A Secure Face Recognition System for Mobile-devices without The Need of Decryption

Shibnath Mukherjee
Deloitte & Touche Consulting India Pvt. Ltd
shibnath@gmail.com

Zhiyuan Chen
University of Maryland Baltimore County
zhchen@umbc.edu

Aryya Gangopadhyay
University of Maryland Baltimore County
gangopad@umbc.edu

Stephen Russell
George Washington University
russells@gwu.edu

Abstract

Face recognition technology has received much attention due to its application in defense and crime prevention. In such applications, there is great need to incorporate face recognition technologies onto mobile devices to allow on-the-spot field usage. However there are four major problems that need to be solved, namely the limited storage and processing power of the mobile device, connection instability, security and privacy concerns, and limited network bandwidth. Existing methods do not solve all the problems. This paper addresses all of the above problems holistically by proposing a novel approach. The core of the approach is a DCT-based compression method. This method has high compression ratio such that the compressed image database can be easily stored at a mobile device. Further, face recognition algorithms can be run directly on the compressed database without decompression, which enables on-the-spot field usage. The overhead of network transfer is also greatly reduced due to compression. The security and privacy issue is addressed by pruning most DCT coefficients of images and by a random permutation protocol. As a result, the reconstructed images are not visually recognizable even if the permutation is known. Additional security can also be provided by encrypting the coefficients for network transfer. The system has been implemented on a commercially available general purpose PDA-phone and experimental results demonstrate the potential of the proposed solution.

Keywords: Face recognition, mobile computing, security, privacy, knowledge management.

1. Introduction

Automatic identification of facial mugshots has become an important component of many systems used by the military and law enforcement to fight terrorism and other crimes. Adapting these systems to mobile devices (that might be carried by field agents on duty) will allow on-the-spot *field usage* and generate significant potential benefits. Though much work has been done in the core areas of developing robust face recognition algorithms [22, 4], very little work has focused on the area of making those technologies feasible in the domain of mobile computing. There are four issues that need to be addressed.

The first issue is the limited processing and storage capabilities for mobile devices. For a majority of the devices used, available storage is too small for the image database. Further, given the computational complexities of the face recognition algorithms, the processing capabilities of the devices are insufficient to handle the expensive calculations on large datasets. An obvious solution is to implement a thin-client architecture for the task where processing and storage are done on a server and results and input are communicated across the network. However this leads to a second problem: thin client architecture demands constant network connectivity to operate thereby constraining the field usage capability severely. Along with this is the threat of constant exposure to interception attacks.

Both problems could be avoided if the database is stored locally with some compression scheme having a very high compression ratio. Unfortunately this leads to the third problem: the sensitivity of the data being dealt with. Typically, the mugshot database used for anti-terrorism or crime prevention are highly classified. Storing them on mobile devices given to field personnel increases the risk and opportunity of compromise through misplacement and theft, given the small size of the devices.

The fourth problem is network bandwidth. In most security related face recognition applications, keeping the mugshot database updated with the latest information is critical. However, limited traffic handling ability prevents fast transfer of voluminous image data over mobile networks, creating a bottleneck for updates.

Commonly used encryption schemes achieve security but fail to compress the database. Further, these schemes do not allow face recognition to be carried out in the encrypted domain. Thus the whole encrypted database has to be decrypted, which is very expensive for mobile devices with limited processing power. Moreover, if cracked, any fixed encryption scheme will give away every bit of data encrypted with it immediately. For example, in case of the mugshot database, all the images will be compromised.

This paper addresses all of the above problems holistically by proposing a novel approach. The core of this approach is a DCT-based compression method which applies Discrete Cosine Transform to images and then prunes most DCT coefficients. As a result, the remaining coefficients are sufficient to provide accurate face recognition, but are insufficient to reconstruct visually recognizable images. This approach also uses a random permutation protocol to enhance the security. As a result, attackers can not reconstruct recognizable images from stolen devices, even if they know the permutation order. If necessary, the DCT coefficients can be also encrypted to prevent network interception attacks. Another benefit of this approach is that the face recognition task can be carried out without decompression or decryption of the whole database, which enables on-the-spot field usage. Further, the compression ratio is very high such that the storage requirement and the overhead of network transfer are greatly reduced. Initial experimental results demonstrate the benefits of the proposed solution.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 presents our approach. Section 4 discusses the security aspect of our approach and the worst case security. Section 5 describes a system implementation and experimental evaluation.

