DESIGN AND CONSTRUCTION OF A SECURE ID-CARD SYSTEM USING ROBUST IMAGE HASHING

by

Mehmet ÖZTEMEL

B.S., in Electrical and Electronics Engineering, Bilkent University, 2006

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in FBE Program for which the Thesis is Submitted
Boğaziçi University
2009

ACKNOWLEDGEMENTS

First of all, I owe special thanks to my supervisor M. Kıvanç Mıhçak, for his invaluable guidance, understanding and support. I have always admired his personality, intelligence and being such hard-working.

I am grateful to Assist. Prof. Murat Saraçlar, for his valuable ideas. It is a great chance to work with such professors with such personalities. In that respect, I thank all Boğaziçi University members; I never felt as a foreigner.

I owe special thanks Temuçin Som, Erinç Dikici and Ekin Şahin for their help on completing this work.

I thank to my dear brother, Melih Öztemel. It is a great chance for me to have such a great brother. He had always encouraged, supported and helped me in the most difficult moments.

Last but not least, I am grateful to a *very special* person, my dear partner Burcu Akin. Without her support, I even could not start this work. I am grateful to her for being patient and supportive in this stressful period. Thanks to her, I always perceived this work as a step to be together forever.

ABSTRACT

DESIGN AND CONSTRUCTION OF A SECURE ID-CARD SYSTEM USING ROBUST IMAGE HASHING

We present a very simple, cryptographically secure, inexpensive and contactless biometric identity verification system. The system we propose is a printout of individual's biometric feature (pictorial data) such as face, hand, iris and a textual message. Furthermore, an HF RFID tag that involves RSA signature, compressed representations of the textual and pictorial data printed on a PVC card provides contactless communication. Digital signature is created using the private key of the signer. Verification is performed by an intelligent device involving a USB Camera and a RFID Reader.

There are mainly two processes: certification and authentication. ID is certified in the following way. First, the textual data is hashed using a cryptographically secure hashing algorithm such as MD5. Next, the pictorial data is hashed using robust hashing algorithms. Finally, hashed pictorial data, text and RSA signature of formers' are encoded into the RFID tag in ID card. Verification is performed by a simple and off-line device that contains the public key of the issuer. The signature, hashed textual and pictorial data are received from RFID tag via RFID reader, authenticity of the signature is checked. On the other side, a snapshot of the ID card is taken via USB camera. The image taken is then rectified to obtain textual and facial data. The textual data on the card is converted into a text-string using reliable optical character recognition. Robust image hashing methods are applied to the facial data. A comparison score is obtained comparing the resulting hash values with the hash values obtained from RFID tag. The identity document is authentic if the score is above the threshold.

ÖZET

GÜRBÜZ GÖRÜNTÜ İŞLEME İLE GÜVENLİ KİMLİK DOĞRULAMA SİSTEMİ TASARIMI VE YAPIMI

Bu çalışmada, kriptografik açıdan güvenli, basit, ucuz ve temassız bir biyometrik kimlik tanıma sistemi sunulmaktadır. Tasarlanan sistem, bireyin yüz, el, gözbebeği gibi resimsel biyometrik özelliklerinin ve isim, kimlik numarası gibi metinsel bilginin PVC üzerine baskısıyla oluşmuştur. Buna ilaveten, metinsel ve resimsel bilgilerin sıkıştırılmış bilgisinin ve RSA imzanın bulunduğu HF RFID etiket ile temassız haberleşme sağlanmıştır. Sayısal imza, imzalayan tarafın gizli anahtarı ile oluşturulmuştur. Doğrulama kısmı USB kamera ve RFID okuyucu içeren akıllı bir cihaz tarafından yapılmaktadır.

Sistem genel anlamda iki aşamalıdır: onaylama ve doğrulama. Kimlik şu şekilde oluşturulur ve belgelendirilir. Önce, metinsel veri MD5 gibi kriptografik açıdan güvenli hash yönetmleriyle sıkıştırılır. Sonra, resimsel veri gürbüz imge sıkıştırma yöntemleriyle sıkıştırılır. Son olarak, sıkıştırılmış resimsel ve metinsel veri ve bunların RSA imzaları ile beraber kimlik kartı içindeki RFID etiketine yazılır. Doğrulama, kimlik kartını imzalayan kısmın gizli anahtarını bulunduran basit ve çevirimdışı bir cihaz tarafından yapılır. İmza, sıkıştırılmış resimsel ve metinsel veri RFID etiketinden RFID okuyucu aracılığıyla alınır ve hemen imzanın geçerliliği kontrol edilir. Bir diğer yandan, kimlik kartının USB kamera tarafından resmi çekilir. Alınan resim, metinsel ve resimsel kart verisinin düzgün çıkarılması için doğrultulur. Metinsel veri, güvenilir optik karakter tanıma yöntemleriyle metin haline dönüştürülür. Gürbüz imge sıkıştırma yöntemleri yüz resmine de uygulanır. RFID'den alınan sıkıştırılmış sayısal bilgilerle kartın resminden çıkarılan bilgilerin sıkıştırılmış hali puan tabanlı bir yapıyla karşılaştırılır. Sonuç eşiğin üzerindeyse kimlik gerçek olarak sınıflandırılır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

$\mathbf{A}$	Model matrix for projective transformation
h_f	Hash value of face
h_t	Hash value of text by traditional hashing
$\mathbf{h_t}$	Hash value of text by novel text hashing
m	Concatenation of face hash and text hash: message
p	Parameter matrix in projective transformation
s	Digital signature
$\mathbf{x_i}$	Feature vector for character c_i

ID	Identity document
MD5	Message digest 5
NMF	Non-negative matrix factorization
NNC	Nearest neighbor classifier
OCR	Optical character recognition
SVD	Singular value decomposition

1. INTRODUCTION

1.1. Problem Statement and Motivation

An identity certification such as a nationality identity document, passport, driver's license or visa, mainly consists of a personal portrait photo, an arbitrary message, and one or more features to guarantee authenticity. Commonly, the authenticity is assured using sophisticated printing procedures that are difficult to replicate: holograms, watermarks, micro-printing and threading, special print paper, and chemical coating [1]. With the improvements in modern printing technologies, high quality printing devices are available with relatively lower prices. With the availability of such printers, most personal ID documents is subjected to forgery with results perceptually comparable to originals. Besides, authentication of imprinted features via electronic devices is complex and expensive [1].

On the other hand, authentication of digital ID schemes such as smart cards or laser cards can be made highly reliable using off-the-shelf public cryptography [2]. Typically, the stored photograph and the textual message are hashed and then signed using the private key of the issuer. In-field authentication is performed using the public key of the issuer by a verification device. The security of such systems can be made to follow strict security standards however; their cost makes them undesirable for widespread applications.

1.2. System Overview

In this study, we present an identity certification system that is contactless, inexpensive, HF-RFID based and cryptographically secure. System relies on asymmetric public-key cryptography for security. Issuer and verifier modules are illustrated in Figures 1.1 and 1.3.

Information that is embedded on the system is both pictorial and textual. The

pictorial data can be one of the biometric features such as face, iris, hand or fingerprint of the ID holder. We will concentrate mainly on face, suggesting application techniques for the other features. The textual data can be of arbitrary length. Both textual and pictorial data is printed on the ID. The authenticity of the ID is certified in the following way. First, the textual data is hashed using a novel algorithm based on cryptographically secure hashing algorithms such as MD5 and SHA1 [3]. The resulting fixed length hash is denoted as $\mathbf{h_t}$. Next, the pictorial data is hashed using robust perceptual image hashing algorithms and denoted as h_f. Hash values h_f and $\mathbf{h_t}$ are merged into a message using an reversible operator. Then, the message m is signed with the private key of the system issuer, producing the digital signature s. Digital signature s together with hash values h_f and $\mathbf{h_t}$ are embedded into HF RFID. Thus the ID card is ready to be used and certification process is complete.

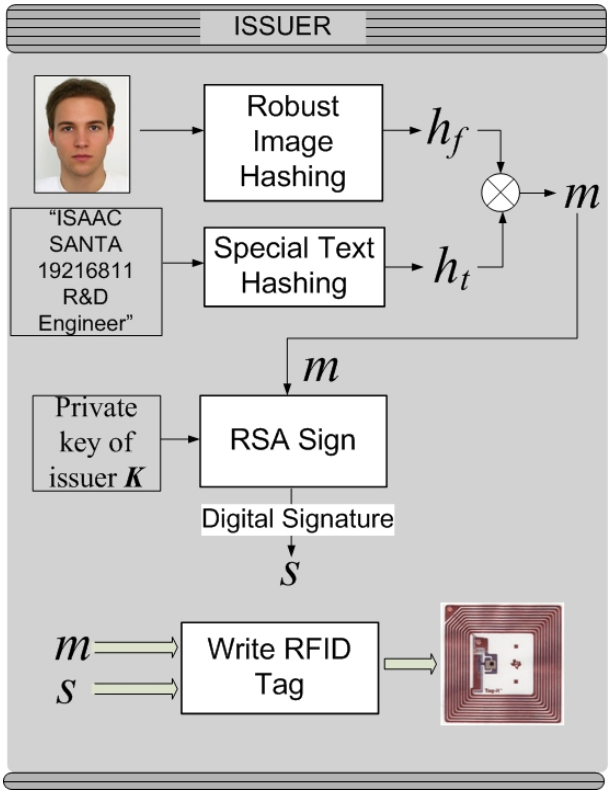


Figure 1.1. Block diagram of card card issuing

The verifier initially captures the photograph of the ID card in which the textual and the pictorial data is included. In order to obtain textual and pictorial data properly, initially the place of ID card must be determined in the image. Having determined the place of ID card roughly in the image, synchronization markers of ID card is

Figure 1.2. Issued card

determined. The synchronization markers on ID card are previously assigned, constant and are placed for rotation invariance. Since it is highly possible that the plane defined by the ID is not exactly parallel to that of camera, the captured photograph must be rectified for synchronization. Synchronization of the photograph is performed by *projective transformation* which will be defined in section 2.2.1. After rectifying the image and obtaining ID from the picture properly, text and face is extracted. Text is recognized using robust optical character recognition schemes. Next, a novel hashing algorithm is applied to text, resulting $\mathbf{h_t'}$. Extracted face is normalized, resized and robust perceptual image hashing algorithms performed. Resulting hash is h_f'. Since original face image is printed on ID on issuer side and scanned from ID on verifier side, there will be serious distortion on obtained image. However, robust perceptual image hashing algorithms produce similar hash values for original and rectified face, as expected.

