

Efficient Face Recognition Using FPGA and Semantic Features for Security Controls

Ying-Hao Yu ^{a*}, Yu-Chen Xu ^b, Yi-Siang Ting ^c and Ngaiming Kwok

^aDepartment of Electrical Engineering, National Chung Cheng University,
No. 168, University Rd., Min-Hsiung Township, Chiayi County 62102, Taiwan

^bSchool of Mechanical and Manufacturing Engineering,

The University of New South Wales Sydney, NSW 2052, Australia

E-mail: yinhaun@gmail.com, p1239688@gmail.com, tingyisiang@gmail.com, nmkwok@unsw.edu.au

Abstract –

Facial features are essential to biometric authentication. Nowadays an effective face recognition system is mostly composed of feature's transformation, geometric analysis, and recursive training to resist illegal intrusion. However, the design challenge always arises from the dilemma of lower computing resource usages, power consumption, and real-time performance under rigorous operations of construction sites. Here a chip-based face recognition is proposed by using semantic features to achieve a minimal dataset and high processing speed without any complicated formulization processes. Our experimental results on single FPGA chip demonstrate the feasibility to realize a miniature and efficient security control system for construction facilities in the future.

Keywords –

Face recognition; Semantic features; FPGA; Security control; Biometrics

1 Introduction

In a construction project, safety and security controls are indispensable for loss reduction from accidents and crimes. The main issue of safety is about the labor working with hazard materials and danger zones. Such problems can be fundamentally precluded by the management to carry out training and operating discipline. However, an impervious security control system relies on high techs to take over manpower for 24 hours a day. Unlike the key and traditional sensors which have been consider insufficient for authentication, the biometric technologies recently has been reported as an efficient choice to resist illegal access by unique features [1].

Biometric authentication is an identification technology based on the unique physiological and behavioral traits, which involves face, fingerprints, ears, keystroke dynamics, voiceprint, gait, and the shape of body [1]. Among these biometric technologies, the face

recognition is the favorite one due to advantages of lower cost, non-invasive sensing, and portable dimension [2], [3]. This kind of methodology not only enables security control system to identify but also tracks people with a longer distance at public or domestic places [3], [4]. Currently, the biometric design with digital camera can be also found on embedded system (e.g., smart phones and tablets) as a new trend to check user's access identity [5].

According to the discussion in [1], the way to design a biometric system with lower dataset, computing load, memory space, and time delay on mobile devices still remains a challenge. Although the biometrics designed with external server and sophisticated face recognition algorithms can achieve a good result [6], the security functionality will be totally collapsed if the network is disconnected [1]. For the face detection on embedded systems, the algorithm named Adaboost with Haar-like features is broadly adopted and improved [7]. However, the Haar-like features is too rough and inaccurate for face features' classification. Currently, the most of face recognition systems employ principal component analysis (PCA) and support vector machine (SVM) for better classification, but the processing speed is usually deteriorated by complicated algorithms [8].

In this paper, we propose a real-time face recognition by using feature extraction of semantics-based vague image representation (SVIR) on single field-programmable gate-array (FPGA) chip, using hardware circuit designs [9]. The SVIR is a rule-based feature extraction without operations of trigonometric functions and matrices, and it also has no need to analyze a pattern via geometry. Moreover, the concise algorithm enables feature's evolutions and recognition to be performed during image pixels scanning without any external memory space. Thus the advantages of miniature dimension, real-time performance, economical resource usages, and power saving by lower system clock speed are entirely sufficient for construction sites, warehouse, and engineering vehicles.

This paper is organized as the follows. In the section 2, the methods of skin color detection using HSV color

space, applied SVIR algorithm, and the mechanism of face recognition will be detailed. The experimental results on single FPGA chip is in the section 3 followed by discussion in section 4. Finally, the conclusion is drawn in section 5.

2 Methods

2.1 Skin Color Detection by HSV

The skin color in a picture is firstly transferred into binary image as the input of SVIR. By considering the work of Yang et al. [10], we sampled faces from Asians and defined new HSV skin color threshold for $160 < H < 184$, $V > 900$, and $S < 50$ with the scale of 0 to 255. At the output of binary image, a real-time noise filter from [11] was implemented for:

$$I_{(i,j)} \langle 1|0 \rangle = I_{(i-1,j)} \cap I_{(i-2,j)} \cap I_{(i-3,j)} \quad (1)$$

where (i, j) denotes the location of pixel in picture and I is the output binary image. The true logic will be confirmed with three successive “1” outputs while the random noise produces a false output logic “0”, as shown in Figure 1.