2. Related Work

Face recognition algorithms: There has been a rich body of work on face recognition algorithms. These algorithms typically first extract some important features from images. Principal component analysis (PCA) is used for this purpose [22]. Independent Component Analysis (ICA) [4], Linear Discriminant Analysis (LDA) [10], Evolutionary Pursuit (EP) [15], Kernel-based methods [25] as well as Bayesian Learning methods [17] are also proposed. Of all these methods, variants of the PCA and ICA based methods are the most well known, well established and most commonly used. Some authors [12] suggested the use of Discrete Cosine Transform feature vectors or different types of wavelets such as Gabor [19] instead of PCA. As reported, DCT based schemes although being much less expensive and data independent, work almost as good as the PCA on highly correlated image data.

Once features are extracted, a variety of classification techniques such as hidden markov models, neural nets, support vector machines, nearest neighbor match etc. are used for the recognition process. Of all these, the nearest neighbor approach [9, 7] is the simplest, highly efficient, and most commonly used one.

Face recognition on mobile devices: [13, 23] and [2] proposed various thin-client architectures for face recognition on mobile devices. [13] also discusses an extension of the basic technology for authentication purposes on mobile devices. However a major drawback of these solutions is the thin client aspect that reduces field usage severely as discussed earlier. These solutions assume delegating data storage and processing to a separate server and not the device. In other words, the device is not made even slightly independent in performing the task. Every time a face recognition task is required, the server needs to be contacted over the network. Thus the system will stop functioning at places without stable network connection, which may be common depending on the field of usage. A simple example might be an agent operating the device at a remote village. Another major challenge with these schemes is the security aspect of the data which turns out to be very important for defense and crime prevention contexts. The requirement for constant communication exposes the data in transit to a very high risk of interception attacks.

Privacy preserving data mining techniques: There has been a rich body of work in the area of applying Secure Multi-party Computation (SMC) [3] techniques to distributed mining. Please refer to [8] for a survey. However this paper focuses on face recognition on mobile devices, which does not include multiple parties other than the server and the device.

Another group of researchers focused on data randomization approaches through additive perturbation [1], multiplicative perturbation [14], random projections [16], and rotations [6] to preserve privacy. [18] discusses a DCT based novel method for data compression and secure distance based mining of correlated data in general.

Gross et al. [21, 11] explores the privacy concerns in face recognition in general. The work incorporates the concept of a well known technique called *k-anonymity* [20] into the domain of face recognition. The primary focus is to make sure that k images in a dataset look similar so that identity-wise, none of the images could be tracked to a single person by machine recognition or human inspection. However this violates the basic motivation for doing a face recognition task in the first place. Once the images are k-anonymized, even an authorized user also will not be able to distinguish between an image and its $k-1$ duplicates in the data. This paper instead draws its motivation from a more practical viewpoint. In order for the system to be useful, legitimate users need to be able to identify the images properly while

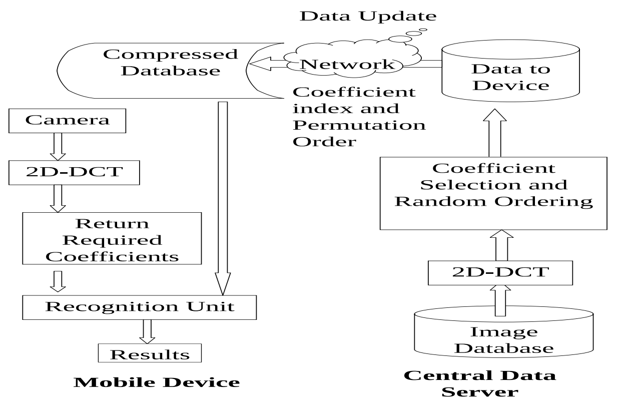

Figure 1. Architecture of our approach

non-legitimate users should not be able to do so. Further if the database is exposed, it would not be possible for an unauthorized user to reconstruct useful images. To the best of knowledge of the authors, this is probably the first attempt ever made to build such a secure system in context of face recognition on hand-held devices.