On the other side, RFID reader receives embedded information on RFID tag. The received information includes s; digital signature and m from which h_f (hash value of the original face image) and $\mathbf{h_t}$ (hash value of the original text) could be reproduced exactly. Once the digital signature, s, is obtained, it is encrypted with the public RSA key and m' (which is actually m if the ID is issued by authorized issuer) is obtained. At this point verifier compares m and m'. If they are same, it is decided that ID is produced by an authorized issuer. Next the verifier generates a score comparing original face and text hash values with obtained face and text hash values obtained via

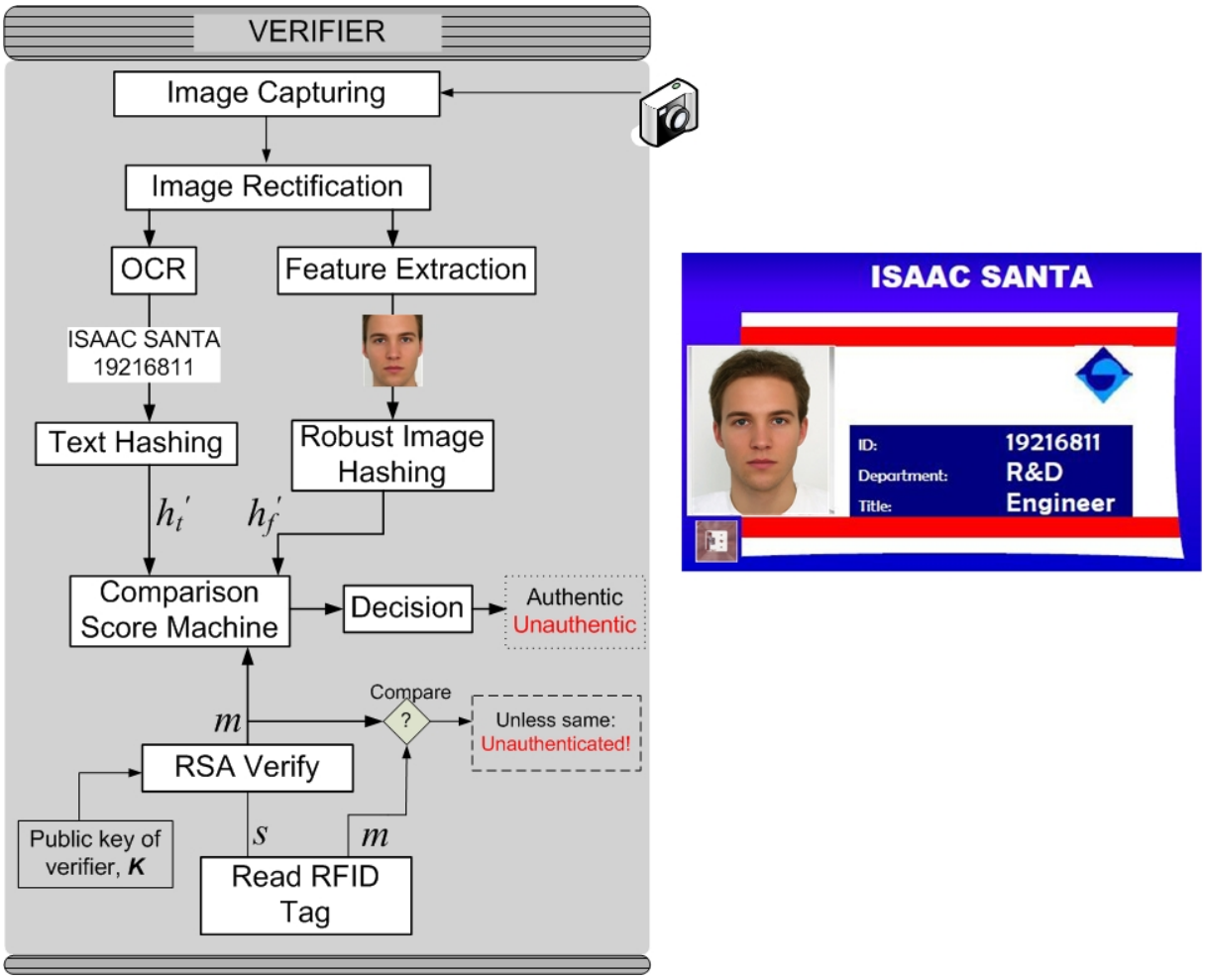


Figure 1.3. Block diagram of card verifying

camera capture. If the resulting score is above the threshold, ID is not tampered and decided to be authentic.

1.3. Related Work

The system we present relies on the work of Kirovski and Jojić [4]. However, there are several distinctions. [4] relies on printing the encrypted and hashed messages on a 2D color bar-code. Therefore, verification step involves scanning of ID which makes the system contacting and thus less user friendly. Besides, [4] is very sensitive scanning. One bit misclassification of barcode causes false negative. Also, one character misclassification of OCR engine causes false negative. However, in our work, we use RFID to keep all digital information. Therefore, the digital information produced by the issuer is exactly the same as the one obtained in the verifier side. Also, for

text hashing we developed a score based novel decision scheme. This prevents false negatives due to misclassification of one or more characters in OCR. The novel text hashing algorithm we developed will be described in 3.2.2. Robust image hashing based related survey will be introduced in 4.

2. PRACTICAL DATA NORMALIZATION

As mentioned in section 1.2, ID verification step starts with capturing photograph of ID via a USB camera.

In order to be sure that identity document is not tampered physically (i.e. insertion of a new photograph or text on identity document), we must be sure that printed pictorial (face) and textual (name, ID) data is exactly the same as in issuer side. Hash values of textual and pictorial data is embedded on RFID tag, therefore we may be able to compare them if we extract the data from captured photograph. Hence, we need to extract the following data from ID capture:

- **Photograph of the ID owner:** Once we have extracted the pictorial data, we will apply robust image methods exactly same as in issuer side. Since hash of original photograph of ID owner is embedded in the RFID tag, we will be able to decide whether printed photograph of ID owner is tampered or not.
- **Textual Data:** Similar to previous item, once textual data is extracted from captured picture of ID, a novel hashing algorithm is applied and a comparison score based on original textual data is calculated.
- **Digital information on RFID:** Digital information is kept in RFID tag on identity document. RFID tag that is used belongs to standard ISO15693. Digital information is read through an USB HF-RFID reader. As previously mentioned, digital information consists of digital signature and concatenation of hash values of textual and facial data. Digital signature is to verify that identity document is issued by an authenticated issuer.

In order to extract mentioned fundamental data from snapshot, it is clear that initially the position of ID card in the image must be determined. Having determined the position, now rectification of ID is necessary. This is because ID card might not probably be parallel to the camera plane. (i.e. ID forms a quadrilateral instead of a rectangle.) After rectification, ID is obtained properly, having a rectangular shape.

Rectification of ID is performed via projective transformation which will be defined in section 2.2.1.

After ID card rectification, regular appearance of ID as in figure 2.1 is obtained for data (text and face) collection.

Figure 2.1. Extracted and rectified ID from captured image

This section starts with capturing image of identity card for verification via camera and consists all steps required to obtain textual (i.e. name) and pictorial (i.e. face) data on ID. Following subsections are explanations of those processes step by step.

2.1. Identity Card Determination in Picture

An image captured for ID verification is exposed to several external influences. Those influences include light, position of ID card and background complexity (Objects with similar color and shape with identity card may exist in image). In order to eliminate those influences and to be able to correctly and robustly determine the position of identity card, identity card has several distinct properties. Those features are shown in figure 2.2.

As a first step, an example photograph captured by the USB camera for ID verification can be seen in figure 2.3.

The purpose of this section is to explain steps required to determine the synchronization points in identity card.

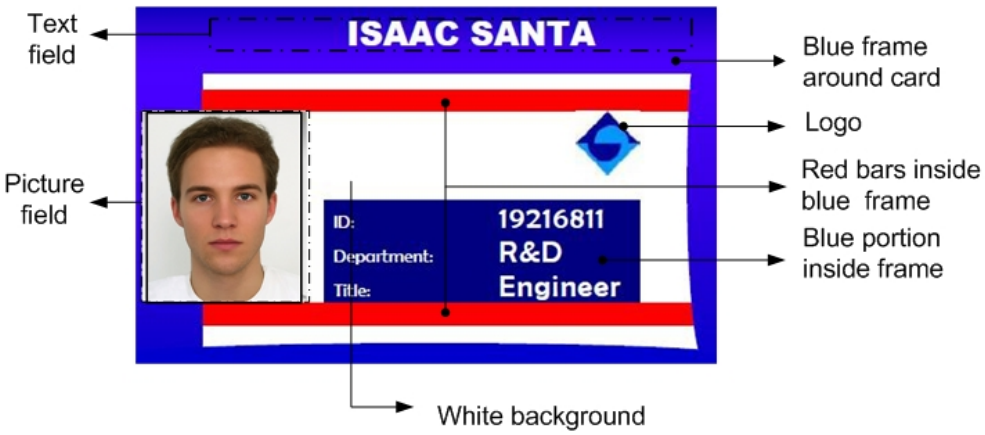


Figure 2.2. ID physical appearance properties

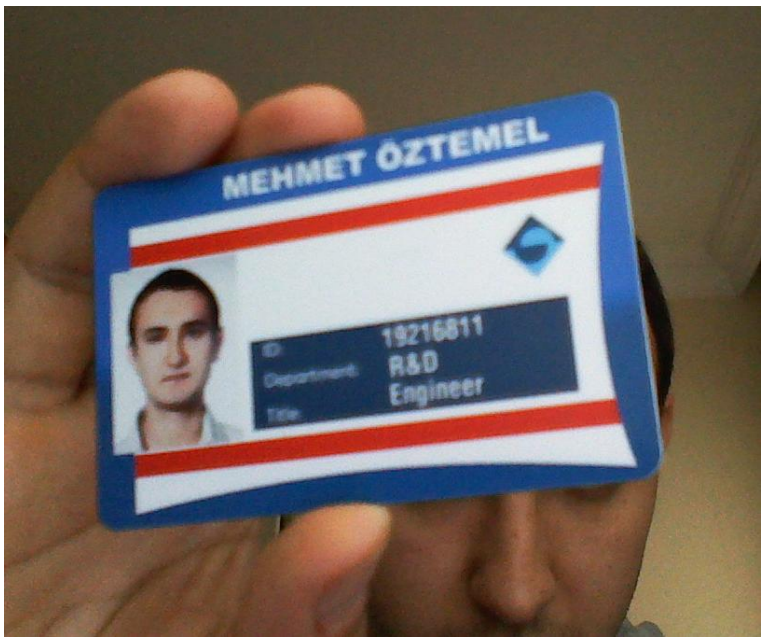


Figure 2.3. Original photograph captured by the USB camera in verifier side

2.1.1. Preprocessing

Environmental conditions such as light, background complexity, color quality, angle and position of identity card in picture may vary in time. On the other hand, determining the positions of synchronization points is quite important because unless identity card is rectified regularly, it is almost impossible to extract facial and textual data from image. A small error in the positions of synchronization points amplify synchronization error rate and cause the verifier to decide false negatives. Therefore, the algorithm to determine identity card needs to work robust under almost all conditions.

Edge detection type classification fails in determining card because it is very sensitive to background noise in the image.

Preprocessing steps can be summarized as follows:

- **Histogram Mapping:** A typical printed identity card is expected to have similar contrast. However, captured photograph of identity card may have distorted contrast due to lighting conditions. The contrast of the image is enhanced by transforming the values in an intensity image so that the histogram of the output image approximately matches desired histogram.
- **Color Filtering:** As discussed before, identity card has some specific properties as shown in figure 2.2. It has a blue frame, two red bars inside, a blue rectangle containing text and a logo. After contrast enhancement, a filter that permits only blue color, is applied to the image. The resulting image is a black and white image. A pixel is white if corresponding pixel is decided to be blue and vice versa.
- **Morphological Operations:** Color filter applied image has generally a noisy structure. To be able to eliminate blue portions other than identity card, we assume that blue portion of ID is connected. Next, we apply several combinations of dilation and erosion to eliminate other objects and noise. Also, objects with small number of connected points are canceled out.
- **Edge detection:** At this point, most probably, the resulting image consists blue frame and edge detection is applied to the image.
- **Check point:** At this point, identity card is roughly determined. To be sure that we have really focused on identity card but not another object, following items are checked:
  - Width-length ratio of identity card
  - Two red bars inside blue frame
  - Logo

  If the points above are not verified, then step 2.1.1 is applied with red filter in combination with blue filter.

2.1.2. Determining Positions of ID Synchronization Points

Synchronization points in original ID card is shown in figure 2.4.

