On Optimising Local Feature Face Recognition for Mobile Devices[⋆]

Mauricio Villegas and Roberto Paredes

Instituto Tecnológico de Informática
Universidad Politécnica de Valencia
Camino de Vera s/n, Edif. 8G Acc. B 46022 Valencia (Spain)
{mvillegas,rparedes}@iti.upv.es

Abstract. Face recognition is currently a very active research topic due to the great variety of applications it can offer. Moreover, nowadays it is very common for people to have mobile devices such as smart phones or laptops which have an integrated digital camera. This gives the opportunity to develop face recognition applications for this type of devices. This paper treats the problem of face verification in mobile devices. The problem is discussed and analysed, observing which are the difficulties that are encountered. Some algorithms for face recognition are analysed and optimised so that they are better suited for the resource constrains of mobile devices.

1 Introduction

Biometrics is the study of methods for recognising humans based on some intrinsic physical or behavioural trait. These methods have proved to be very useful for different tasks which are common in the current society. Among the different modalities used on biometric systems, face images are very popular because it is an unobtrusive method, well tolerated by users, and with a wide range of applications. Moreover, nowadays it is very common for people to have mobile devices such as smart phones or laptops which have an integrated digital camera. This gives the opportunity to develop face recognition applications for this type of devices.

All of the research done on face recognition could be applied to resource constrained devices. However, the problem lies in the fact that the algorithms require a significant amount of resources. This problem can make the algorithms too slow to be useful in a real application. The research on face recognition has been focused on obtaining low errors and the algorithms are rarely analysed by the amount of resources they require. Initial efforts to develop face recognition applications for mobile devices are based on the approach that the mobile device only captures the image used for recognition [1]. This image is then transferred through a wireless network to a server which does the actual recognition, so

⋆ Work partially supported by the Spanish projects TIN2008-04571 and Consolider Ingenio 2010: MIPRCV (CSD2007-00018).

the cost of the recognition algorithm is not a concern. These systems commonly target face identification applications, which for instance can be very useful for law enforcement agencies. In [1] a face identification system for pervasive computing is presented. The system presented in [2] is also for face identification but based on GPRS. In the work of Hazen the same approach is used, however it includes the face and speech for doing identification [3].

In this work, the interest is on doing all of the computation in the mobile device, without depending on a communication network. This is obviously a harder challenge. In [4] a simple face recognition system based on 2D PCA is proposed, however it is only a preliminary analysis and it does not present any results. The work of Schneider [5], describes a face recognition system developed in C++ for a Symbian platform. It includes a face detector based on skin colour segmentation and a face recognition algorithm based on the position of facial features. The paper includes results with very few images, 21 to assess the face detection and 14 to assess the recognition, therefore the results are not reliable. In [6] a face recognition system for mobile phones is presented however it requires a special camera because it is based on Near-Infrared light. The work presented in [7], describes a system which does face detection using Viola and Jones and recognises faces by means of correlation filters. A complete analysis is made about the computation requirements and how much can be gained by using fixed point arithmetic.

The main objective of this work was to develop a face verification application which worked on commercial mobile devices. However, the idea was not only to have a working software, but also to consider the difficulties that needed to be addressed. The basic problem that face recognition has is that it generally requires a lot of computational resources, something which is not common to have on a mobile device. In this work, some algorithms for face recognition are analysed, and some optimisations for these are proposed. The proposed optimised algorithms were implemented and an extensive empirical analysis was performed.

The remainder of the paper is organised as follows. Section 2 describes the face recognition algorithms considered in the work and the respective proposed optimisations. Then, section 3 presents an empirical analysis of the different algorithms with the proposed optimisations. The results first show the effectiveness of the proposed optimisations, followed by an estimate of how well the algorithms will perform in such circumstances and what is the amount of resources that is actually required. The final section states the conclusions, making emphasis on which objectives were fully reached and which are the areas that need further improvement.

2 Face Recognition in Mobile Devices

This section describes the algorithms to be considered and the proposed optimisations in order to make them more suitable for a mobile device.