3. Our Approach

Figure 1 shows the architecture of our approach. Next we describe the processes at the server and client side.

```
Algorithm 1:  DCT-H(Data $D$, Parameter $\mu$)
(1)  for each image $i$ of $D$
(2)   DI(i)=2D-DCT(i)
(3)  end
(4)  for each row $i$ of $DI$
(5)   Mark the $\Delta$ AC coefficients with
      highest energy
(6)  end
(7)  for $j$-th coefficient
(8)   $frq(j)$ = number of rows with
      $j$-th coefficient marked
(9)  end
(10)  select coefficients with $\mu$
      highest $frq$
(11)  permute the selected coefficients
```

Figure 2. Compression Algorithm.

Server side: The server stores the original image database, and generates a compressed database. Figure 2 shows the compression algorithm DCT-H. This algorithm has two steps. In the first step, the images in the database are transformed using two dimensional Discrete Cosine Transform (DCT-II). The DCT coefficients of each image are stored in a row vector by concatenating DCT coefficients from each row of the image side by side. Note that DCT being an orthogonal transform, the Euclidean distances between images remain unchanged after the transform.

In the second step, the method selects μ DCT coefficients which have high energy in a majority of instances. This set of coefficients will capture major chunk of the data variance and hence preserve the Euclidean distances between images with nominal error. Thus the nearest neighbor face recognition algorithms which use distances can generate accurate results over such datasets.

The selection is done by first mark the μ coefficients with the highest energy for each image. We also exclude the DC coefficient because it represents brightness and is not useful for face recognition. The algorithm then counts for each coefficient, the number of images having that coefficient marked. The μ coefficients with the highest count will be selected. Thus the algorithm selects coefficients that have high energy in the most of the images. Finally, these coefficients are permuted randomly.

μ is decided by the user based on required tradeoff between compression and accuracy of recognition. Experimental results in Section 5 show that a very small μ can be used to achieve high accuracy of recognition and high level of compression at the same time.

The random permutation can be considered as a cheap alternative to encryption. It reduces the risks of interception attacks because the permutation order needs to be known to reconstruct the images from coefficients. The details will be discussed in Section 4. The indexes of these coefficients and the permutation order are then encrypted using symmetric key encryption with a public key based key exchange protocol. The selected coefficients and encrypted indexes and permutation order will be loaded into the mobile device.

Our approach can also handle incremental updates to the image database. The compression algorithm does not need to recompute the DCT coefficients of the existing images in the database. The server also does not need to send the whole compressed database. The details are not shown due to space constraints.

Client side: The device stores the compressed image database. The client also stores the selected coefficient index and permutation order in encrypted form. When the camera in the device captures a field photograph of a test subject, the face recognition algorithm can be run locally on the handheld and directly over the stored compressed image database. This process consists of the following two steps.

At the first step, the mugshot of the test subject is transformed with a DCT-II, and reaped to a single row vector. The device will also decrypt the coefficient index and permutation order, and only retain those coefficients retained in the image database and permute them in the same order as the images in the compressed database.

This step can be done efficiently on a mobile device for three reasons listed below. First, the DCT transform vectors are data independent and can be generated on the fly. Second, there exists special plug-in hardware for performing DCT extremely fast. Third, there is no need to generate all DCT coefficients. Only the set of coefficients that are retained in the compressed image database needs to be generated on the fly.

At the second step, the generated, ordered coefficients are fed to the face recognition unit. In this paper, 1-nearest neighbor matching is used to for the recognition task. Note that the mugshot image to be matched is essentially compressed and permuted in the same way as the images in the database. Thus there is no need to decompress the compressed database. However since there is barely any loss of data variance, the nearest neighbor match approach will run accurately as illustrated in the experimental results presented in Section 5.

Running the face recognition algorithm on compressed data is also much cheaper than running it on uncompressed data because the compressed images are much smaller (only having μ coefficients each) than the original images.

4. Security and Privacy Aspects

This paper mainly considers the type of attacks where the device is stolen and someone tries to recover images stored on the device. The protection comes from the pruning of most DCT coefficients and the random permutation protocol discussed in Section 3. The key observation is that it is difficult to invert the DCT coefficients to get back the original data without knowing the random permutation order in which DCT-H shuffles the coefficients. Unlike other transforms such as principal component analysis, it is difficult to decide the order based on values of DCT coefficients (e.g., the second coefficient is not necessary larger than the third).