Figure 2.4. Location of eight synchronization points in identity document for image rectification. (Yellow squares)

Correctly determining synchronization points requires captured image to be preprocessed with attention. The output image of the preprocessing step is shown in figure 2.5.

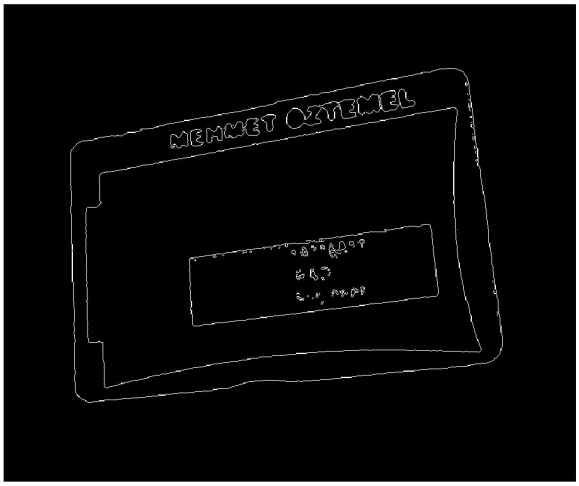


Figure 2.5. Preprocessed image for determining synchronization points.

For each side of identity document and the rectangle inside, we select some points randomly. The reason for that is, we view preprocessed image as a combination of

virtual lines. We use those points to form lines and intersection point of those lines gives us synchronization points. Synchronization points of captured image is shown in the image 2.6.

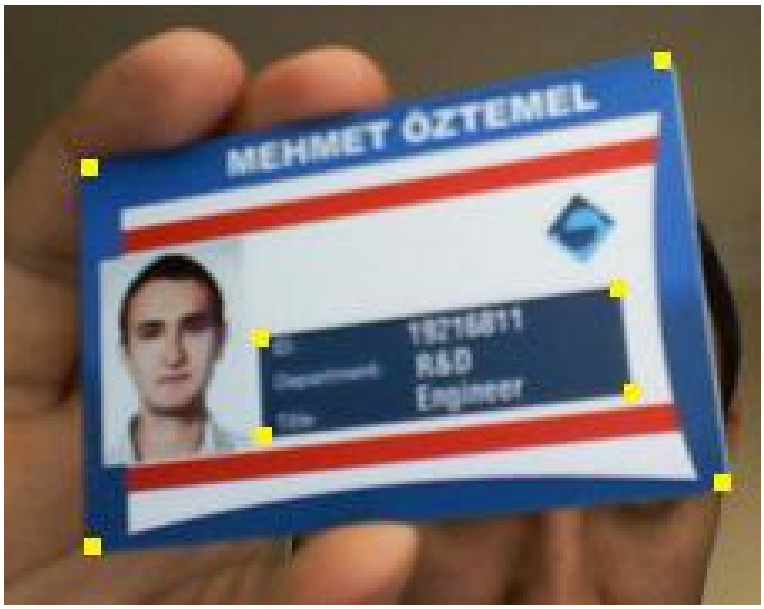


Figure 2.6. Captured image with synchronization points are determined

Positions (coordinates) of synchronization points form a model matrix that determine parameters essential for rectifying image.

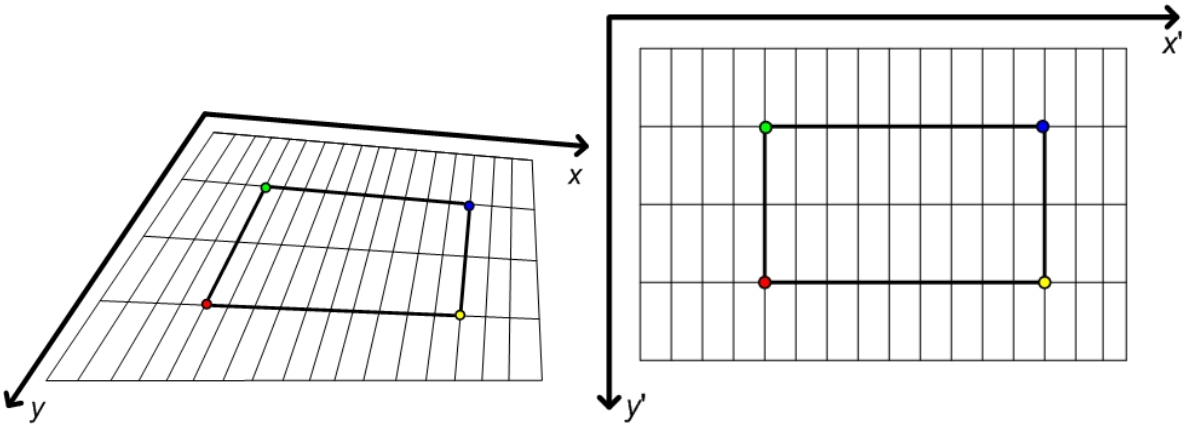
2.2. Identity Card Rectification

After synchronization points coordinates' are obtained, we are now able to transform captured image such that identity document card has a regular shape. To relate original image and captured synchronization points *projective transformation* is used. Section 2.2.1 describes *projective transformation* and application to current image rectification problem.

2.2.1. Projective Transformation

Projective transformation is defined as a linear transformation among different coordinate systems. Projective transformation can be applied to an image, f, defined over (x, y) coordinate system to produce an image, g, defined over an coordinate system

(x', y'). The transformation produces images where straight lines remain straight, but parallel lines converge towards vanishing points.



(a) Original image (b) Image after projective transformation

Figure 2.7. Illustration of projective transformation on an image

It has several applications on multimedia such as image registration and rectification. Projective transformation, when applied, generally modifies the local geometry of an image with a set of identifiable points called source or features, to fit a set with the same number of points in another image, called target. For example, it may be of interest to align two or more images taken at roughly the same time (pointing out that images are time invariant during the process), but using different instruments, such as an MRI (Magnetic Resonance Imaging) scan and a PET (Positron Emission Tomography) scan. Another example could be that the images are taken at different times using the same instrument, such as satellite images of a location taken several days apart. In these examples, input image geometry is modified via projective transformation to obtain relevant patterns or cascaded images [5].

2.2.1.1. Model. Using the fact that projective transformation maps lines to lines, the transformation that maps (x_i, y_i) to (x'_i, y'_i) is defined by the equation

$$\begin{pmatrix} x'_i \\ y'_i \end{pmatrix} = \frac{1}{gx_i + hy_i + 1} \begin{pmatrix} ax_i + by_i + c \\ dx_i + ey_i + f \end{pmatrix} \tag{2.1}$$

where (x_i, y_i) is the original point, (x'_i, y'_i) is the transformed point and (a, b, c, d, e, f, g, h) are the projective transformation parameters. Note that the model is linear in the parameters. Then the projective transformation equation can be rearranged as follows

$$\begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i x'_i & y_i x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -x_i y'_i & -y_i y'_i \end{pmatrix} . \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{pmatrix} \tag{2.2}$$

If there are n points to be transformed, Equation(2.2) can be modified as

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 x'_1 & -y_1 x'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 x'_2 & -y_2 x'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 1 & -x_n x'_n & -y_n x'_n \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 y'_1 & -y_1 y'_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 y'_2 & -y_2 y'_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_n y'_n & y_n y'_n \end{pmatrix} . \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{pmatrix} \tag{2.3}$$

The equation above then can be written in a compact matrix form as follows

$$\mathbf{b=Ap} \tag{2.4}$$

where b is the $2n \times 1$ vector of x and y, $\mathbf{A}$ is the $2n \times 8$ model matrix and p is the 8×1 vector of unknown transformation parameters.

2.2.1.2. Estimation of Transformation Parameters. The conditions related to existence of solution to the equation 2.4 is explained in [6]. However, in some cases the conditions may be true in a mathematical sense, but not usefully true in a numerical sense: it may not be possible to compute a reliable solution using finite-precision arithmetic on a physical computation device. Another point is that, systems of equations that are poorly conditioned are sensitive to small changes in the values. Since, practically speaking, there are always inaccuracies in measured data (the coordinates of synchronization points in this project), the solution to these equations may be almost meaningless.

The singular value decomposition (SVD) provides robust solutions of both overdetermined and underdetermined problems, matrix approximation, and conditioning of ill-conditioned systems [6]. In this problem, we use 8 synchronization points as shown in figure 2.4. Using 8 synchronization points, matrix A becomes overdetermined ($A_{8\times16}$). Therefore we need an approximation to $\mathbf{A}$ which we shall denote as $\mathbf{A_k}$.

The singular value decomposition (SVD) of a rectangular matrix $A \in R^{m\times m}$ is a decomposition of the form

$$A = USV^T \tag{2.5}$$

where U and V are orthogonal matrices, and S is a diagonal matrix. The columns u_i and v_i of U and V are called the left and right singular vectors, respectively, and the diagonal elements s_i of S are called the singular values. The singular vectors form orthonormal bases, and the important relation

$$Av_i = s_i u_i \tag{2.6}$$

shows that each right singular vector is mapped onto the corresponding left singular vector, and the scale is the corresponding singular value. Finally, the approximated

matrix A_k can be formed as

$$A_k = u_1 s_1 v_1^T + \ldots + u_k s_k v_k^T \tag{2.7}$$

where k is the number of retained singular values. Hence small singular values of A is discarded [7].

2.2.1.3. Resampling. As we have discussed before, a projective transformation (a geometric transformation in the most general sense) defines the relation between the points in two images. This relation can be expressed in two ways.

$$\mathbf{x}' = T(\mathbf{x}) \text{ Forward mapping} \tag{2.8}$$

$$\mathbf{x} = T^{-1}(\mathbf{x}') \text{ Inverse mapping} \tag{2.9}$$

where T specifies the mapping function and T^{-1} its inverse. Most computational methods for applying a projective transformation falls into two categories; methods that use *forward mapping* and methods that use *inverse mapping*.

Methods that use *forward mapping* scans each input pixel in the image, apply the transformation and obtain the destination pixel copying the value of the input pixel. Generally, the pixel of the input image lies in between the pixels of the output image. With forward mapping, it is not appropriate to write the value of the input pixel just to the nearest pixel in the output image. Then, it may happen that a value is never written to some of the pixels of the output image (holes) while others receive a value more than once (overlap). Therefore, an appropriate technique must be found to distribute the value of the input pixel to several output pixels. The easiest procedure is to regard pixels as squares and to take the fraction of the area of the input pixel that covers the output pixel as the weighting factor. Each output pixel accumulates the corresponding fractions of the input pixels which add up to cover the output pixel [8].

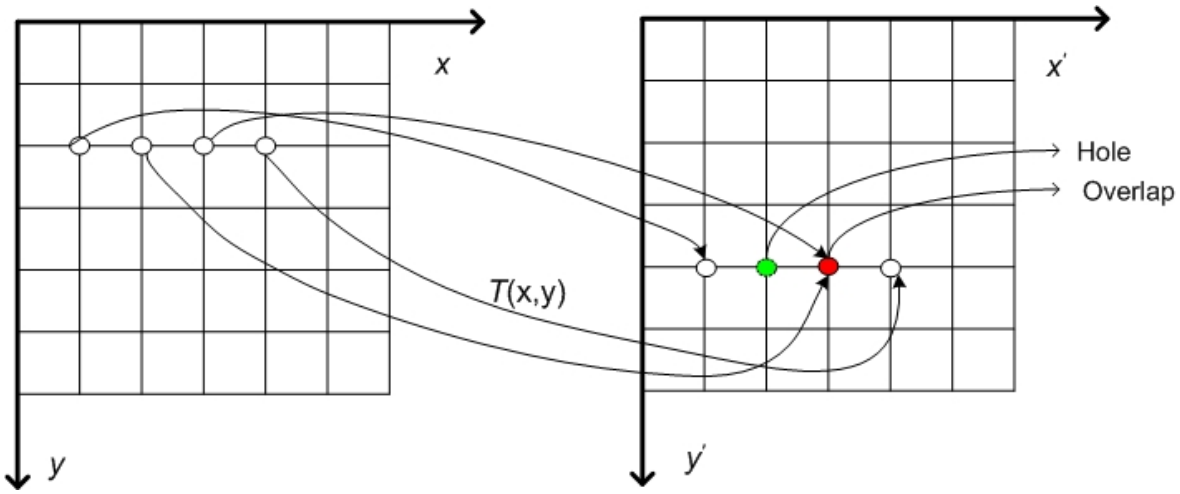


Figure 2.8. Illustration of a forward mapping with transformation matrix $T(x, y)$. Hole and overlapped pixels exist in output image.

