A Video Surveillance System for Fingerprint Acquisition

Yijing Su, Jianjiang Feng, Jie Zhou
Department of Automation
Tsinghua University, Beijing 100084, China
Email: {jfeng, jzhou}@tsinghua.edu.cn

***Abstract*—In fingerprint-based watch-list systems used by law enforcement, immigration, and military agencies, the targeted users (such as criminals, illegal immigrants, terrorists) have strong incentive to cheat the fingerprint system. A simple but effective method to cheat the system is to present the wrong fingers or even not a finger (e.g., palm). The current solution to prevent such behavior is to let the official closely monitor the fingerprint acquisition process. However, automatic solution is desired because the number and scale of fingerprint-based watch-list systems are rapidly increasing while human labor is expensive. We propose a video surveillance system to ensure the validity of input fingers. The proposed system captures the region around the fingerprint scanner using a webcam, determines whether the posture of the hand is valid, and estimates finger position (left index, right middle). Experimental results demonstrate the validity of the proposed algorithm.**

I. Introduction

The uniqueness of fingerprint makes it an effective biometric characteristic for person recognition. The recognition is primarily accomplished in manual approach rather than through computer aided system. And later the increased demand for recognition resulted in the development of Automated Fingerprint Identification Systems (AFIS). They were first used in law enforcement agencies for identifying criminals. Now development makes them available for commercial-used, such as border control, access control, commercial transaction and so on.

Depending on the application context, some fingerprint systems operate in positive mode while the others in negative mode. A positive recognition system aims to prevent multiple people from using the same identity while a negative recognition system aims to prevent a single person from using multiple identities [1]. Systems used by law enforcement, immigration, and military agencies typically operate in negative mode. For example, the immigration agencies can record fingerprints of expelled illegal immigrants in a watch-list database and later the officials will be able to tell if an international visitor has a record of illegal immigration by searching his/her fingerprints against the watch-list.

In large scaled AFIS, finger position information (left index, right middle, etc.) is usually required to speed up matching. In this case, when acquiring fingerprints, the fingerprint system typically asks the user to present one's ten fingers in a predetermined order. A simple but effective method to cheat the fingerprint system is to present the ten fingers in an incorrect order. Note that the user of a watch-list system cannot be trusted since they may have strong incentive to cheat the fingerprint system. The current solution to prevent such behavior is to let the official closely monitor the user. However, automatic solution is desired because the number and scale of fingerprint-based watch-list systems are rapidly increasing while manual monitoring is expensive and tedious. And sometimes errors of finger order occur even when the whole fingerprint acquisition process is monitored by the official.

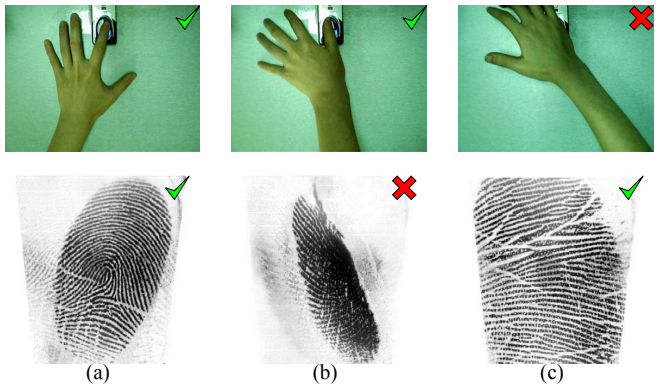


Fig. 1. Images from a fingerprint scanner and a webcam which is mounted above the fingerprint scanner in three cases: (a) the fingerprint is accepted since the pressed is a finger and the fingerprint quality is good, (b) it is not accepted since the fingerprint quality is bad, (c) it is not accepted since the pressed is not a finger. Without the proposed video surveillance system, the print in (c) will be incorrectly accepted since the print looks like a fingerprint.

We propose a video surveillance system to automatically determine whether the posture of the hand is valid or not and also determine which finger is pressed. A fingerprint scanner (U.are.U 4500) and a webcam, which is mounted above the fingerprint scanner to capture a large region around the scanner, are used in the system. This fingerprint scanner can trigger an event when an object is pressed on it. Once receiving the event, an image is captured using the webcam. Then the image is examined whether the posture of the hand is valid or not. If the posture is valid, the finger position of the pressed finger is estimated. Meanwhile, the image quality of fingerprint is estimated by the quality evaluation module of the fingerprint scanner. Only when the fingerprint quality is good and the hand

This work was supported by the National Natural Science Foundation of China under Grants 61020106004, 61005023, and 61021063.