The worst case scenario occurs when a third party guesses the coefficient indexes and the permutation order correctly. Assume P be the probability of occurrence of the worst case. Also assume the image size is $u \times v$ and $u \times v = n$. If μ denotes the number of coefficients chosen, then P is given by

$$P = \frac{1}{\mu! \binom{n}{\mu}} \tag{1}$$

The size of image is typically pretty big while the number of coefficients selected (μ) is pretty small (experimental results in Section 5 reports that $\mu = 10$). The denominator in Equation (1) in such case will be big enough making the probability for the worst case scenario extremely small.

In the worst case, the permutation protocol is cracked. Given a sample image one can check out if it is in the compressed database through a nearest neighbor match. Actually there is no way to prevent this unless we encrypt the images on the device, which will incur significant computational overhead resulting from the decryption now required for the face recognition task. However it will be impossible to visually inspect any other images in the database.

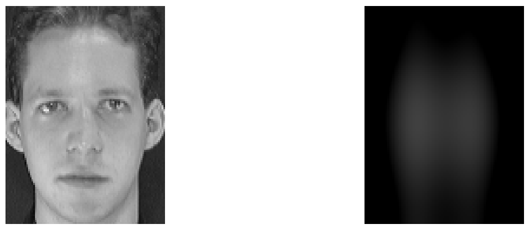
(a) Original (b) Reconstructed

Figure 3. Example images

The number of coefficients required for an accurate recognition task is much smaller than the number of coefficients needed to reconstruct an image that can be recognized by human being. Thus the images reconstructed from the compressed database have very poor quality (one can barely tell it is a face). For example, Figure 3 (a) shows an original image and Figure 3 (b) shows an image reconstructed from 10 DCT coefficients (other coefficients are assumed to be zero). The reconstructed image is clearly not recognizable.

Another type of attack is network interception attack. The random permutation protocol only provides limited protection against such attacks (in the worst case, the order may be known). To protect against such attacks, the server can apply symmetric encryption with public key exchange protocol over the DCT coefficients, before sending them to the device. The device can then decrypt the coefficients. Note that the number of coefficients (μ) is much smaller than the original image size, thus the cost of encryption and decryption will be much lower compared to encrypting and decrypting the original images.

5. Experiments

This section experimentally evaluates the performance of our approach. Section 5.1 describes the setup of the experiments. Section 5.2 reports the results.

5.1 Experiment Setup

Implementation: The server side program is written in Matlab 7.0. The client side program is written in Microsoft Visual Basic .Net 2003 (using the Smart Device Application compile option). The face recognition algorithm is k-nearest neighbor where k=1.

The experiments on the client were conducted over a Palm Treo 700W smart phone, with an Intel PXA272 312 MHz processor, and 88 MB memory (24.45 MB program memory and 62.94 MB storage memory). The device runs Windows Mobile 5 Pocket PC Phone Edition Software. It has Bluetooth v1.2 and EVDO 3G wireless connections. The experimental server was a 3.5Ghz Pentium 4 machine with 4 GB of RAM running Windows XP. Various network environments have been tested, including 14.4 kbps, 56 kbps, 128 kbps, 640 kbps, and 1.5M bps. Since the communication time over the network and processing time at

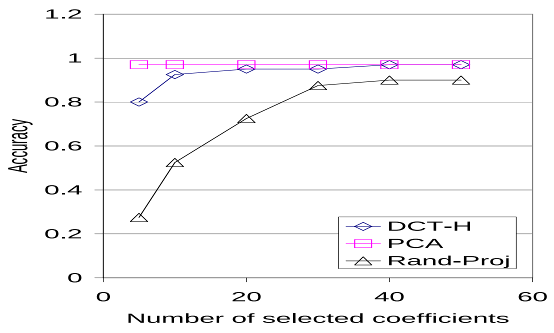

Figure 4. Accuracy of face recognition

the mobile varies in practice, the average of 10 runs were reported.

Dataset: The well-known ORL [5] database of faces is used. The database consists of 400 images. There are ten different images of each of 40 distinct subjects. The size of each image is 112×92 pixels, with 256 gray levels per pixel. Thus each image contains 10304 pixels and unpruned DCT coefficients.