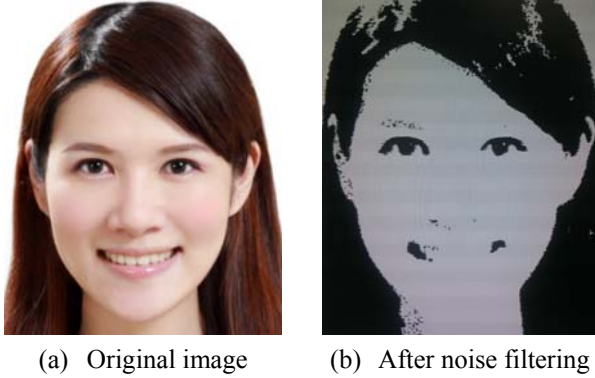


Figure 1. Binary image with noise filtering

2.2 SVIR Algorithm

The method of SVIR is based on our previous work in [9], which is consisted of bipolar image encoding, vertical evolution of sub-patterns, and lateral combination with sub-windows to perform feature's extraction and evolutions. Due to the small dimension of facial features, the algorithm of lateral combination was unemployed in this design and a couple of SVIR rules were also amended for face recognition.

2.2.1 Bipolar Image Encoding

Unlike the Adaboost sampling a picture pixel-by-pixel for sub-images matching [8], the SVIR samples a picture by stationary sub-windows $S_{(i,j)}$. Each sub-window is also split up into ten sub-sections s_y in order to integrate pixels number c during image pixels scanning as:

for $2 \leq y \leq 9$,

$$s_y(P, N) = \begin{cases} (1, 0), & \text{if } |c_y - c_{y-1}| > |c_{y-1} - c_{y-2}| \\ (0, 1), & \text{if } |c_y - c_{y-1}| < |c_{y-1} - c_{y-2}| \\ (0, 0), & \text{if } |c_y - c_{y-1}| = |c_{y-1} - c_{y-2}| \end{cases} \quad (2)$$

where y is sub-section's order by image pixels scanning direction from the left- to right-hand side. The P and N are sub-pattern working as polarities of bipolar encoding to represent pixels' variation. Meanwhile, the s_1 is an exception of (2) determined by the variation between c_0 and c_1 , and s_0 is undefined in SVIR.

In addition to P and N annotations, the sampled sub-pattern can be also described with a concave or convex trend. Thus the superscript “+” is for $c_y > c_{y-1}$ (concave) and the superscript “-” is defined for $c_y < c_{y-1}$ (convex). When c_y is equal to c_{y-1} , the trend of sub-pattern is determined by c_{y-1} and c_{y-2} instead. Besides, s_y will be denoted as Z_e for a rectangle pattern, and the notation Z_0 is for empty sub-sections from c_{y-2} to c_y . Furthermore, a Z_u is defined for positive slope and Z_d is for the negative condition of Z_u .

The sketched sampling result of bipolar encoding is shown in Figure 2. Nine sub-patterns from s_1 to s_9 compose of a bigger sub-feature. It can be seen that SVIR does not express a feature by geometric formulization or abstract patterns (e.g., Fourier and Haar-like features) but with a series of comprehensible contours. Such result provides a convenience to represent a pattern in semantic way, e.g., a pattern in concave, convex, rectangular, or protrudent shape.

2.2.2 Vertical Evolution

Pattern recognition with irregular contour usually dramatically increases the difficulty of classification and training. For this, SVIR provides rule-based evolutions during image pixels scanning in order to approximate a sampled feature [9]. The proposed evolution rules for two sub-features stacking up are as follows:

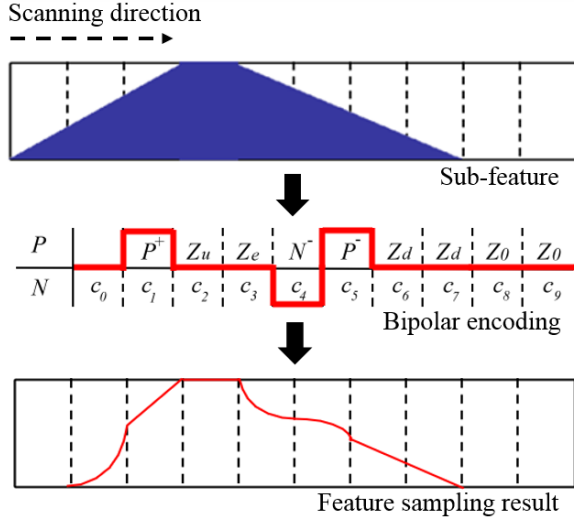


Figure 2. Sub-feature sampling by bipolar encoding

1. Proposition 1: Invariability

Two homogeneous sub-patterns stacking up will result in a similar pattern to original one as:

$$P^{(+,-)}\left(N^{(+,-)}\right)+P^{(+,-)}\left(N^{(+,-)}\right)=P^{(+,-)}\left(N^{(+,-)}\right) \quad (3)$$

and

$$Z_{(u,d,e)}+Z_{(u,d,e)}=Z_{(u,d,e)} \quad (4)$$

2. Proposition 2: Cancellation

Two different sub-patterns stacking up with homogeneous superscript will result in invariant conditions of $Z_{(u,d)}$.

$$P^{(+,-)}\left(N^{(+,-)}\right)+N^{(+,-)}\left(P^{(+,-)}\right)=Z_{(u,d)} \quad (5)$$

3. Proposition 3: Complementation

If both sub-patterns have homogeneous polarity and different superscripts, the evolution result will be the same as the sub-pattern at the bottom.

$$P^{(+,-)}\left(N^{(+,-)}\right)+P^{(-,+)}\left(N^{(-,+)}\right)=P^{(-,+)}\left(N^{(-,+)}\right) \quad (6)$$

4. Proposition 4: Mirroring

The evolution result becomes an invariant state (rectangle) while both stacking sub-patterns are in a mirroring relationship to each other,

$$P^{(+,-)}\left(N^{(+,-)}\right)+N^{(+,-)}\left(P^{(+,-)}\right)=Z_e \quad (7)$$

and

$$Z_{(u,d)}+Z_{(d,u)}=Z_e \quad (8)$$

5. Proposition 5: Carry

SVIR's evolution is not only occurring at individual sub-section but also involves with vicinities by carrying mechanisms as addition. With this mechanism, evolution results after stacking up sub-features will be gradually dominated by the bottom (recent) features. The outmoded (top) sub-features will be continually phased out in two directions of s_6 to s_9 and s_5 to s_2 . There are five cases for SVIR's carry as:

Case 1: $P^{(+,-)}$ and $N^{(+,-)}$ stacking with $Z_{(u,d,e)}$

This case is applied to sub-sections s_2 to s_9 and will produce a carry as original one ($P^{(+,-)}$ or $N^{(+,-)}$).

$$P^{(+,-)}\left(N^{(+,-)}\right)+Z_{(u,d,e)}=Z_{(u,d,e)} \quad (9)$$

and

$$Z_{(u,d,e)}+P^{(+,-)}\left(N^{(+,-)}\right)=Z_{(u,d,e)} \quad \text{carry } P^{(+,-)}\left(N^{(+,-)}\right) \quad (10)$$

Case 2: $Z_{(u,d)}$ stacking with Z_e

Once $Z_{(u,d)}$ stacks with Z_e among sub-sections s_2 to s_9 , the carry is similar to *Case 1* as:

for $s_* \neq Z_e$,

$$Z_{(u,d)}\left(Z_e\right)+Z_e\left(Z_{(u,d)}\right)=Z_e \quad \text{carry } Z_{(u,d)} \quad (11)$$

where s_* denotes the next sub-section. This case will not be triggered if the evolution result of s_* is still at Z_e .

Case 3: Stacking at Mirroring and Invariability

The carry operation should follow propositions of invariability or mirroring and is inactive as:

$$Z_{(u,d)} + Z_{(d,u)} = Z_e, \text{ mirroring} \quad (12)$$

$$Z_{(u,d,e)} + Z_{(u,d,e)} = Z_{(u,d,e)}, \text{ invariability} \quad (13)$$

Case 4: Stacking with Z_0

The carry operation is also inactive once sub-patterns stack with Z_0 .