2.1 Face Recognition Algorithms Studied

Eigenfaces and Fisherfaces: In the literature the two most cited and popular face recognition algorithms are eigenfaces [8] and fisherfaces [9]. These algorithms are included in this study because they are simple and well understood, and are commonly used as reference. Both of these approaches take the pixel values of the images as a feature vector which is projected onto a lower dimensional space. In the case of eigenfaces, the lower dimensional basis is obtained by Principal Component Analysis (PCA), and fisherfaces first reduces dimensionality by PCA and afterwards further reduces the dimensionality by Linear Discriminant Analysis (LDA). The low dimensional feature vectors $\boldsymbol{y}$ are obtained by

$$\boldsymbol{y} = \boldsymbol{B}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{\mu}) \ , \tag{1}$$

where $\boldsymbol{B} \in \mathbb{R}^{D \times d}$ is the projection basis and $\boldsymbol{\mu}$ is the estimated mean of the feature vectors in the original space.

After the projection, several classifiers could be used for the face verification. A simple one which works a bit better than the one originally proposed in [8] would be the following. A set of face images from people that are not users of the system are used as a world model. The images from the world model and the user are dimensionally reduced using eigenfaces or fisherfaces. As a score to verify an image $\boldsymbol{x}'$ the following is used

$$s(c, \boldsymbol{x}') = \frac{\mathrm{d}(\boldsymbol{y}', \boldsymbol{y}_w)}{\mathrm{d}(\boldsymbol{y}', \boldsymbol{y}_w) + \mathrm{d}(\boldsymbol{y}', \boldsymbol{y}_c)} \ , \tag{2}$$

where $\boldsymbol{y}_w$ and $\boldsymbol{y}_c$ are the nearest neighbours of the world model and the user model respectively. This score is an approximation of the posterior probability of the nearest neighbour classifier.

Local Features: A face verification algorithm which achieves recognition accuracy comparable with other state of the art algorithms is the one known as Local Features [10–12]. Although this algorithms has high processing and memory requirements, just like other state of the art algorithms, it has certain interesting properties. Similar to eigenfaces and fisherfaces, the local features algorithm is based on a simple linear projection and distance calculations. Furthermore the amount of resources required depend completely on a few parameters which can be easily adjusted making a compromise between the recognition performance and computational requirements.

In this approach, each image is represented by a set of feature vectors, named local features, $f = \{\boldsymbol{x}_1, \ldots \boldsymbol{x}_F\} \ \ \boldsymbol{x}_f \in \mathbb{R}^{w^2}$. The set of feature vectors is obtained from all of the possible overlapping regions of the image for a window of size $w \times w$ extracted from a grid. The local features are then dimensionally reduced using a PCA basis.

The face verification approach works as follows. A set of face images from people that are not users of the system are employed to construct a world model which are all the local features extracted from these images. The user models are the extracted features from the training images of that particular subject. When a test image is presented to the system, as a verification score the following approximation to the posterior probability of the client c given the image $\boldsymbol{x}'$ is used

$$s(c, \boldsymbol{x}') = \frac{1}{F} \sum_{i=1}^{F} \mathrm{H}\left(\mathrm{d}(\boldsymbol{y}'_i, \boldsymbol{y}_c) - \mathrm{d}(\boldsymbol{y}'_i, \boldsymbol{y}_w) \right) , \tag{3}$$

where $\boldsymbol{y}_w$ and $\boldsymbol{y}_c$ are the nearest neighbours of the world model and the user model respectively, and $\mathrm{H}()$ is the Heaviside step function. Intuitively this is a simple count of how many features are closer to the user model than to the world model. For further detail refer to [12].

2.2 Algorithm Optimisations

One common characteristic of the face verification algorithms considered is that in the testing phase they all rely on very simple mathematical operations, multiplications and additions/subtractions. In fact all the algorithms have in common that they first perform a dimensionality reduction using a linear projection base, and afterwards a nearest neighbour search using the squared euclidean distance is carried out. These are the two aspects that are going to be optimised both in speed and in memory requirements, and therefore this optimisation will lead to an improvement of all of the algorithms.