An inverse mapping procedure scans each output pixel, computes the corresponding location in the input image using the inverse transformation matrix and interpolates among the neighbor input image pixels to determine the output pixel value. With *inverse mapping*, the coordinates of the output pixels are mapped backed onto the input image as shown in figure 2.9. It is quite obvious that, with this scheme, any hole and overlap problem in the output image is avoided. However, the coordinate of the output image does not in general hit a pixel in the input image but lies in between the pixels which is an interpolation problem. Therefore, its correct value must be interpolated from the neighbor pixels in the input image.

The transformation in equation 2.10 determines the relation among coordinates of points in the reference image with the coordinates of the corresponding points in the sensed image.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \frac{1}{gx + hy + 1} \begin{pmatrix} ax + by + c \\ dx + ey + f \end{pmatrix} \tag{2.10}$$

Given the (x, y) coordinates of a point in the reference image, equation determines

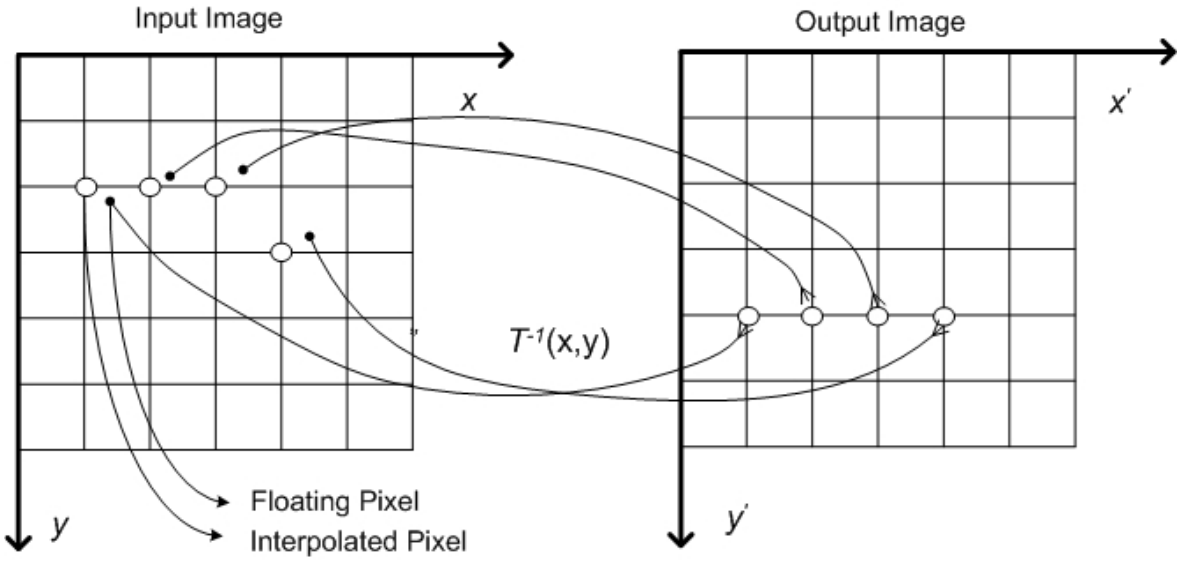


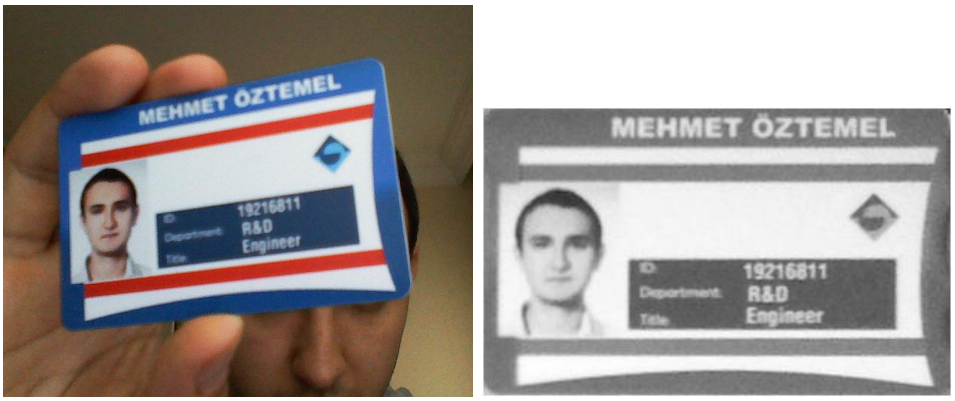
Figure 2.9. Illustration of an inverse mapping with transformation matrix $T(x, y)$. Interpolation problem occurs in input image.

the (x', y') coordinates of the corresponding point in the sensed image. By reading the intensity at (x, y) in the sensed image, and saving it in the reference image for point (x', y'), the sensed image is point-by-point resampled. Although (x, y) are integers, (x', y') are floating numbers. Since intensities at only integer coordinates are available in the sensed image, the intensity at point (x', y') has to be estimated from the intensities of a small number of surrounding pixels. Different interpolation methods to achieve this estimation have been developed. A few example is nearest neighbor, bilinear interpolation and cubic interpolation. We will be using nearest neighbor interpolation to resample the image.

An example of identity photograph captured by camera and its extracted version can be seen in figure 2.10.

2.3. Optical Character Recognition

Optical character recognition (OCR) is one of the most studied application of pattern recognition [9]. While OCR problem is popular, it has various sub-applications in different media with different physical environment. In this work we will be dealing with OCR in still images, with constant font and size. While some OCR applications



(a) Original image captured by camera (b) Extracted card after rectification

Figure 2.10. Illustration of identity card extraction with projective transformation

take advantage of the content to fit a language model and improve character error rate, we will work on textual data with no priori information.

After card rectification, since the position (upper and lower coordinates) of textual data is almost constant, frame that includes text can easily be extracted. Next, text frame is segmented according to the vertical projection of pixels and hence character extraction is complete.

Size of extracted characters is 20×20 pixels at most. Actually, the size of characters totally depends on the captured image because the further the identity card to the camera, the smaller characters will be extracted and vice versa. However, after determining the synchronization points in identity card, it is resized to a desired dimension, therefore character sizes are constant as mentioned before.

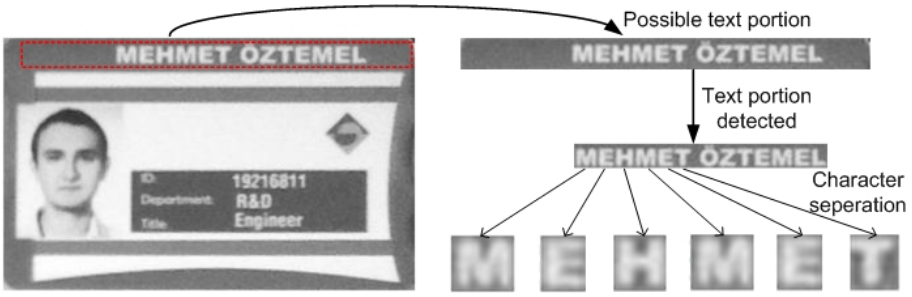


Figure 2.11. Character separation from extracted ID for OCR machine

In the next section, nearest neighbor classification which will be used in OCR is described.

2.3.1. Nearest Neighbor Classification

Nearest neighbor classification is a method for classifying objects based on closest training examples in the feature space. Generally, a set of objects with known classification (training set) is obtained. Next, a new object to be classified is *compared* among all possible objects in training set. NN algorithm finds the nearest neighbor to the target object. Hence the target object falls into the class of its nearest neighbor [10].

Terms *compare* and *nearest* should be expressed more explicitly. In pattern recognition, objects are generally represented as a point in multi-dimensional feature space. The dimension is determined by the number of features used to describe patterns. Comparison is performed using a predefined distance metric such as Euclidean distance and Manhattan distance. The nearest neighbor is selected among the training samples with smallest distance in chosen metric [11].

2.3.2. k-Nearest Neighbor Classification

If the number of training points (preclassified points) is large, distribution of classes in feature space may not be separated perfectly. Therefore, instead of choosing the nearest neighbor, a choice can be performed among the nearest k neighbors. This method is referred to as the k-nearest neighbor classification. Notice that NN classification is same as 1-NN classification.

The number k should be:

1. Large to minimize the probability of misclassifying x
2. Small (with respect to the number of training samples) so that the points are close enough to x to give an accurate estimate of the true class of x.

2.3.3. Optical Character Recognition Using k-Nearest Neighbor Classification

After segmenting characters from captured photograph as in figure 2.11, we are now able to apply classification algorithm. However, as mentioned before, we need to define a feature set x and a distance metric $d(x_i, x_j)$ where x_i is the feature set corresponding to the sample i.

In the literature, for optical character recognition problem, various feature extraction schemes are used. To extract features, we divide a character pattern which is about 20×20 pixels to sub-rectangles of 5×10 pixels (10 pixels height, 5 pixels width). For each sub-rectangle, we calculate the average and variance of intensity values. An example of character division and a feature of intensity mean and variance is shown in figure 2.12 where x_{i1} and x_{i2} denote intensity mean and variance of i^{th} sub-rectangle.

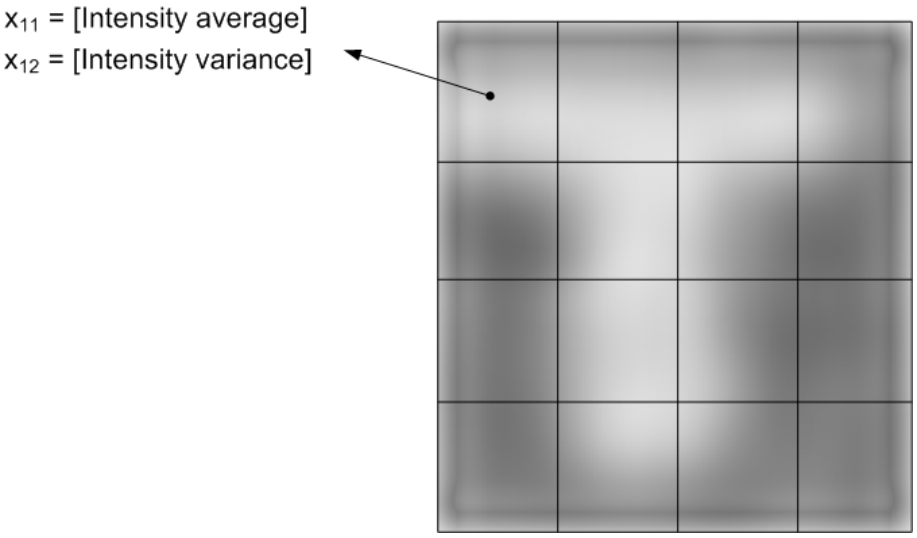


Figure 2.12. Feature description of character T.