$$Z_0 + P^{(+,-)}(N^{(+,-)}) = P^{(+,-)}(N^{(+,-)}) \quad (14)$$

$$P^{(+,-)}(N^{(+,-)}) + Z_0 = P^{(+,-)}(N^{(+,-)}) \quad (15)$$

$$Z_0(Z_{(u,d,e)}) + Z_{(u,d,e)}(Z_0) = Z_{(u,d,e)} \quad (16)$$

Case 5: Carry at s_1

The carry at s_1 is also disable and the evolution rules at this sub-section are defined as:

$$\left(P^{(+,-)}, N^{(+,-)}\right)_1 + Z_0 = \left(P^{(+,-)}, N^{(+,-)}\right)_1 \quad (17)$$

and

$$Z_0 + \left(P^{(+,-)}, N^{(+,-)}\right)_1 = \left(P^{(+,-)}, N^{(+,-)}\right)_1 \quad (18)$$

$$Z_e + \left(P^{(+,-)}, N^{(+,-)}\right)_1 = \left(P^{(+,-)}, N^{(+,-)}\right)_1 \quad (19)$$

$$\begin{aligned} & \left(P^{(+,-)}, N^{(+,-)}\right)_1 + Z_e \\ &= \begin{cases} \left(P^{(+,-)}, N^{(+,-)}\right)_1, & \text{if entire } [c_2, c_9] \neq Z_e \\ Z_e, & \text{if entire } [c_2, c_9] = Z_e \end{cases} \quad (20) \end{aligned}$$

In the end, an evolution result for two sub-features' stacking up is shown in Figure 3. Two different sub-features are located at successive sub-windows and combine into a larger pattern (feature). Generally, with a larger pattern, the system also needs more memory spaces to keep dataset at the same time. However, after applying vertical evolution, only a set of bipolar encoding is required to describe the larger pattern with the same budget of memory space. This leads significant improvements in computing load, and the memory consumption and access time are also less than the other traditional algorithms.

2.3 Face Sampling and Recognition

Face recognition is usually based on the classification of facial features for eyes, nose, mouth,

and ears. In our design, we classified different faces by eyes and nose after considering the stability of image resources. As shown in Figure 4, the testee aligned his

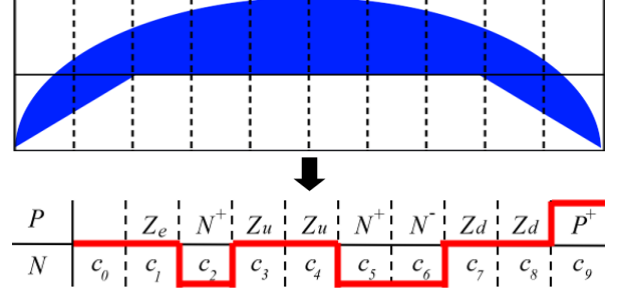


Figure 3. Vertical evolutions for two sub-features

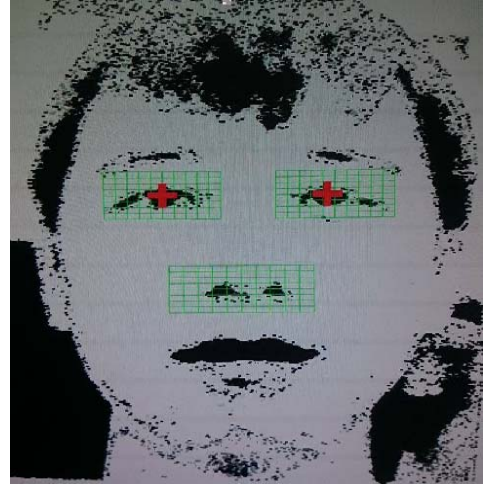


Figure 4. Face features sampling by sub-windows

face by overlapping pupils on the cross marks. Meanwhile, there were eight sub-windows at every facial feature, and each feature was also split into two parts in order to perform vertical evolution by four sub-windows. Thus we totally obtained six features from the locations of eyes and nose.

For the dataset collection, we firstly provided forty SVIR eye-templates for eyes and the other ten nose-templates for nose. Next the system sampled a face for thirty times within one second and then determined testee's face type by the highest similarity to feature templates. The face, thus, can be easily represented by a set of codes and recorded in chip's registers without the use of external memory, as shown in the Table 1.