Memory Usage Reduction: This will be achieved both for data storage and for program execution by representing all of the data using 8-bit signed integers (byte) instead of the 32-byte floats. This includes the feature vectors which conform the user and world models and the dimensionality reduction base $\boldsymbol{B}$. Since $\boldsymbol{B}$ is inherently real valued, this matrix is approximated by multiplying it by a factor and rounding it, i.e.

$$\boldsymbol{B}_{byte} = \mathrm{round}(2^n \boldsymbol{B}) \ . \tag{4}$$

Also the *kd*-tree structures will use moslty 8-bit integers, although due to precision issues a few variables require 16-bit or 32-bit integers.

Computation Time Reduction: This will be achieved by doing all of the operations using integer arithmetic, first to avoid conversions to and from floats since the data is stored as integers, but also because mobile devices are considerably slower doing floating point arithmetic. To do the projection by integer arithmetic, the following approximation is used

$$\boldsymbol{y}' = \frac{\boldsymbol{B}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{\mu})}{2^m} \approx \mathrm{fitrange}\left(\frac{\overbrace{\mathrm{round}(2^n \boldsymbol{B}^{\mathsf{T}})}^{\boldsymbol{B}^{\mathsf{T}}_{byte}}(\boldsymbol{x} - \boldsymbol{\mu})}{2^{n+m}} \right) , \tag{5}$$

Table 1. Total error rates (%) for face verification on the XM2VTS database with the Lausanne protocol configuration I for different data types.

Algorithm		**Evaluation Set**	**Test Set**
Eigenfaces	float	24.3	19.0
	short	24.3	19.0
	byte	**24.0**	**19.1**
Fisherfaces	float	15.0	11.0
	short	15.0	11.0
	byte	15.0	11.0
Local Features	float	3.9	4.7
	short	3.9	4.7
	byte	3.9	**4.8**

where for 8-bit signed integers

$$\mathrm{fitrange}(z) = \begin{cases} -128 & \text{if } z < -128, \\ 127 & \text{if } z > 127, \\ z & \text{otherwise,} \end{cases} \tag{6}$$

and n, m are positive integers chosen so that for the component with most variance, at least three standard deviations are kept within the range. Choosing the factors to be powers of two is so that the operation can be done a bit faster using bit displacement operations.

The other computation optimised is the nearest neighbour search, which is performed using the squared euclidean distance (so that an expensive square root operation is avoided) employing a *kd*-tree structure to do it efficiently. The search using a *kd*-tree structure can be done using only integer arithmetic. Also for the local feature algorithm, an approximate nearest neighbour search is enough for good recognition performance [12], which depending on a factor ϵ, reduces considerably the time of the search.

3 Results

To be able to assess the proposed optimisations, all of the algorithms were implemented in Java using for the feature vectors 32-bit floating point variables (float) and 32-bit, 16-bit and 8-bit signed integers (int, short and byte). During the realisation of this work only a Nokia N70 mobile phone was available for testing, therefore the execution times are presented only for this device.

3.1 Effect of the Optimisations on Recognition Performance

To assess the effect of the optimisations on the recognition performance, the well known XM2VTS multi-modal database [13], using the Lausanne protocol [14] for face verification was used. Each face verification algorithm presented in

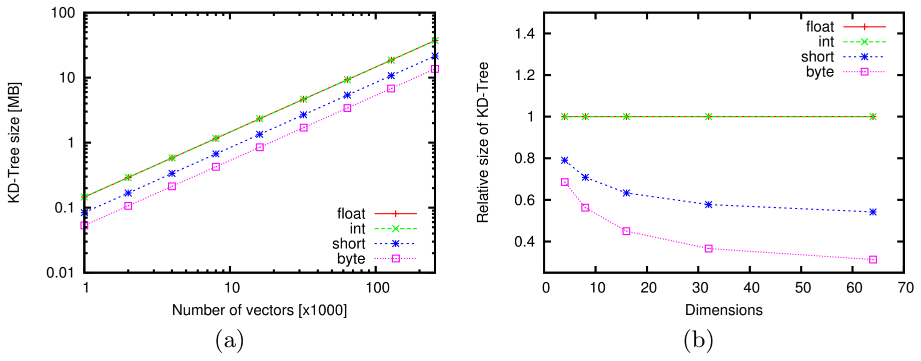


Fig. 1. Size of the *kd*-tree structure for different data types for (a) $d = 32$ and (b) relative to float.