Mean of a sub-rectangle is calculated as follows:

$$\begin{aligned} x_{i1} &= \mu_i \\ &= \sum_{k=1}^{n} I_{ik} \end{aligned} \tag{2.11}$$

where x_{i1} denotes mean of intensity values of i^{th} sub-rectangle, I_{ik} denotes the pixel

intensity of k^{th} pixel at i^{th} sub-rectangle and n is the number of pixels in sub-rectangle. Similarly, variance is calculated as

$$\begin{aligned} x_{i2} &= \sigma_i^2 \\ &= \sum_{k=1}^{n} (\mu_i - I_{ik})^2 \end{aligned} \tag{2.12}$$

where x_{i2} denotes variance of intensity values of i^{th} sub-rectangle, $I_{i,k}$ denotes the pixel intensity of k^{th} pixel at i_{th} sub-rectangle and n is the number of pixels in sub-rectangle. Having obtained mean and variance values, the feature vector is formed as follows

$$\mathbf{x_i}' = \begin{bmatrix} x_{1,1} & x_{2,1} & \dots & x_{16,1} & x_{1,2} & x_{2,2} & \dots & x_{16,2} \end{bmatrix} \tag{2.13}$$

More precisely, we call x'_m a nearest neighbor to x if $\min d(x_i, x) = d(x'_m, x)$ where $i = 1, 2, \dots, n$ and n is the number of training samples. Remaining k-nearest neighbor of x is determined in the same way. Among k-neighbors, the one with the highest number of existence is chosen (a simple voting scheme).

2.3.3.1. Conclusion. k-NN method has several parameters and each parameter should be examined for the character recognition rate. Distance metric and the number k are k-NN algorithm method parameters. Here, the input image (segmented characters) is of great importance. The angle and distance of ID card to the camera are the main parameters that affect input image quality. Hence, the following parameters are examined:

- **Distance metric:** Euclidean and Manhattan distance metrics will be examined. Euclidean distance between points $\mathbf{p}$ and $\mathbf{q}$ is given by:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{2.14}$$

  Manhattan distance between points $\mathbf{p}$ and $\mathbf{q}$ is the sum of the lengths of the

projections of the line segment between the points onto the coordinate axes and given by:

$$d(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} |p_i - q_i| \tag{2.15}$$

- **Number of nearest neighbor, k:** As mentioned before, nearest neighbor, k should be large to minimize the probability of misclassifying and small so that the points are close enough to x to give an accurate estimate of the true class of x.
- **The angle of ID card to the camera:** After capturing image of ID card via camera for verification, synchronization points are determined and projective transformation is applied to the input image in order to extract ID card regularly as shown in figure 2.1. While forming new image (rectified ID card), the image is resampled and interpolation methods are applied. Resampling and interpolation -even if they are perfectly applied- causes loss of pixel intensity information.
- **The distance of ID card to the camera:** Similar to the parameter *angle*, the *distance* of ID card to the camera is quite important for character pixel resolution. Although the upper distance limit to the camera is around 12cm (due to receiver range of HF-RFID reader), we examine the parameter effect with higher limits (upto $45cm$).

All of the parameters above are tested with all possible combinations. Parameters are selected from a set as shown in table 2.1.

Table 2.1. Test parameters for k-NN algorithm

Parameter	**Range/Type**
Distance metric	Euclidean, Manhattan
Number of nearest neighbors, k	1,3,5,4,5,10
Angle (degrees)	0, 5, 15, 30, 45
Distance (cm)	10, 15,20, 30, 45

The rest of the section includes results and comments. Parameters in the tables and their correspondences in figure 2.2. While examining all possible combinations of

Table 2.2. Explanation of parameters used in experimental setup

Abbreviation	**Explanation**
Distance metric	Distance metric used in k-NN classifier to compare two points
k	Number of nearest neighbor in k-NN classifier
Angle (degrees)	Angle between identity card and camera
Distance (cm)	Distance between identity card and camera

determined set of parameters, we keep distance metric and k fixed and observe the effect of other parameters. In experiment 1, Euclidean metric and $k = 1$ chosen, the

Table 2.3. Experiment 1. Character recognition rates for $k = 1$, Euclidean distance

Recognition rate (%)					
Metric: Euclidean, $k = 1$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	80.92	78.14	74.94	71.07	65.10
5	80.72	78.24	74.16	69.12	62.89
15	80.10	77.52	71.61	66.52	59.90
30	79.22	77.63	72.23	63.13	59.13
45	77.43	76.54	71.48	62.91	58.82

success rate is not acceptable. The effect of angle and distance is obvious. As the angle and distance increase recognition rate decreases. Now, distance metric will be switched to Manhattan metric with the same parameters in table 2.3.

In experiment 2, Manhattan metric and $k = 1$ chosen, the success rate is not still acceptable. Changing distance metric to Manhattan has slightly decreased recognition rate. Influence of angle and distance is still obvious. The biggest decrease in recognition rate is caused by 30-45 degrees rotation and 30-45 cm distance change.

Table 2.4. Experiment 2. Character recognition rates for $k = 1$, Manhattan distance

Recognition rate (%)					
Metric: Manhattan, $k = 1$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	80.76	77.98	73.77	72.12	66.21
5	80.42	78.63	73.68	71.68	63.98
15	80.20	77.12	70.77	66.59	62.70
30	78.12	76.88	69.69	61.74	61.13
45	76.51	76.35	68.18	59.62	58.14

In the third experiment, k which is an important parameter of the classifier is increased, therefore we expect recognition rate to increase.

Table 2.5. Experiment 3. Character recognition rates for $k = 5$, Euclidean distance

Recognition rate (%)					
Metric: Euclidean, $k = 5$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	94.93	93.88	93.11	91.88	87.34
5	94.72	93.53	93.16	91.12	86.89
15	94.76	93.19	92.21	90.58	85.90
30	94.12	92.69	91.26	87.13	83.13
45	93.88	91.88	89.12	85.91	81.84

In experiment 3, introducing the variable $k = 5$, the recognition rate quite increased. The reason for that is, distribution of training sets in multi-dimensional feature space is close to each other. Therefore, the nearest neighbor might correspond to another class with very similar features to the class it should supposed to be. In that case, we choose 5 nearest neighbors and vote among them. However, there is not

a unique value for parameter k. In the following experiments, influence of parameter k is investigated.

The next step is changing distance metric to Manhattan with the other parameters are same as in table 2.5, experiment 3. In experiment 4, distance metric is switched

Table 2.6. Experiment 4. Character recognition rates for $k = 5$, Manhattan distance

Recognition rate (%)					
Metric: Manhattan, $k = 5$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	96.01	94.97	94.01	93.07	91.10
5	95.72	94.02	93.17	92.56	90.89
15	95.66	93.32	92.7	91.03	90.27
30	94.78	92.63	91.31	90.14	89.54
45	94.12	92.04	90.48	89.91	87.82

to Manhattan remaining all other parameters same as in experiment 3. Using Manhattan metric increased recognition rate around 1.5% The reason for that is, Manhattan distance provides training sets to be separated better than Euclidean distance.

In the next experiment, we increase the number of nearest neighbors, k, and observe influence.

In experiment 5, the setup is same with experiment 3 except that k increased to 9. As we have mentioned before, optimal number of nearest neighbor is not unique and found heuristically.

The results as shown in table 2.7 show us that recognition rate still increases. Experiments with higher number of nearest neighbor will be performed; however, in the previous experiments, we have experienced that Manhattan performs better than

Table 2.7. Experiment 5. Character recognition rates for $k = 9$, Euclidean distance

Recognition rate (%)					
Metric: Euclidean, $k = 9$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	95.77	94.89	94.47	92.78	89.14
5	95.72	94.51	94.12	92.39	88.65
15	95.32	94.12	93.79	91.87	88.53
30	94.45	93.36	93.29	91.19	87.79
45	93.79	92.47	92.69	90.17	87.12

Euclidean distance, therefore keeping k and all other parameters constant, an experiment with Manhattan distance metric is performed next.

Table 2.8. Experiment 6. Character recognition rates for $k = 9$, Manhattan distance

Recognition rate (%)					
Metric: Manhattan, $k = 9$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	98.67	97.98	97.54	95.84	92.12
5	98.70	97.43	97.10	95.43	91.75
15	98.12	97.01	96.80	94.89	91.43
30	97.40	96.33	96.30	94.20	90.87
45	96.88	95.54	95.71	93.12	90.12

In experiment 6, only the distance metric is changed, keeping all other parameters fixed as in experiment 5. The results as in table 2.8 show us that using Manhattan distance metric instead of Euclidean performs 2.9% better.

In previous examples, we are able to observe the effect of angle, distance and distance metric. However, we have not reached near-optimal k value yet. This issue is handled in the next two experiments.

Table 2.9. Experiment 7. Character recognition rates for $k = 15$, Euclidean distance

Recognition rate (%)					
Metric: Euclidean, $k = 15$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	96.27	95.33	94.87	93.39	89.74
5	96.26	95.01	94.49	92.87	89.35
15	95.91	94.73	94.12	92.30	89.21
30	94.99	93.84	93.69	91.73	88.33
45	94.12	92.88	93.12	90.51	87.51

Table 2.10. Experiment 8. Character recognition rates for $k = 15$, Manhattan distance

Recognition rate (%)					
Metric: Manhattan, $k = 15$					
Angle (degrees)	**Distance (cm)**				
	10	15	20	30	45
0	98.88	98.12	97.63	96.01	92.27
5	98.79	97.55	97.21	95.62	91.98
15	98.33	97.21	96.92	94.98	91.57
30	97.35	96.56	96.44	94.42	91.00
45	97.01	95.66	95.85	93.31	90.32

In table 2.9 and 2.10, the results of experiments 7 and 8 are demonstrated. In experiment 7 and 8 k is 15 and Euclidean and Manhattan distance effect is observed

respectively, same as before previous experiments.

Increasing nearest neighbors, k, to 15 does not actually contribute to recognition rate. Therefore, we can conclude that near-optimal k value is 9.

Another point is that, in experiments 7 and 8, the contribution of Manhattan distance with respect to Euclidean distance is obvious. Manhattan distance performs 2.6% better compared to Euclidean distance.

To summarize, the influence of each parameter can be explained as:

- **Distance metric:** We have used two distance metrics: Manhattan distance and Euclidean distance. Except the experiments 1 and 2, where the nearest neighbor parameter, k is 1, Manhattan distance performed quite better. The difference among two distance metrics is 2.83% on the average. The reason for Manhattan distance to perform better is that, in terms of Manhattan distance metric, the classes are separated quite better.
  In this work, two famous distance metrics are used, however other distance metrics
- **Number of nearest neighbors, k:** We have performed experiments based on k where its value is $1, 5, 9$ and 15. The reason we have used odd number of neighbors is to prevent two candidates with an equal number of neighbors. In our experiments, we have observed that recognition rate is highest when $k = 9$. However, this value is not unique and based on class distributions. Even with different distance metrics and training sets, this value would probably be different.
- **Angle:** Angle is an important parameter in character recognition. As mentioned before, due to the angle between camera and identity card, projective transformation is applied to the picture captured by camera. As the angle gets larger, extracted identity card resolution decreases which causes loss of intensity information. This directly affects recognition rate. The highest decrease in recognition rate occurs when angle switched from 30 degrees to 45 degrees.
- **Distance:** Distance of the identity card to the camera is the most important parameter in character recognition. Because, while the identity card is closer to

the camera, character pixel sizes are higher. In fact, in the final extracted and rectified card, all characters are of same size however this is provided by resizing the image to a desired dimension. Distance causes loss of character intensity information, which in turn decreases the recognition rate.