Finally, we could achieve face recognition by checking the similarity of sub-features between testee and dataset. Here we empirically adjusted different weights for each sub-section at upper eyes (5 points),

lower eyes (3 points), and nose (2 points). Accordingly, there is a total of 162 points with six sub-features, and the similarity between testee and dataset can be determined by:

$$\text{Similarity}(\%) = \frac{\text{Final score}}{162} \times 100 \quad (21)$$









3 Experimental Results

The proposed face recognition system was implemented on a developing platform with Cyclone II EP2C70F896C6 FPGA chip. Our digital camera module's resolution was set for 1280×1024 pixels. The video frame rate was at 34 fps with 77 MHz pixel clock. This pixel clock was also employed to synchronize with SVIR and recognition circuits.

The definition for sub-patterns and corresponding codes are list in Table 1. Each sub-pattern discussed in Section 2 can be simply translated into a number. Thus the representation of a sub-feature before/ after evolutions only requires a set of nine numbers. This design concept is very important to chip and embedded system designs in that miniature systems usually do not equip with sufficient onboard and inbound memories.

As the example in Figure 5, forty-six Asian faces were collected as strangers to compete with our testee. It can be seen that similarity between testee and recorded dataset is noticeably higher to the other strangers. This result can help us to set a threshold for discriminating testee from strangers. In our design, three testees with different face types had been invited to compete with these Asian faces. The collected test results then became a useful basis for threshold determination.

Table 1. The list of SVIR coding with corresponding sub-patterns

Code Name	Code	Pattern	Code Name	Code	Pattern
P^+	11		Z_0	00	
P^-	12		Z_u	01	
N^+	21		Z_d	02	
N^-	22		Z_c	03	

According to our tests with three testees, we derived false rejection rates (FRRs) and false acceptance rates (FARs) based on Bayes' theorem. Here we picked out highest FAR and FRR from tests by different similarity thresholds, as shown in Figure 6. The equal error rate (EER) with FAR and FRR then could be found about 85% similarity, which is the balanced point for FAR and FRR adjustments. This threshold makes a clear separation of our testees from strangers for 7% at least.

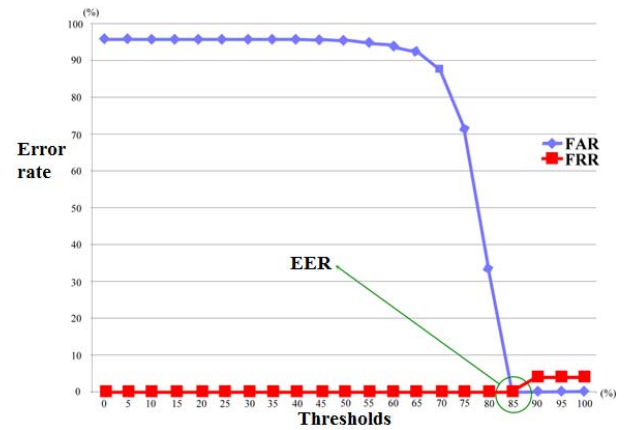


Figure 6. Face recognition error rates with different recognition thresholds

Table 2. Hardware resource usages of FPGA

Designs	Logic elements (%)
Image capture/ Demosaicking/VGA	1,700(2.4%)
HSV colour space/ Binary image / noise filter	1,300(1.9%)
SVIR $\times 6$	5,898(8.6%)
Template Comparison $\times 6$	5,497(8%)
Dataset & Recognition	698(1%)
Test circuits (LCD/LED)	736(1%)
Total	15,829(23%)
Inbound RAM	82,864 bits (7%)

4 Discussion

Face recognition based on SVIR can attain a significant improvement in hardware resource usages. As the logic elements (LEs) consumption in Table 2, we initially duplicated our SVIR circuits for upper eyes, lower eyes, and nose with higher LEs usage. Nevertheless, our LEs consumption for feature

extraction (5,898 LEs) is still much lower than the chip-based principle component analysis (PCA) (24,958 LEs) and Fourier Transform (12,014 LEs) designs [12], [13].

Besides, another salience of the proposed system is real-time performance. Since SVIR can perform feature extraction during image pixels scanning at ears and nose [9], the similarity comparison between testee and datasets can be easily performed in the rest of picture. This advantage allows us to check testee's face image quality for 30 times per second and then pick out the best face image for recognition processes.

Finally, the use of 90 SVIR templates for defining testee's face type can reach 1.6 million combinations, which is sufficient for real-life applications. Here testees' face features are only stored in chip's registers with concise data format. Such advantage is very important to reduce the power consumption and installation dimension for construction sites or vehicles.