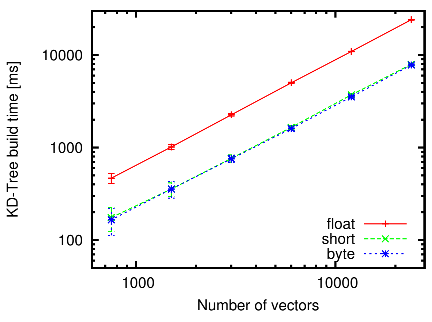
section 2 was tested, and the results for each data type can be observed in table 1. There is a difference between the results for short and byte because for short the fitrange operation is not needed for approximating the projection, therefore it was not applied.

As can be observed in table 1, there is very little difference between the results for the different data types. For fisherfaces, there is no difference up to the second decimal. The differences for all the algorithms are so small that they are not statistically significant. This result confirms that the proposed approximations do not make a noticeable negative impact on the recognition performance of the algorithms. The actual values of the performance are not important because the database is not representative of the conditions that would be encountered in a mobile environment. More adequate values for the estimation of the performance are presented in section 3.4.

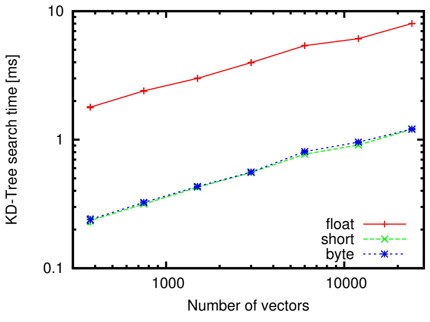
3.2 Memory Requirements

The amount of memory required for storage of the models is $N \times d$ bytes for eigenfaces and fisherfaces, and $F \times N \times d$ bytes for local features. Comparing to the baseline of storage using 32-bit float variables, the space required is reduced to a quarter, although this is leaving out a possible final step of data compression.

Estimating the amount of memory required for processing is a bit more complicated since a *kd*-tree structure is required and its size depends on factors such as the implementation and parameters of the *kd*-tree. As can be observed in figure 1(a), the size of the *kd*-tree increases linearly with the number of vectors. Since the *kd*-trees cannot be represented entirely using 8-bit data, the reduction of memory requirement is not a quarter, and depends on the dimensionality of the vectors. This relationship can be observed in figure 1(b). For normal values of dimensionality the reduction for byte is about a half.



(a)



(b)

Fig. 2. (a) Average *kd*-tree construction time and (b) average nearest neighbour search time, for different data types on a Nokia N70 and $\epsilon = 2$, bucket = 10, $d = 32$.

Table 2. Estimation of verification times (ms) for the different algorithms and data types on a Nokia N70.

Algorithm		**extraction**	**projection**	**score**	**Total**
Eigenfaces (32×32, d_{PCA}=48)	float	379	92.4	22.8	494 ± 9
	short	373	19.5	2.9	396 ± 6
	byte	368	17.8	2.8	389 ± 5
Fisherfaces (32×32, d_{PCA}=96, d_{LDA}=29)	float	375	55.1	14.2	444 ± 8
	short	366	11.2	2.0	378 ± 6
	byte	363	11.1	1.9	375 ± 6
Local Features (48×48, w=9, d=24)	float	232	1266	2175	**3764 ± 13**
	short	119	272	357	**749 ± 15**
	byte	119	271	369	**759 ± 14**

3.3 Computation Time of the Algorithms

As was noted in section 2.2, all of the face recognition algorithms considered rely on first doing a dimensionality reduction by a simple linear projection and afterwards a nearest neighbour search is performed. Apart from this, each algorithm has to do some image processing [15] and a feature extraction process. Each of these tasks have been evaluated and the computation times were measured.