It should also be noted that, HF-RFID receiver operates at 12cm distance to the RFID tag. Therefore, for system (identity verification system) performance, only 10-15cm distances should be considered, which have relatively higher recognition rates compared to other distances. Experiments are performed with higher distances to test classification performance.

3. SECURITY PRIMITIVES

3.1. Introduction

In this chapter, hashing schemes MD-5, SHA-1, RSA signature scheme and the novel text hashing algorithm based on MD-5 is described.

3.2. HASH FUNCTIONS

Hash functions take a message as input and return an output called *hash*. More specifically, a hash function h maps strings of any length to fixed length (i.e. n bits). For a domain D and range R with $h : D \rightarrow R$ and $|D| > |R|$, the function is obviously many to one. Therefore, the existence of collisions, namely distinct pair of inputs yielding an identical output is inevitable. The probability that two randomly chosen inputs to yield same output value is 2^{-n} (independent of t). The basic idea of cryptographic hash functions is that a hash value of a message is a compact representative digest and can be treated as if it uniquely identifies the input message.

3.2.1. MD5 and SHA1

MD5 and SHA1 are two famous cryptographically secure hashing algorithms that have been employed in a wide variety of security application. MD5 is 128 bits and SHA1 is 160 bits hash sized. The most famous property of these hash functions is that, one bit change of the input extremely changes the output. The output is also almost perfectly uniformly distributed. Those hash functions having the greatest attention in practice are based on the MD4 hash function. Details of MD4 is explained in [12].

3.2.2. Novel Score Based Text Hashing Algorithm

As we have mentioned before, secure hashing algorithms are extremely sensitive to input which is illustrated in table 3.1.

Table 3.1. Illustration of sensitivity of MD5 to small input changes

Traditional MD5

Input	**Output**
"ISAAC SANTA 19216811"	"b001e3558145b1ca7aa139a63239fa83"
"ISAAC SAMTA 19216811"	"eed6e6e553226f989884a0c4af973bd9"
"ISAAC SAMRA 19216811"	"86979970e1efab6d16e481e2089fa132"

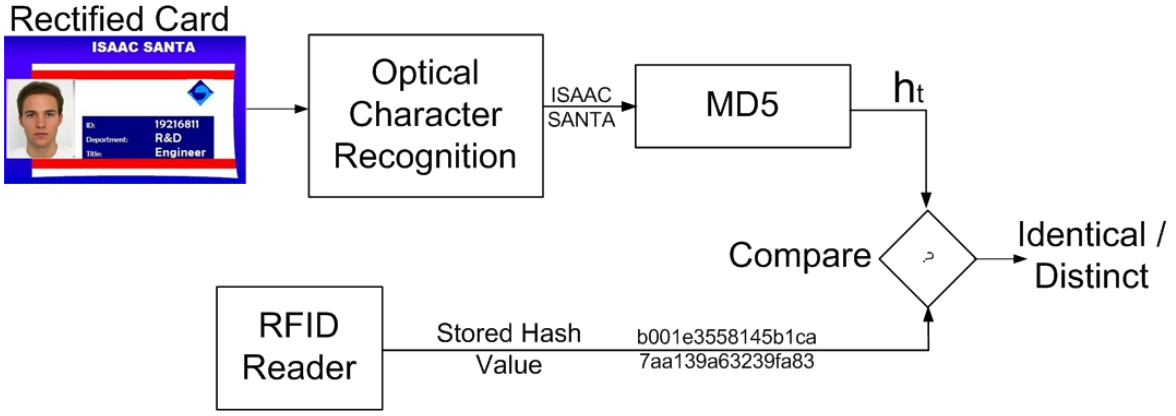


Figure 3.1. Illustration of a traditional MD5 hashing and verification

That means, traditional text hashing schemes in verification systems based on input recognition (scanning, image capturing etc.) suffers from mentioned sensitivity. Because, in any step of practical data normalization, even if one bit error occurs, the output hash will be completely different, hence the verifier will misclassify an identity card. In other words, even if extracted text (or any related information) matches with original text over 99.99 percentage, hash values will completely be different and this will cause a misclassification. This kind of verification is quite inefficient. Considering the need for a resistant hashing scheme especially for systems that collect information (i.e. text) through physical noisy media (camera, scanner, etc.), we have developed a novel text hashing algorithm. In this algorithm, instead of hashing the whole text for once as in figure 3.1, a new set $\mathbf{S}$ that consists several texts based on input text is formed and each element is hashed separately.

Suppose $\mathbf{I} = \{c_1, \ldots, c_r\}$ is a set whose elements (c_i's) are the characters of original text. To give an example, suppose the input text is "LOIS LAGRANGE",

then

$$\mathbf{I} = \{L, O, I, S, L, A, G, R, N, A, G, E\} \tag{3.1}$$

Next suppose that $\mathbf{S} = \{S_1, \ldots, S_n\}$, where $S_i = \{c_{i_1}, \ldots, c_{i_m}\}$ and i_k is determined by a function g, that depends on both public and private key of issuer. Distribution of i_k is the same for both verifier and issuer side with function g. For $m = 20$, we obtain S_i as follows

Table 3.2. Example of elements of set **S**

S_1	GRA	S_{11}	GAL
S_2	OLE	S_{12}	LAR
S_3	***SIE***	S_{13}	ERI
S_4	RAN	S_{14}	***GIR***
S_5	EGN	S_{15}	AER
S_6	GRA	S_{16}	LAL
S_7	LGR	S_{17}	RGL
S_8	GRN	S_{18}	GLR
S_9	LAN	S_{19}	ERA
S_{10}	***LSI***	S_{20}	LSE

In table 3.2, the whole elements of **S** is presented. In this new algorithm, all the elements of **S** is hashed using traditional MD5 algorithm in issuer side. In verifier side, after extracted characters are segmented and recognized, set **S** is performed with the public key of issuer and function g. Next, similar to the procedure in issuer side, all elements of **S** is hashed using traditional MD5 algorithm. At the end we compare output hash values at issuer and verifier side and obtain a score.

The great point here is that, even if a character is misclassified, not all the ele-

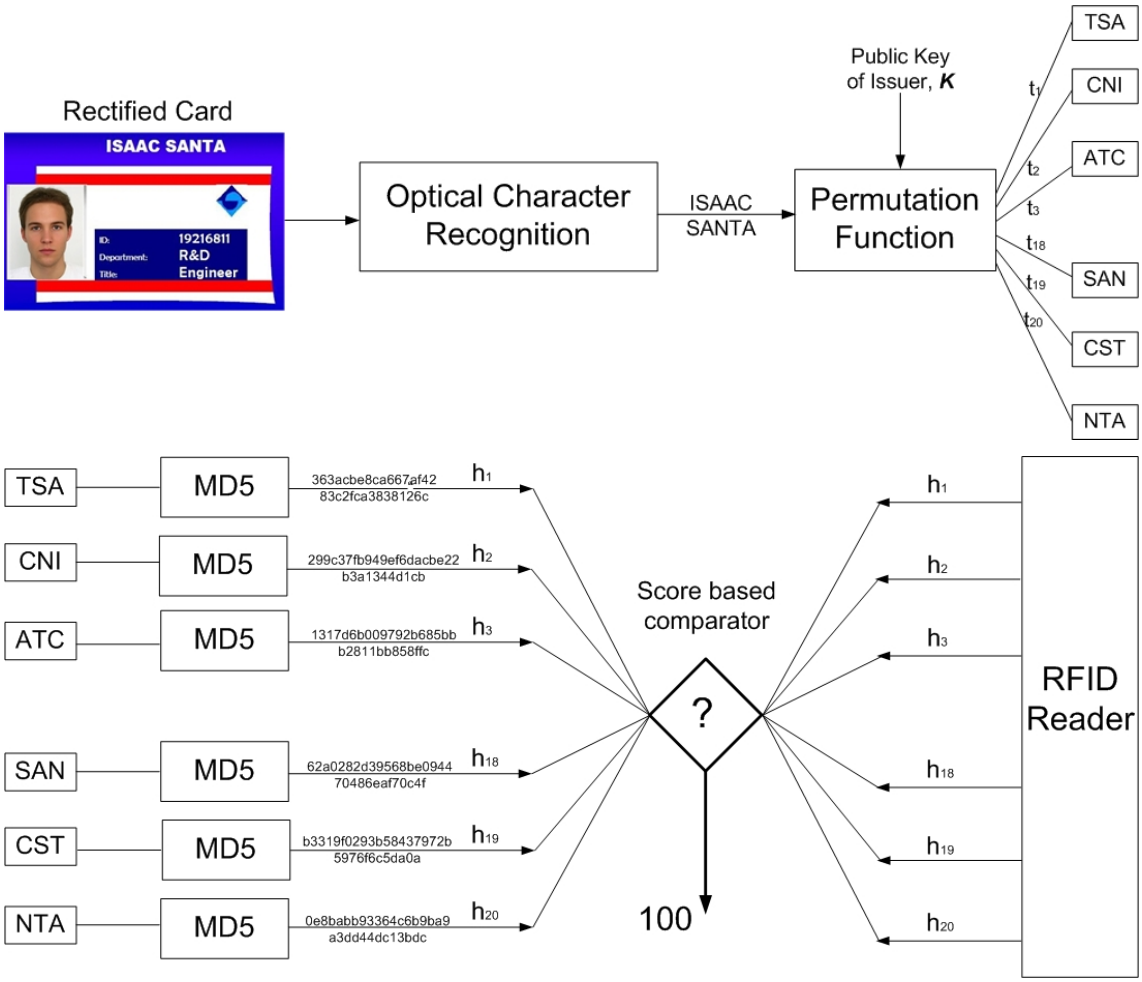


Figure 3.2. Illustration of novel hashing and verification algorithm

ments of **S** is affected. For instance, suppose the letter I is misclassified (Misclassified elements of **S** is bold and italic in table 3.2. Only 3 elements of **S** suffers from misclassifying. Notice that, 85% hash value is still correct. Since the set **S** is formed with the public key cryptography, system is not affected in terms of security. In addition to that, if desired, this algorithm is easily bypassed. Since the algorithm is score based and tolerant to small recognition errors, in case the score threshold is chosen as 100%, the algorithm is bypassed.

3.3. DIGITAL SIGNATURES

A *digital signature* of a message is a message digest dependent on some secret only known by the signer and on the content of the message being signed. Digital signatures

are widely used in information security, including authentication, data integrity, and non-repudiation. The first method used as a digital signature is RSA signature scheme, which remains today one of the most practical techniques available. More information on digital signatures is available in [12].

3.3.1. RSA

RSA signature is the first practical public-key encryption and digital signature scheme discovered by Rivest, Shamir, and Adleman in 1978. RSA is widely used in information security and is believed to be secure provided that long keys are used. The RSA signature signature scheme is a deterministic digital signature scheme which provides message recovery. Therefore, a priori knowledge of the message is not required for the verification algorithm.

3.3.1.1. Key Generation for The RSA Signature Scheme. Each entity (issuer and verifier) creates an RSA public key and a corresponding private key. Each entity should do the following:

- Generate two large distinct random primes p and q, each roughly the same size.
- Compute $n = pq$ and $\phi = (p-1)(q-1)$.
- Select a random integer e, $1 < e < \phi$, such that $gcd(e, \phi) = 1$.
- Use the extended Euclidean algorithm [12] to compute the unique integer d, $1 < d < \phi$, such that $ed \equiv 1 \pmod{\phi}$
- A's public key is (n, e); A's private key is d.

3.3.1.2. RSA Signature Generation and Verification. Entity A signs a message $m \in M$. Any entity B can verify A's signature and recover the message m from the signature.

1. *Signature generation.* Entity A should do the following:
   (a) Compute $m' = R(m)$, an integer in the range $[0, n-1]$.