5 Conclusion

Face recognition is usually impeded by image's resolution and system clock speed. Many designs, therefore, adopt the strategy to degrade picture's quality or dramatically increase system clock speed in order to attain real-time performance. Contrarily, the rule-based SVIR can avoid such problem from feature's extraction, evolutions, and recognition during image pixels scanning. Additionally, the concise features format also allows us to leave out the requirement of external memory. The use of SVIR evolutions and templates can avoid tedious training processes and designs. These advantages imply a possibility to install the proposed system at a narrow and rigorous environment without the supports from external server and networks. It can be seen that the proposed face recognition system is promising for the application in construction sites. The future work will focus on the recognition with different face features.

Acknowledgement

Supports from Ministry of Science and Technology funded by the government of Taiwan under Grant: MOST 104-2221-E-194-049-MY2 is gratefully acknowledged.

References

- [1] Mitra S., Wen B., and Gofman M. *Biometrics in a data driven world*, CRC Press, 6000 Broken Sound Parkway NW, Suite 300, Boca Raton, FL 33487-2742, 2016.
- [2] Shams N., Hosseini I., Sadri M S., and Azarnasab E. Low cost fpga-based highly accurate face recognition system using combined wavelets with subspace methods. In *Proceedings of the International Conference in Image Processing*, pages 2077–2080, Atlanta, USA, 2006.
- [3] Zuo F. and Peter H N. Real-time embedded face recognition for smart home. *IEEE Trans on Consumer Electronics*, 51(1), 183-190, 2005.
- [4] Jain A K., Duin R P W., and Mao J. Statistical pattern recognition: a review. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 22(1), 4-37, 2004.
- [5] Chimote M G R. and Tarbani N M., A survey paper on authentication system in android phones. *The International Journal of Scientific & Engineering Research*, 7(2), 238-240, 2016.
- [6] Kremic E., Subasi A., Hajdarevic K. Face recognition implementation for client server mobile application using pca. In *Proceedings of the International Conference on Information Technology Interfaces (ITI)*, Dubrovnik, Croatia, 2012.
- [7] Cho J., Mirzaei S., Oberg J., and Kastner R. Fpga-based face detection system using haar classifiers. In *Proceedings of International Conference on Field Programmable Gate Arrays (ACM/SIGDA)*, pages 103-112, 2009.
- [8] Jeon D., Dong Q., Kim Y., Wang X., Chen S., Yu H., Blaauw D., and Sylvester D. A 23-mw face recognition processor with mostly-read 5t Memory in 40-nm cmos. *IEEE Journal of Solid-state Circuits*. (To appear)
- [9] Yu Y H., Lee T T., Chen P Y., and Kwok N. On-chip real-time feature extraction using semantic annotations for object recognition," *Real-Time Image Processing*, DOI 10.1007/s11554-014-0474-2, 2014. (6 pages)
- [10] Yang L., Li H., Wu X Y., and Zhao D. An algorithm of skin detection based on texture. In *Proceedings of IEEE International Congress on Image and Signal Processing (CISP 2011)*, Shanghai, China, 2011.
- [11] Yu Y H., Kwok N M., and Ha Q. Color tracking for multiple robot control using a system-on-programmable-chip. *Automation in Construction*, 20, 669-676, 2011.
- [12] Zhong F., Capson D W., and Schuurman D C. Parallel architecture for pca image feature detection using fpga. In *Proceedings of IEEE Conference on Electrical and Computer Engineering*, Canada, pages 1341-1344, 2008.
- [13] Bahoura M. and Ezzaidi H. Fpga implementation of a feature extraction technique based on fourier transform. In *Proceedings of IEEE International Conference on Microelectronics*, Algiers, Algeria, pages 1-4, 2012.



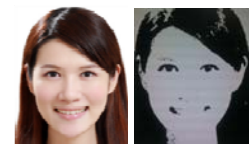
91% (Testee)



93% (Testee)