The graphs on figure 2, show the average *kd*-tree construction time and the average nearest neighbour search time for features of 32 dimensions. The construction of the *kd*-trees are about three times faster and the nearest neighbour searches are about six times faster when compared to float.

In table 2 the execution times for each data type and algorithm is presented. For eigenfaces and fisherfaces the gain due to the optimisations is very little. For these configurations most of the processing time is taken by the local histogram equalisation which is integer based and was not possible to optimise. On the other hand, the gain obtained for the local features is almost five times better.

Table 3. Face verification statistics for the different algorithms including average execution times on a Nokia N70.

Algorithm	TER@ FAR=0.1% [%]	Model Size [kB]	Memory [MB]	Enrol. Time [s]	Verif. Time [s]
Eigenfaces (32×32, d_{PCA}=48)	57.0	0.141	0.118	0.387	0.389
Fisherfaces (32×32, d_{PCA}=96, d_{LDA}=29)	37.2	0.085	0.075	0.374	0.375
Local Features (48×48, w=9, d=24)	29.8	28.23	1.093	0.566	0.759

This takes the verification time from almost four seconds, which would be very frustrating for a user of the system, to less than one second. The 95% confidence intervals are included which confirm the significance of the result.

3.4 Summary of Face Verification on a Mobile Device

Using the results from the previous sections, some statistics of the face verification algorithms have been estimated and organised, see table 3. In order to estimate the recognition performance of the algorithms for similar circumstances than would be encountered on a mobile device, the Matched Degraded scenario of the BANCA database [16] was used. It is composed of images taken with a webcam both for training and testing. The results presented are only for the parameters of the algorithms which gave the best performance and the optimised versions of the algorithms. The parameters for the local features algorithm were chosen so that the required resources were adequate for a mobile device. If images of size 64×64 are used, the TER@FAR=0.1% goes down to 17.8%.

These results can be compared with the OKAO Vision face recognition engine for mobile devices. In [17] they report recognition and enrolment times of 0.99 and 2.84 seconds respectively for an ARM920T 200MHz processor. This processor is similar to the one of Nokia N70, and therefore the processing times are comparable to the ones obtained here. However in memory usage the local features approach requires more than 1MB compared to 480kB required by OKAO Vision.

4 Conclusions

This work has been oriented to the difficulties that are encountered when trying to do facial analysis and recognition in mobile devices. Some face recognition algorithms were described and analysed, and this made it possible to propose some key optimisations targeted at the specific characteristics that are encountered in resource constrained devices. The objective was to reduce the system requirements in terms of memory and processing speed without affecting the recognition accuracy.

The proposed optimisations were very simple and general which could be applied to all the algorithms in question, however the ideas can also be applied to many other pattern recognition problems. The optimisations were first to use only 8-bit signed integers to store the data and second to do as much of the processing a possible using only integer arithmetic. These ideas were used for optimising linear feature projections used in dimensionality reduction and for efficient nearest neighbour searching by means of a *kd*-tree structure.

The algorithms were implemented with and without the proposed optimisations and an extensive empirical assessment was performed. Experiments using real face images confirm that the optimisations do not have a significant impact on the recognition accuracy of the algorithms. Furthermore, experiments done on a Nokia N70 mobile phone show that the algorithms can be almost five times faster by using integer arithmetic. The improvements in memory requirements are also significant. For processing of the algorithms the memory used is reduced by a factor of two and for the storage of data by a factor of four.

The recognition accuracy of the algorithms were estimated trying to somewhat simulate the circumstances that are generally found in mobile applications. The results obtained are comparable to existent commercial software, both in processing time or resource requirements. The recognition accuracy needs further improvement, however the performance was measured using very low quality webcam images. For current mobile devices, the quality of the images is higher, therefore the estimated performance is a bit pessimistic.

As a result of this work, three J2ME MIDlet applications were developed[1]. One for doing face detection, another for doing face verification, and the final one which does face gender recognition [18]. They all work considerably well, however they can still be improved much more. As future work an application could be developed which has some real use not just being a demonstration.