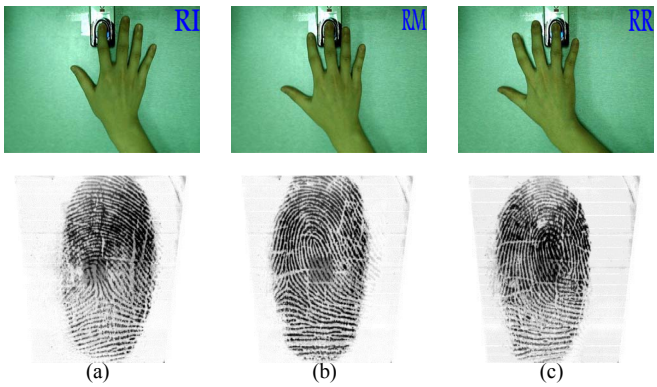


Fig. 2. While it is almost impossible to tell the finger position from the fingerprint images shown on the bottom row, it is a relatively easy task to determine finger position (shown on the top-right of the image) based on the image of the whole hand shown on the top row.

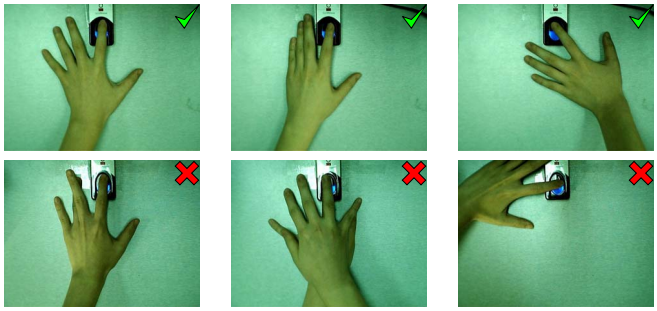


Fig. 3. Various postures to present the left index. The top three postures are accepted while the bottom three postures should be rejected.

posture is valid, the fingerprint is accepted. Figure 1 shows the images captured by the webcam and the fingerprint scanner in three cases. Case (a) is accepted since the pressed is a finger and the fingerprint quality is good. Case (b) is not accepted since the fingerprint quality is bad. Case (c) is not accepted since the pressed is not a finger. Without the proposed video surveillance system, case (c) will be incorrectly accepted since the print looks like a fingerprint according to any fingerprint image quality measure [2]. In addition, as shown in Fig. 2, while it is almost impossible to tell the finger position from a fingerprint image, it is a relatively easy task to determine finger position based on the image of the whole hand.

Since human hand has 30 degrees of freedom [3], the postures of presenting the same finger can be quite different (see Fig. 3). Allowing various postures will make the problem unnecessarily difficult and increase the possibility of errors. Thus our system requires subjects to stretch theirs hands, so that all fingers are presented in the image without crossing with each other. Other postures will be automatically rejected.

The rest of the paper is organized as follows. We first review related work in Section II and then describe the proposed algorithm in Section III. Experimental results are presented in Section IV. Finally, the conclusions and discussions are presented in Section V.

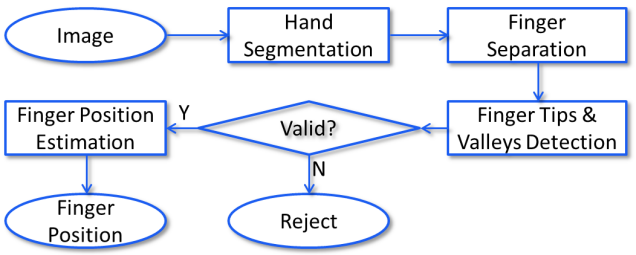


Fig. 4. Flowchart of the proposed system.

II. Related Work

One major goal of the proposed system is to prevent people in watch-lists from cheating the system. Related topics in the fingerprint literature include using fingerprint quality assessment algorithm to detect poor quality fingerprints [2], using liveness detection techniques to detect spoof finger [4], and using altered fingerprint detector to detect fingerprint alteration [5]. But these techniques cannot defer purposely presenting wrong fingers by malicious users.

Another related topic is gesture recognition used for human-computer interface. Zhang et al. [6] use a camera to track movements of a finger and map the virtual typing of the user on a paper to keyboard input. However, they only detect finger movement, not specific finger position.

Sugiura and Koseki [7] proposed to capture ten fingerprints of the user and later recognize finger position by matching the input fingerprint to stored fingerprints. Different fingers are used as shortcuts to launch common software. In [8], LED tags of five colors (pink, blue, green, yellow, purple) are attached to the corresponding fingers and then finger position recognition can be performed by a simple image processing algorithm. Apparently, it is impractical to apply these solutions to our scenario.

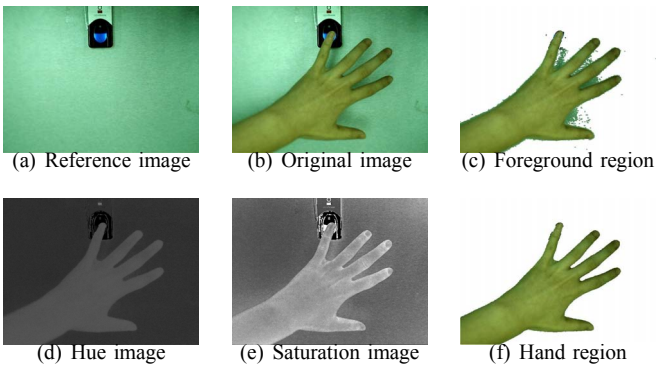
III. Finger Position Estimation

The proposed system consists of the following steps: 1) hand segmentation, 2) finger separation, 3) finger tips and valleys detection, and 4) finger position estimation. See Fig. 4 for the flowchart.