Compression Methods: Our approach uses DCT-H in Figure 2 to compress the image database. Two other compression methods, PCA and random projection, were also implemented to compare against DCT. PCA gives the provable best compression but is more expensive than DCT and is not computationally feasible for mobile devices. Thus in this paper the experiments using PCA was conducted at the server side. Another problem of PCA is that it is easy to decipher the permutation order of selected PCA components because the associated Eigen values are always in descending order. Thus PCA will lead to lower degree of security if even the same random permutation protocol as mentioned earlier is applied on the principle components.

The random projection method [16] multiplies each image with a randomly generated matrix $n \times k$ matrix where n is the image size. The result is a size k vector for each image. As $k < n$, compression is achieved and the distance between original images are kept to some degree.

5.2 Results

Accuracy of Face Recognition: Figure 4 shows the accuracy of face recognition over ORL data (containing 400 images) with 10% as testing data. DCT-H achieves almost the same accuracy as PCA. DCT-H is also much better than random projection because our method better preserves distance between images. Note that unlike our approach, PCA method is not safe (it is easy to find out the order of PCA coefficients).

Storage and processing overhead at devices: Table 1 reports the storage and processing overhead for mobile device when 10 DCT coefficients were used. The processing time

Table 1. Storage and Processing Overhead at mobile

	Storage	Time for DCT	Time for face recognition
Uncompressed	3.7 MB	N/A	N/A
Compressed	14 KB	12.8 sec	0.25 sec

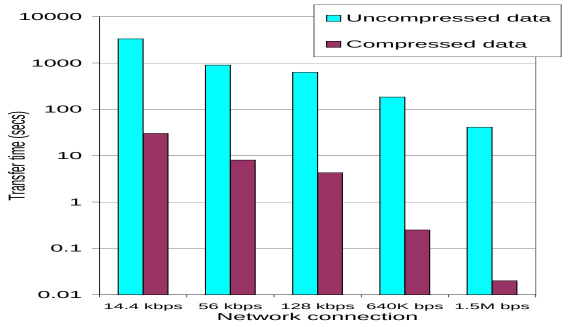

Figure 5. Time to transfer the whole database over various network connections.

is further divided into the time for DCT over the test image and the time for running the face recognition algorithm on the hand-held. The storage overhead for uncompressed image database is also reported. The processing time for direct usage of the uncompressed image database is not available because it is infeasible to run face recognition on the mobile device with it.

The results show that using DCT the image database was compressed to $1/257$ of the original image database. Thus it is easy to fit the compressed image database to a mobile device. Further, it took about 12.8 seconds to generate the required DCT coefficients from a test image. It took about 0.25 second to run the face recognition algorithm (k-nearest neighbor) because each image now only has 10 coefficients. The overall response time is about 13 seconds, which is quite acceptable given the resource constraints of a mobile device. Further special hardware can be used to further reduce the time for DCT transform.

Network transfer time: Figure 5 reports the time to transfer the compressed and uncompressed database over various network connections. The compression method is DCT with 10 coefficients selected. Note that the y-axis is in logarithmic scale. The results show that the time to transfer the compressed image database is orders faster than the time to transfer the uncompressed database. For example, using a 128 kbps network, it took 10 minutes and 30 seconds to transfer the uncompressed database while it took only 4.28 seconds to transfer the compressed database.

Worst case security: In the worst case, an unauthorized party may obtain the compressed database and find out the indexes and correct permutation order of selected DCT coefficients. It can then reconstruct the image from these coefficients by assuming all pruned coefficients having value zero. A typical measure for the quality of the reconstructed images is Peak Signal to Noise Ratio (PSNR) [24]. The higher the PSNR, the better the quality of the reconstructed image. In literature [24], a PSNR below 25 dB is unacceptable because human eye can not recognize/distinguish/identify the image. Figure 6 reports the PSNR using 5-50 DCT coefficients. The results show that the PSNR is very low, ranging from 16 dB when 10 coefficients were selected to 22 dB when 50 coefficients were selected. Clearly, in the worst case, the reconstructed images are still not recognizable.