References

1. Eugene Weinstein, Purdy Ho, Bernd Heisele, Tomaso Poggio, Ken Steele, and Anant Agarwal. Handheld face identification technology in a pervasive computing environment. In *Short Paper Proceedings, Pervasive 2002*, pages 48–54, 2002.
2. O. Al-Baker, R. Benlamri, and A. Al-Qayedi. A gprs-based remote human face identification system for handheld devices. *Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on*, pages 367–371, March 2005.
3. Timothy J. Hazen, Eugene Weinstein, and Alex Park. Towards robust person recognition on handheld devices using face and speaker identification technologies. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 289–292, New York, NY, USA, 2003. ACM.

[1] `http://www.iti.upv.es/~mvillegas/research`

4. Daijin Kim Sang-Ho Cho, Bong-Jin Jun. Face recognition on a mobile device. In *Proceedings of International workshop of Intelligent Information Processing (IWIIP)*, 2005.
5. C. Schneider, N. Esau, L. Kleinjohann, and B. Kleinjohann. Feature based face localization and recognition on mobile devices. *Control, Automation, Robotics and Vision, 2006. ICARCV '06. 9th International Conference on*, pages 1–6, Dec. 2006.
6. Song yi Han, Hyun-Ae Park, Dal Ho Cho, Kang Ryoung Park, and Sangyoun Lee. Face recognition based on near-infrared light using mobile phone. In *ICANNGA (2)*, pages 440–448, 2007.
7. Chee Kiat Ng, Marios Savvides, and Pradeep K. Khosla. Real-time face verification system on a cell-phone using advanced correlation filters. *Automatic Identification Advanced Technologies, IEEE Workshop on*, 0:57–62, 2005.
8. A.P. Turk, M.A.; Pentland. Face recognition using eigenfaces. *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 586–591, 3-6 Jun 1991.
9. P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, Jul 1997.
10. R. Paredes, J. C. Pérez, A. Juan, and E. Vidal. Local Representations and a direct Voting Scheme for Face Recognition. In *Proc. of the Workshop on Pattern Recognition in Information Systems (PRIS 01)*, Setúbal (Portugal), July 2001.
11. M. Villegas and R. Paredes. Illumination Invariance for Local Feature Face Recognition. In *1st Spanish Workshop on Biometrics*, pages 1–8, Girona (Spain), June 2007. -.
12. M. Villegas, R. Paredes, A. Juan, and E. Vidal. Face verification on color images using local features. In *Proceedings of the Workshop on Biometrics, in association with CVPR 2008*, Anchorage, AK, USA, June 2008. IEEE Computer Society.
13. K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre. XM2VTSDB: The extended M2VTS database. In R. Chellapa, editor, *Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 72–77, Washington, USA, March 1999. University of Maryland.
14. J. Luettin and G. Maître. Evaluation Protocol for the extended M2VTS Database (XM2VTSDB). IDIAP-COM 05, IDIAP, 1998.
15. M. Villegas and R. Paredes. Comparison of illumination normalization methods for face recognition. In Mauro Falcone Aladdin Ariyaeeinia and Andrea Paoloni, editors, *Third COST 275 Workshop - Biometrics on the Internet*, pages 27–30, University of Hertfordshire, UK, October 27-28 2005. OPOCE.
16. Enrique Bailly-Bailliére, Samy Bengio, Frédéric Bimbot, Miroslav Hamouz, Josef Kittler, Johnny Mariéthoz, Jiri Matas, Kieron Messer, Vlad Popovici, Fabienne Porée, Belén Ruíz, and Jean-Philippe Thiran. The BANCA database and evaluation protocol. In *AVBPA*, pages 625–638, 2003.
17. Yoshihisa Ijiri, Miharu Sakuragi, and Shihong Lao. Security management for mobile devices by face recognition. In *MDM '06: Proceedings of the 7th International Conference on Mobile Data Management*, page 49, Washington, DC, USA, 2006. IEEE Computer Society.
18. M. Villegas and R. Paredes. Simultaneous learning of a discriminative projection and prototypes for nearest-neighbor classification. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.