(b) Compute $s = m'^d \bmod n$.

(c) A's signature for m is s.

2. *Verification.* To verify A's signature s and recover the message m, B should:

(a) Obtain A's authentic public key (n, e)

(b) Compute $m' = s^e$

(c) Verify that $m' \in M$; if not, reject the signature.

(d) Recover $m = R^{-1}(m')$.

3.3.1.3. Demonstrating RSA Signature. In this project, we make use of RSA signature scheme to be a message recovering scheme which is of great importance. In issuer side, we generate a message m, and sign it with the private key of the issuer, obtaining a digital signature, s. Issuer embeds two information: message m and digital signature s to RFID tag. In fact, embedding message m is just to check the authenticity of digital signature s (A malicious attack may change the content of signature). RSA signature encryption and decryption schemes are illustrated in figure 3.3.

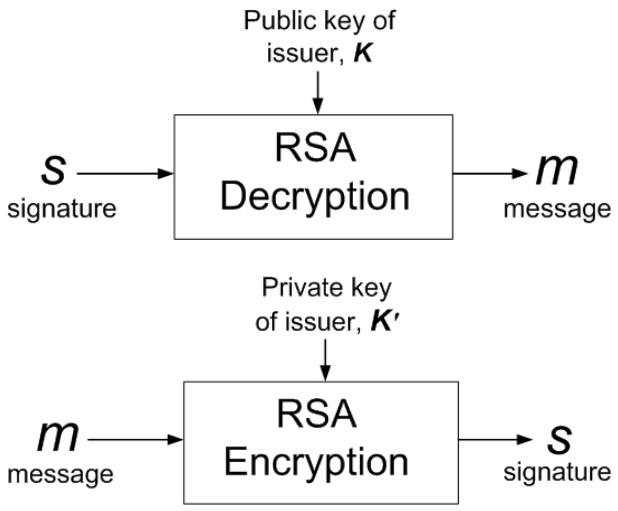


Figure 3.3. Illustration of RSA encryption and decryption

4. ROBUST PERCEPTUAL IMAGE HASHING

4.1. Perceptual Robust Image Hash Functions

A perceptual image hash function maps an image to a short string based on the image's appearance to the human eye [13]. In particular, two images that look the same to the human eye should map to the same perceptual hash value, even if the images are different in digital representation sense. This is the main difference between a perceptual hash and traditional cryptographic hashes, such as SHA-1 and MD-5 [12]. SHA-1 and MD-5 hashes are extremely sensitive to the input data, i.e., a one bit change in the input produces a great change in output.

Perceptual image hash functions are useful to compare and search of images in large databases in which several perceptually same versions of an image may exist. Further need for such functions arise in *image authentication*. Due to nature of digital media, digital media can be tampered and hence there exists a need to be able to verify the content of the media to ensure its authenticity. However, this approach is expected to be sensitive to any malicious modification while being able tolerate to incidental modifications such as JPEG, histogram equalization, image enhancement. In other words, perceptual image hash functions should reflect the perceptual features of the image while being sensitive to small malicious modifications.

Let $\mathcal{I}$ denote a set of images with finite cardinality. Let $\mathcal{K}$ denote private key space. The perceptual robust image hash function then takes two inputs: an image $I \in \mathcal{I}$ and a private key $K \in \mathcal{K}$ to produce fixed-length hash value $h = H(K, I)$. Let $I_{ident} \in \mathcal{I}$ denote a perceptually similar image to I. Similarly, let I_{diff} denote a perceptually different image from I. Finally, let p_1, p_2 satisfy $0 < p_1, p_2 < 1$. Then a robust perceptual image function has the following desired properties:

- **Perceptual Robustness:** The probability of distances of hashes of two perceptually similar images being below a threshold is highly possible.

$Probability(\| H(I,K) - H(I_{ident},K) \| \leq \tau) \geq 1 - p_1$, for a given p_1

- **Sensitivity to perceptually distinct images:** The probability of distances of hashes of two perceptually different images being above a threshold is highly possible.
  $Probability(\| H(I,K) - H(I_{diff},K) \| \geq \tau) \geq 1 - p_2$, for a given p_2
- **High entropy of the hash:** The computed hash $H(I_0,K)$ for a fixed image I_0 with varying key K should be unpredictable.
  $Probability(H(I,K) = t) \approx \frac{1}{2^r}$, $\forall t \in \{0,1\}^r$ where r is fixed length of hash.

The rest of this chapter is the application, review and comparison of existing hashing approaches.

4.2. Non-Negative Matrix Factorization (NMF) Based Hashing

4.2.1. Background

In this section, a brief introduction and known results of NMF will be introduced.

Given a non-negative matrix V of size $m \times n$, NMF algorithms seek to find non-negative matrix factors W and H such that

$$V \approx WH, \text{ where } W \in \mathcal{R}^{m \times r} \text{ and } H \in \mathcal{R}^{r \times n},$$

or equivalently, the columns $\{v_j\}_{j=1}^n$ are approximated such that

$$v_j \approx Wh_j, \text{ where } v_j \in \mathcal{R}^m \text{ and } h_j \in \mathcal{R}^r.$$

For the class of full (nonsparse) matrices, this factorization provides a reduction in storage whenever the number of vectors r, in the basis W is chosen such that $r < \frac{mn}{m+n}$. In [14] the problem of choosing r for NMF is emphasized to be considerably more

cloudy than looking at the decay of the magnitudes of the eigenvalues of the data, as is done in traditional rank reduction techniques. In practice, r is usually chosen such that $r \ll min(m, n)$.

NMF has several benefits on robust image hashing. Those and more is explained in [14]. In this study, we use the proposed algorithm (NMF-NMF-SQ), the work of V. Monga *et al.* in [14]. In NMF-NMF-SQ algorithm, the images are viewed as matrices and the goal of hashing as a randomized dimensionality reduction that retains the essence of the original image matrix. For security reasons, NMF is applied pseudo-randomly chosen image regions. The work is motivated by the fact that standard rank reduction techniques such as the QR, and Singular Value Decomposition (SVD), produce low rank bases which do not respect the structure (i.e. non-negativity for images) of the original data. The experiments revealed that the NMF based hash possesses excellent robustness under a large class of perceptually insignificant attacks while significantly reducing collision probability for hash values obtained from perceptually distinct images.

4.3. SVD Based Hashing

4.3.1. Introduction

In this scheme, images are treated as matrices and a hash is formed based on coefficients in a low-rank matrix approximation of the image. Particularly, the well known Singular Value Decomposition (SVD) is used to generate the low-rank approximation. The reason why the authors [15] chose SVD is that is the best lower-dimensional approximation to a matrix in the sense of Frobenius norm are produced by the first K singular vectors and the corresponding singular values.

A two-stage cascade application of SVD (instead of a single one) is based on empirical observations that it serves to enhance robustness under perceptually insignificant attacks while introducing more randomness. More than two stages are rejected because experiments revealed that it would severely affect the perceptual qualities of the hash.

The major gains of SVD based hashing [15] are in increased robustness to geometric attacks, e.g. rotation up to 15 degrees and 20% cropping.

However, when a low-rank approximation is obtained via SVD, the orthogonality constraint imposed by construction of SVD translates into a holistic representation of the image. Since the local features are ignored, this results in a high misclassification rate. This problem can be avoided by choosing the rank of the approximation to be large, but this time robustness decreases while length of the hash increases to practically unacceptable levels.

The SVD-SVD hashing algorithm [15] follows:

1. Given an image, pseudo-randomly select p subimages $A_i \in R^{m \times m}$, $1 \leq i \leq p$.
2. Obtain the SVD of each subimage:

$$A_i = U_i S_i V_i^T,$$

   where U_i, V_i are the $m \times m$ real left and right singular vector matrices and S_i is the real $m \times m$ diagonal matrix consisting of the singular values along the diagonal. Collect the first left and right singular vectors that correspond to the largest singular value from each subimage. Let $\Gamma = \{\vec{u}_1, \ldots, \vec{u}_p, \vec{v}_1, \ldots, \vec{v}_p\}$, where $\vec{u}_i$ (respectively $\vec{v}_i$) is the first left (respectively right) singular vector of the i-th subimage.
3. Pseudo-randomly form a smooth image J from the set Γ: Given a pseudo-randomly selected initial singular vector, form J by selecting and replacing subsequent vectors from Γ such that the next chosen vector is closest to the previous vector in L_2 norm sense. Hence after $2p$ steps all the elements of Γ are pseudo-randomly re-ordered and J (of size $m \times 2p$) is formed.
4. Re-apply SVD to q subimages B_i's such that B_i's are pseudo-randomly picked from J.
5. The hash vector $\vec{h}$ is formed by retaining the corresponding set of the first r left and right singular vectors from each B_i, i.e., let $f_i = \{\vec{u_{b1}}^i, \ldots, \vec{u_{br}}^i, \vec{v_{b1}}^i, \ldots, \vec{v_{br}}^i\}$

and $\vec{h} = \{f_1, ..., f_q\}$.

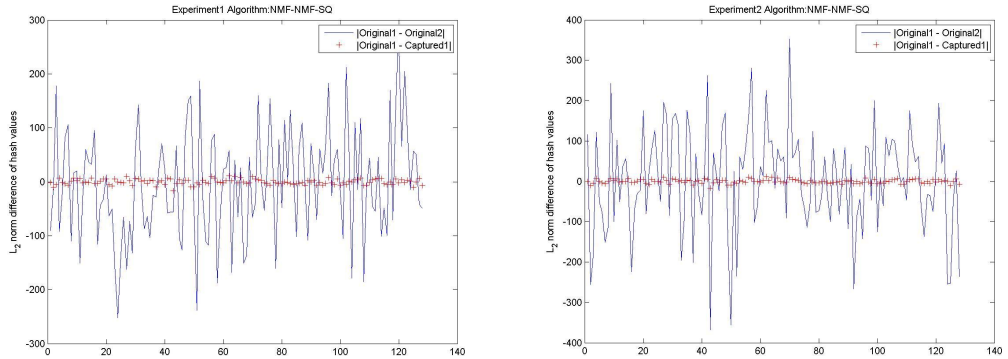
4.4. Experiment Results

Firstly, to illustrate the perceptual robust image hashing ability of NMF-NMF-SQ algorithm, we will apply this algorithm on 3 face images. Those images belong to two people as shown in figure 4.1.

(a) Original (b) Captured (c) Original

Figure 4.1. Face images of person 1 and 2.

We have three distinct pictures and two of them are perceptually the same. We will apply NMF-NMF-SQ algorithm to those pictures (captured picture of the same person is changed in each comparison) and compare the distances among each other.



(a) (b)

Figure 4.2. Componentwise difference between the NMF-NMF-SQ hash vectors of the face image of person1, its captured version and person2.

Figure 4.4(c) and 4.4(d) are the results of two experiments. Notice that the energy of the difference between the original face and its captured version is much

smaller than the energy of the difference between two different face images. This validates the excellent robustness properties of NMF while avoiding misclassification.

In figure 4.2, we have observed excellent robustness properties of NMF-NMF-SQ algorithm in randomly chosen single images. Now, we will observe average L2 norm between hash vectors of original face images and captured face images (original and captured faces belong to the same person, namely they are perceptually the same) across 150 different images. L2 norm between hash vectors of the original and a completely different face image (belongs to different person) are also plotted for comparison purposes. Note that these are average values of different realizations (angle, distance). Notice that the separation between norm differences is quite large and there is no overlap between the red and the blue portions. However, also note that, since these are the average values, no overlap between red and blue portions actually does not mean no misclassification.

