Results

In order to accomplish the goals of this research, our automated fingertip detection algorithm must accomplish two things. First the algorithm must successfully detect fingertips in an image. Second, it must be able to do so in images captured in uncontrolled environments that contain varying numbers of fingers. To test the success of these two goals we set up a two-prong image collection approach. First we wanted to use a simple off-the-shelf point-and-shoot camera to prove that high enough quality images could come from such simple equipment. With this camera we took images in both indoor and outdoor environments with different lighting. We also collected images with the hand in various orientations as well as with varying numbers of fingers present. Examples of images from our collection can be seen in Figure 7.1. The most common number of fingers in the image was four since it is easiest for the subject to hold their hand out facing up with all of the fingers extended. However, we tested images of various numbers of fingers, images with some fingers bent or occluded, and so forth. The image in Figure 6.1 was collected outdoors in the evening with the flash active on the camera. We expected this image to be easy for our algorithm since the detail in



(a) Ours 1 (b) Ours 2 (c) Ours 3 (d) Ours 4

Figure 7.1: Web collection samples. Examples of the images we gathered with our 7 megapixel consumer camera

(a) Index finger (b) Middle finger (c) Ring finger (d) Pinky finger

Figure 7.2: Resulting fingertip images. Fingertip images extracted by our algorithm from the image in Figure 6.1

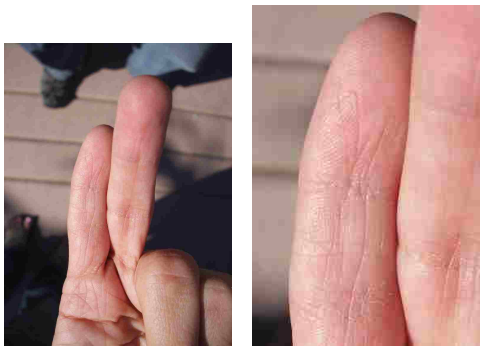
the fingerprints is easy to see. Our algorithm successfully identified four core regions in the image and extracted the images seen in Figure 7.2. An example of an image we expected our algorithm to struggle with can be seen in Figure 7.4. The image in this figure shows that one of the fingers, the middle finger, is not in focus enough to visually determine the fingerprint. Our algorithm was able to successfully identify the index finger but not the middle finger. Finally, an example of an image we expected to be hard for our algorithm to process because of the extreme lighting and hard shadow can be seen in Figure 7.3. Our algorithm was able to identify the index finger but given the difficulty with the lighting it identified the incorrect region of the middle finger.

The second goal of this research was to have our fingertip detection algorithm work in an uncontrolled environment i.e. under any lighting conditions, with any background, with the hand at any orientation etc. The images that we took ourselves could be seen to have an element of control in them because we knew what kind of image we wanted and were trying to get the best images we could even though we were not using any controlled lighting or background or a capture apparatus. To eliminate even the appearance of control we determined the best approach would be to gather images from various image search websites. This would reduce the appearance of control because those who took the pictures did not do so with the intent to send it through our algorithm. That being said, we still had to settle on

(a) Harsh lighting (b) Index finger (c) Middle finger

Figure 7.3: Harsh lighting example: (a) Image collected outdoors in full afternoon sun with hard shadow (b) the index finger as identified by our algorithm (c) the portion of the middle finger not in the hard shadow. The algorithm failed to capture the correct core region of the image

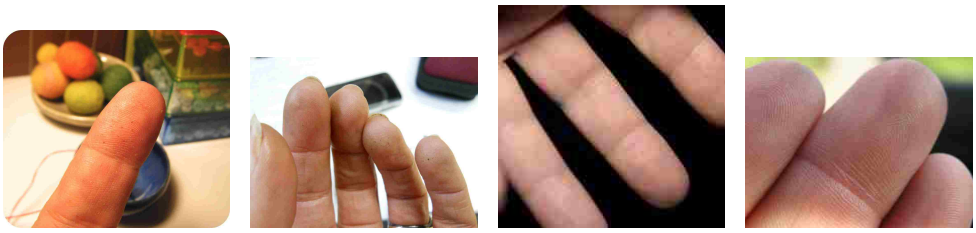
some selection criteria for the images that we would be searching for. Given that we were attempting to find images that could be used to ultimately extract fingerprint information from, the selection criteria we settled on were that the fingertips had to contain humanly discernible fingerprint data and, if possible, more than one finger.



(a) Two Fingers (b) Index finger

Figure 7.4: Out of focus example: (a) two fingers in full sun, one in focus the other not (b) the index finger passed the texture mask and was detected successfully

Collecting adequate images via image search results on the internet proved to be more difficult than we had originally thought. The requirement that the details of the fingerprint be visible disqualified a large number of the hand images that we saw. The images in Figure



(a) Web 1 (b) Web 2 (c) Web 3 (d) Web 4

Figure 7.5: Our collection samples. Examples of the images we gathered from various internet image searches

(a) Hand (b) Index finger (c) Middle finger (d) Ring finger

Figure 7.6: Four finger example from web: (a) an image from our web collection (b) the index finger, (c) the middle finger and (d) the ring finger extracted by our algorithm. The little finger was not identified by our algorithm

7.5 and Figure 7.6 are examples of the images that we were able to collect from our web search.

Of the collection that we were able to gather from the internet our algorithm achieved a recall of 78.57% and a precision of 68.75%. Of the images that we collected ourselves, our algorithm had a recall of 69.77% with a precision of 78.95%. As can be seen by the low percentage of precision, the problem of robust fingertip detection is very difficult to solve. Without control in the environment there are many variables that can hinder a positive detection and reduce the precision. Our automatic fingertip detection algorithm, while not yet perfect, is able to detect varying numbers of fingertips from images captured in real-world uncontrolled environments. It lays the foundation for moving fingerprint capture via digital image into the real world on such devices as smart phones and tablets.