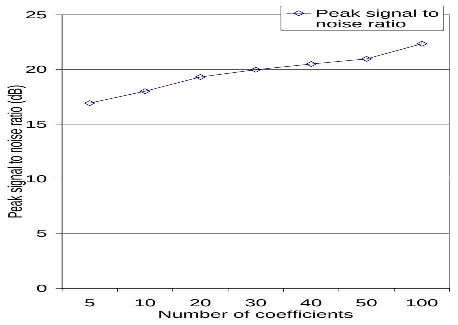

Figure 6. Peak-signal-to-noise ratio in the worst case.

References

[1] R. Agrawal and R. Srikant. Privacy preserving data mining. In *2000 ACM SIGMOD Conference on Management of Data*, pages 439–450, Dallas, TX, May 2000.

[2] O. Al-Baker, R. Benlamri, and A. Al-Qayedi. A GPRS-based remote human face identification system for handheld devices. In *Second IFIP International Conference on Wireless and Optical Communications Networks*, pages 367–371, 2005.

[3] M. J. Atallah and W. Du. Secure multi-party computational geometry. In *WADS2001: Seventh International Workshop on Algorithms and Data Structures*, pages 165–179, Providence, Rhode Island, August 2001.

[4] M. Bartlett, J. Movellan, and T. Sejnowski. Face recognition by independent component analysis. *IEEE Trans. on Neural Networks*, 13(6):1450–1464, 2002.

[5] A. L. Cambridge. ORL face dataset. `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

[6] K. Chen and L. Liu. A random rotation perturbation approach to privacy-preserving data classification. In *IEEE Intl. Conf on Data Mining 2005*, Houston, Tx, November 2005.

[7] T. M. Cover. Rates of convergence for nearest neighbor procedures. In *Inter. Conf. on Systems Sciences*, pages 413–415, 1968.

[8] W. Du and M. J. Atallah. Secure multi-party computation problems and their applications: a review and open problems. In *2001 Workshop on New Security Paradigms*, pages 13–22, Cloudcroft, NM, 2001.

[9] R. Duda and P. E. Hart. *Pattern classification and scene analysis*. John Wiley & Sons, 1973.

[10] K. Etemad and R. Chellappa. Discriminant analysis for recognition of human face images. *Journal of the Optical Society of America A*, 14(8):1724–1733.

[11] R. Gross, E. M. Airoldi, B. Malin, and L. Sweeney. Integrating utility into face de-identification. *Springer Lecture Notes in Computer Science*, 3856:227–242, 2006.

[12] Z. M. Hafed and M. D. Levine. Face recognition using the discrete cosine transform. *International Journal of Computer Vision*, 43(3), 2004.

[13] T. J. Hazen, E. Weinstein, and A. Park. Towards robust person recognition on handheld devices using face and speaker identification technologies. In *Fifth International Conference on Multimodal Interfaces*, pages 289 – 292, 2003.

[14] J. J. Kim and W. E. Winkler. Multiplicative noise for masking continuous data. Technical Report 2003-01, Statistical Research Division, U.S. Bureau of the Census, April 2003.

[15] C. Liu and H. Wechsler. Evolutionary pursuit and its application to face recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):570–582, 2000.

[16] K. Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on Knowledge and Data Engineering.*, 18(1):92–106, January 2006.

[17] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11):1771–1782, 2000.

[18] S. Mukherjee, Z. Chen, and A. Gangopadhyay. A privacy preserving technique for euclidean distance-based mining algorithms using fourier-related transforms. *VLDB Journal*, 15(4):292–315, 2006.

[19] L. Shen and L. Bai. A review on gabor wavelets for face recognition. *Pattern Analysis and Applications*, 9(2-3):273–292, 2006.

[20] L. Sweeney. K-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[21] L. Sweeney, E. M. Newton, and B.Malin. Preserving privacy by de-identifying face images. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):232 – 243, 2005.

[22] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neurosicence*, 3(1):71–86, 1991.

[23] E. Weinstein, P. Ho, B. Heisele, T. Poggio, K. Steele, and A. Agarwal. Handheld face identification technology in a pervasive computing environment. In *Pervasive 2002*, pages 48–54, Zurich, Switzerland, 2002.

[24] S. Winkler. *Digital Video Quality: Vision Models and Metrics*. Wiley, 2005.

[25] M. H. Yang. Face recognition using kernel methods. *Advances in Neural Information Processing Systems*, 14(8), 2002.