In table 4.1 number of face misclassifications in the related experiments is presented. The parameters, distance and angle are defined same as in section 2.3.3.1. 150 face images are tested in experiments; in order to obtain ROC curves, it is sure that we need images in the order of 1000.

Table 4.1. Number of face misclassifications in verification experiments

Number of Face Misclassification(s) in Verification Experiments			
Method	**Distance (cm)**	**Angle (degrees)**	**# of Misclassification**
NMF-NMF-SQ	15	0	0
NMF-NMF-SQ	15	30	0
NMF-NMF-SQ	30	0	0
NMF-NMF-SQ	30	30	1
SVD-SVD	15	0	1
SVD-SVD	15	30	2
SVD-SVD	30	0	2
SVD-SVD	30	30	2

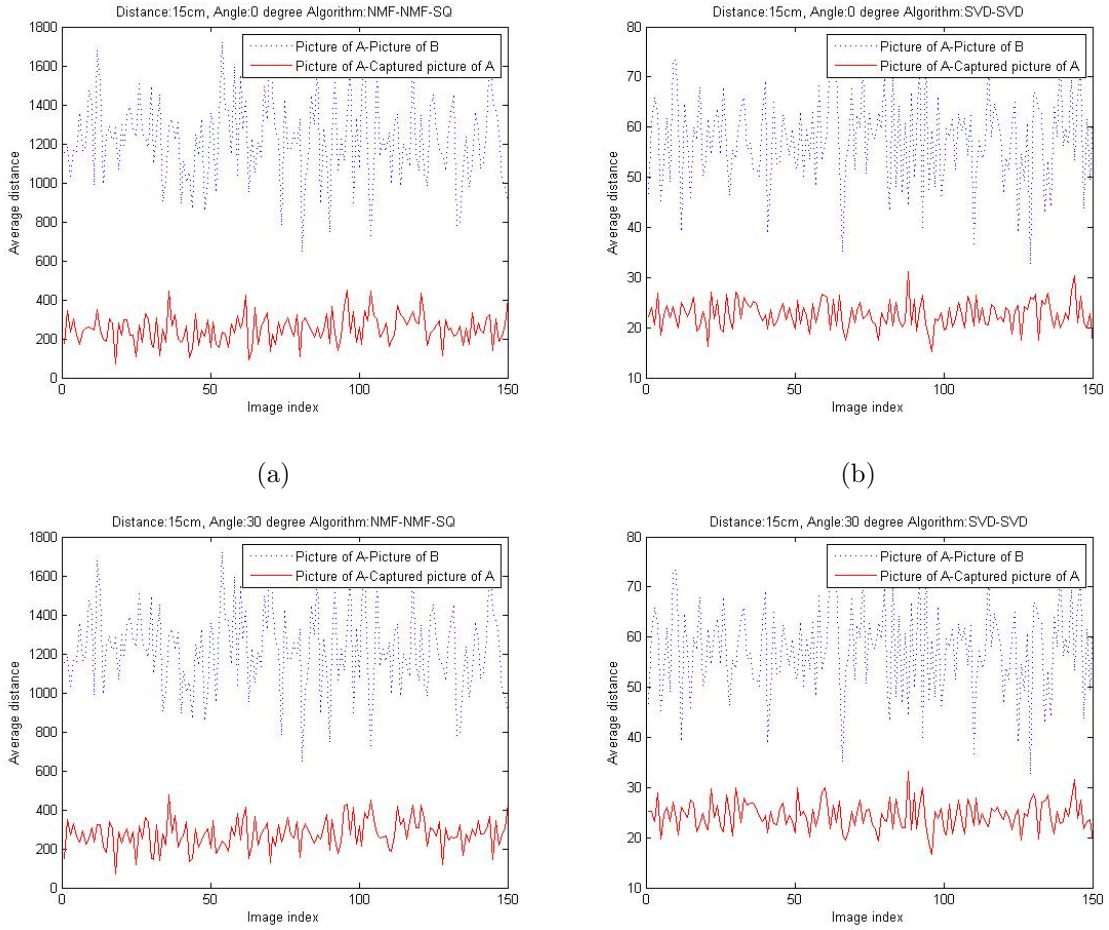


Figure 4.3. L2 norm between difference hash vectors of the original face images and captured face images (belong to same person, A) across 150 images. L2 norm between hash vectors of the original face image (belongs to person A) and a completely different image (belongs to person B), distance 15cm

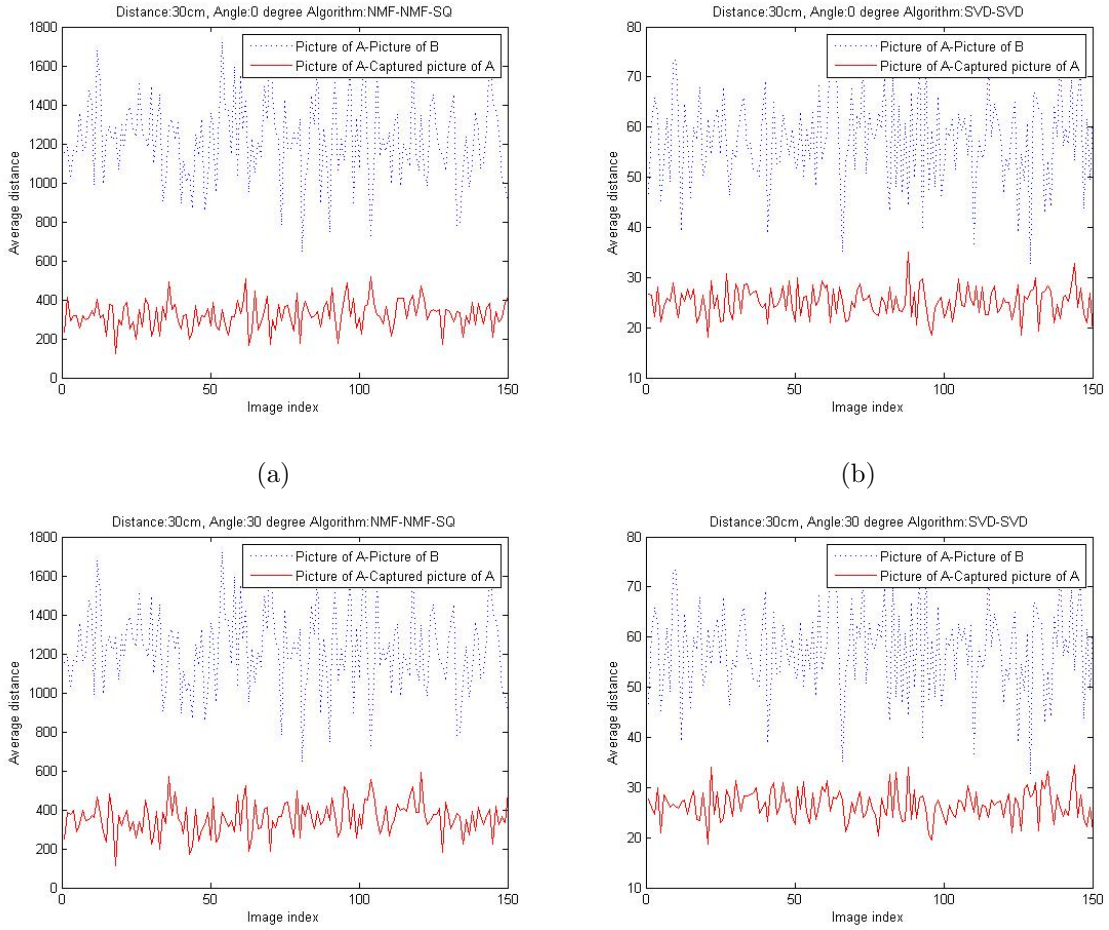


Figure 4.4. L2 norm difference between hash vectors of the original face images and captured face images (belong to same person, A) across 150 images. L2 norm between hash vectors of the original face image (belong to person A) and a completely different image (belong to person B), distance 30cm.

5. CONCLUSIONS

We introduced a new, cryptographically secure, RFID based (contactless) identity verification system. The elements required to issue (create) an authentic identity card are a colorful PVC or paper printer an HF-RFID and a RFID writer/reader. Similarly the required elements to verify an identity document are a simple USB camera, a DSP based intelligent device, an LCD, and an HF-RFID reader.

Typically an identity document that is created by the system we have proposed costs less than 1$. The reason for that is, identity document requires no special and complicated printing techniques and papers. Similarly, a verifier and issuer cost on the order of 100$ which is much cheaper than other verification systems with comparable level of security. A typical identity card with barcode costs $10-30$ whereas a lasercard reader costs about 2400$ [4].

To rectify captured identity card via camera, projective transformation together with synchronization points is used. Higher number of synchronization points than we need (in mathematical sense) provides a way determine projective transformation parameters more robustly.

We have used k-Nearest Neighbor classifier for text recognition. We have an sub-optimal value k and observed the behavior of classifier in different environmental combinations. The recognition rate in optimal environmental conditions is 98.88% and could be enhanced with different combinations of training patterns and feature sets.

The novel text hashing scheme that we have proposed provides score based verification. In traditional text hashing and verification systems, the result is either authentic or unauthentic. The reason for that is output hash value is compared with the hash value of original text. When there is a misclassification error in one of the input characters, the result hash value will completely be different.

Using RSA digital signature scheme provides a secure way to deliver message from issuer to verifier. With the use of RSA digital signature, the exact value of message can be retrieved and compared.

Perceptual robust image hashing methods, especially NMF-NMF-SQ, show that face authenticity verification can be performed with large separation. Among over 1000 different comparisons, NMF-NMF-SQ has no misclassification error. The separation between distinct face images can be even further enhanced using a different secret key in each image.

REFERENCES

1. R.L. Van Renesse. Optical Document Security. Artech House, 1998.

2. R.L. Rivest, et al. A method for obtaining digital signatures and public-key cryptosystems, CACM, Vol.21, No.2, pp.120-6, 1978.

3. A.J. Menezes, et al. Applied Cryptography. CRC Press, 1996.

4. D. Kirovski, N. Jojic, and G. Jancke. Tamper-Resistant Biometric IDs. Information Security Solutions Europe, pp.160-75, 2004.

5. Rafael C. Gonzalez, Richard E. Woods, Steven L. Eddins, Digital Image processing using MATLAB, Upper Saddle River, N.J, Pearson Prentice Hall, 2004.

6. Todd K. Moon and Wynn C. Stirling, Mathematical Methods and Algorithms for Signal Processing, Prentice-Hall, 2000

7. Per Christian Hansen, The truncated SVD as a method for regularization, BIT, pp. 534-553, 1987

8. Berne Jähne, Practical Handbook on Image Processing For Scientific and Technical Applications, Boca Raton, CRC Press, 2004

9. Handley, J.C. Xerox Corp., Webster, NY; "Improving OCR accuracy through combination: a survey", Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference

10. Duda, Richard O. and Hart, Peter E., Pattern Classification and Scene Analysis, A Wiley-Interscience Publication, 1973

11. Cover, T.M. and Hart, P.E., Nearest Neighbor Pattern Classification, IEEE Transactions on Information Theory, Volume IT-13(1), pp.21-27, 1967

12. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, 1998.

13. Vishal Monga, Arindam Banerjee, and Brian L. Evans, "Clustering Algorithms for Perceptual Image Hashing", Proc. IEEE Work. on Digital Signal Processing, pp. 283-287, Aug. 1-4, 2004

14. V. Monga and M. K. Mihcak, Robust Image Hashing Via Non-Negative Matrix Factorizations, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2006.

15. S. S. Kozat, K. Mihcak, and R. Venkatesan, "Robust perceptual image hashing via matrix invariances", Proc. IEEE Conf. on Image Processing, pp. 3443-3446, Oct. 2004