One technique we thought about using in order to perform validation testing of our technique was to compare our technique head-to-head with other techniques. However, a meaningful comparison of the performance of our solution to that of other systems was not possible. Because the main goal of our system was different from the main goals of the systems developed by other researchers. The main goals of the previous systems varied from providing unique capture apparatuses for capturing digital images of fingerprints to providing the possibility of extracting fingerprint data from digital images to enhancing the extraction of fingerprint data. The main goal of our research was to automatically detect varying numbers of fingertips in digital images captured in an uncontrolled environment. The performance metric commonly used in fingerprint extractions systems is that of a positive recognition rate, or how many fingerprints were matched to their respective owners. For our performance metric however, we decided to use the common classification metric of precision versus recall. The difference in metrics invalidates any comparison that might be done.

Chapter 8

Future Work

Our novel solution to automatic fingertip detection builds on the previous work done and pushes the technology forward through the use of new methods. Our enhanced skin color masking technique provides more accurate color masking of the image. The combination of the color mask and texture mask allows us to handle images collected in uncontrolled environments with varied lighting and backgrounds. Our novel two-step approach to friction ridge orientation estimation provides accurate orientation estimation on images with fingers of varying size. The use of the Poincare index as input to the SVM classification algorithm allows us to leverage the information on the relationships between orientation values for the different blocks to correctly classify the core blocks of the fingertip. Using all of this information in connection with our auto-rotate and connected neighbors search we are able to successfully automate the detection of varying numbers of fingertip regions in an image collected in an uncontrolled environment. This is the crucial first step in taking automated fingertip recognition in digital images into a real world environment.

Given the difficulty of the problem of uncontrolled fingerprint capture via digital camera and the necessity to use a collection of complicated processes to solve it, there are ways to increase the accuracy and performance of our solution. One of the greatest hindrances to automatically detecting fingertips in a color image of varying numbers of fingers is the lack of stark contrast between the ridges and valleys such as is present in an image of a fingerprint captured via a typical fingerprint sensor. Given this lack of stark contrast, accurate estimation of friction ridge orientation is difficult. Application of tools used for

actual fingerprint extraction such as a Gabor filter to enhance the detail of the ridges and valleys in the fingertip may help improve fingertip detection even further.

Additionally, it may be possible to increase the accuracy of the friction ridge orientation estimation by taking steps to attenuate noise in the image before estimating the orientation. As we found in our research and in the research we studied the orientation values are critical to the calculation of the Poincare index and the rest of the process. Taking steps then to increase the accuracy of the ridge orientation may further improve our accuracy.

Another area in which further work might be able to increase the performance and/or accuracy of automated fingertip detection would be to use the skin color estimation capability we developed to perform active contouring as presented by Weixin et al. [2009]. One of the shortcomings of using a skin color mask the way we have done in our solution is that, when coupled with extreme lighting conditions that can occur in an uncontrolled capture environment, portions of the skin within the hand can be masked because of the intensity of the light reflecting off of the skin. Active contouring might be able to address this problem by determining the outside border of the hand. Active contouring would still need to be used in conjunction with the texture masking as we have developed it to allow the regions of the hand that would be out of focus in the image to be excluded from consideration.

References

C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, pages 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm.

B.Y. Hiew, A.B.J. Teoh, and D.C.L. Ngo. Automatic digital camera based fingerprint image preprocessing. In *Proceedings of the IEEE International Conference on Computer Graphics, Imaging and Visualization*, pages 182–189, 2006.

B.Y. Hiew, A.B.J. Teoh, and Y.H. Pang. Digital camera based fingerprint recognition. In *Proceedings of IEEE International Conference on Telecommunications and Malaysia International Conference on Communications*, pages 676–681, 2007.

L. Hong, Y. Wan, and A. Jain. Fingerprint image enhancement: algorithm and performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):777 –789, 1998.

A. Jain, L. Hong, and R. Bolle. On-line fingerprint verification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:302–314, 1997.

C. Lee, S. Lee, J. Kim, and S.J. Kim. Preprocessing of a fingerprint image captured with a mobile camera. In *Proceedings of International Conference on Advances in Biometrics*, pages 348–355, 2006.

S. Lee, C. Lee, and J. Kim. Model-based quality estimation of fingerprint images. In *Advances in Biometrics*, volume 3832 of *Lecture Notes in Computer Science*, pages 229–235. Springer Berlin / Heidelberg, 2005.

J. Qi and M. Xie. Segmentation of fingerprint images using the gradient vector field. In *Proceedings of IEEE Conference on Cybernetics and Intelligent Systems*, pages 543 –545, 2008.

Y. Song, C. Lee, and J. Kim. A new scheme for touchless fingerprint recognition system. In *Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems*, pages 524 – 527, 2004.

T.H. Van and H.T. Le. An efficient algorithm for fingerprint reference-point detection. In *Proceedings of International Conference on Computing and Communication Technologies*, pages 1 –7, 2009.

S. Wang and Y. Wang. Fingerprint enhancement in the singular point area. *IEEE Signal Processing Letters*, 11(1):16 – 19, 2004.

B. Weixin, X. Deqin, and Z. Yi-wei. Fingerprint segmentation based on improved active contour. In *Proceedings of International Conference on Networking and Digital Society*, volume 2, pages 44 –47, may 2009.

P. Yu, D. Xu, H. Li, and H. Zhou. Fingerprint image preprocessing based on whole-hand image captured by digital camera. In *Proceedings of International Conference on Computational Intelligence and Software Engineering*, pages 1 –4, 2009.

W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35:399–458, 2003.