56%



68%



57%



77%



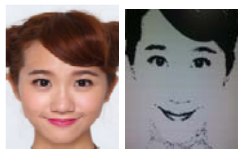
78%



29%



74%



68%



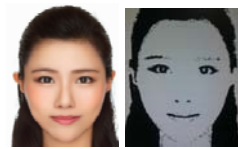
59%



64%



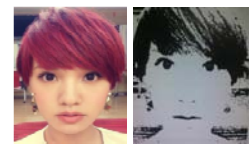
72%



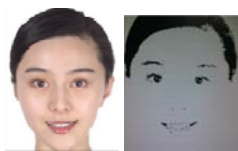
74%



74%



68%



54%



68%



61%



62%



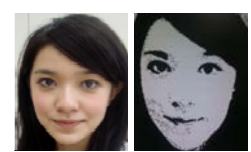
68%



65%



74%



74%



68%



68%



53%



63%

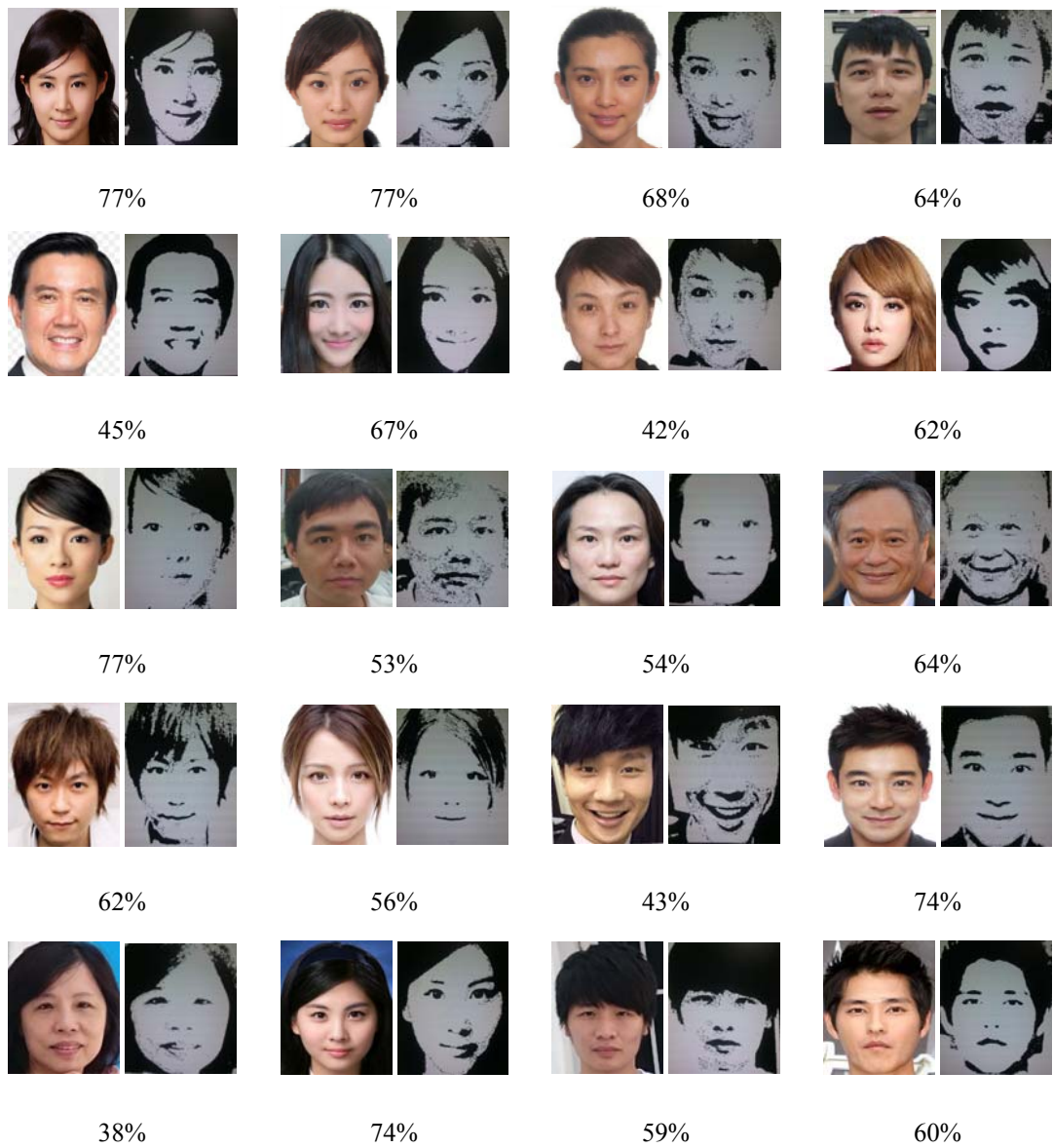


Figure 5. Face recognitions by competing similarity between testee and strangers