A. Hand Segmentation

We assume that the background is relatively static, the fingerprint scanner is fixed, and the color of background is different from the skin color. These assumptions can usually be satisfied. We model the background pixel with a Gaussian Model in the HSV color space and use background subtraction method to obtain the foreground region (see Fig. 5(c)).

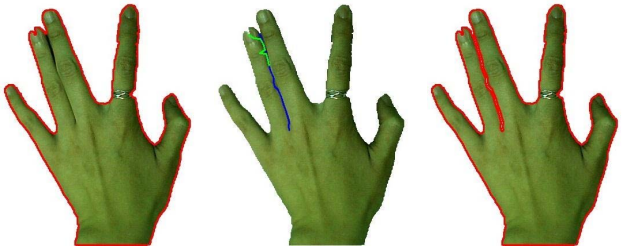
Since the foreground region may contain both the hand region and its shadow (see Fig. 5(c)), a shadow detection is implemented. The main characteristic of shadow is that it has similar chromaticity but lower brightness than those of the same pixel in the background image. Chromaticity is determined by its hue and saturation and we found that skin region looks more convergent in the hue component (see



(a) Reference image (b) Original image (c) Foreground region

(d) Hue image (e) Saturation image (f) Hand region

Fig. 5. Hand segmentation



(a) (b) (c)

Fig. 6. Finger separation. (a) A segmented hand with clung fingers (hand contour is shown in red), (b) detected gaps (shown in blue), (c) new hand contour.

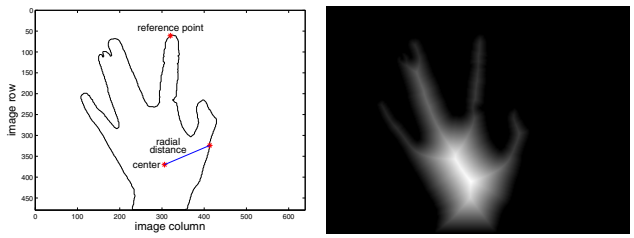
Fig. 5(d) and Fig. 5(e)). So we divide the foreground pixels into two classes (hand and shadow) using Otsu's thresholding algorithm in the hue components. The characteristic of the shadow in the intensity component is not considered since it is sensitive to illumination change. Fig. 5(f) shows the obtained hand region.

B. Finger Separation

The proposed system estimates finger position based on detecting finger tips and valleys on the hand contour. However, it is not easy to detect finger tips and valleys when the adjacent fingers are clung together (see Fig. 6(a)). To facilitate the extraction of finger tips and valleys, we detect the gap between clung fingers and then re-compute the hand contour.

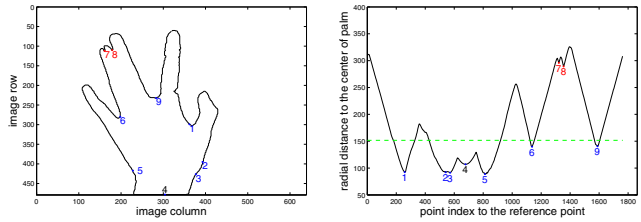
Before describing finger separation algorithm, we first introduce several features and representations of the hand contour which will be used later. The center of the palm is defined as the point which is farthest from the background, namely, the brightest point in the distance transform map (Fig. 7(b)). Intersection between the hand contour and the fingerprint detector's vertical axis (which is known and fixed) is set as the starting point (reference point) of the contour and the points on the contour are arranged in a clockwise order. The distance between a point on the boundary and the palm center is termed as the radial distance of the point (see Fig. 7(a)). The radial distance representation (see Fig. 8(b)) of the hand contour shows the radial distance of each point on the hand contour [9].

To detect the gap between clung fingers, we first locate a



(a) (b)

Fig. 7. (a) Reference point, the center of the palm, the radial distance, and (b) distance transform of the hand.



(a) A hand contour with 9 minima (b) Radial distance

Fig. 8. Candidate starting points for separating clung fingers.

starting point of the gap (termed as separating point). From Fig. 8, we can see that both separating points and finger valleys are minimum extremities in the radial distance representation. But different from finger valleys, the separating point usually has a larger radial distance. Thus, we regard minima with large radial distance as candidate separating points. If we found a straight and sufficiently long line starting from the separating point and extending toward the palm, the line will be regarded as a finger gap. In our implementation, the lines are found using active contour model [10]. Given a initial contour, the algorithm can find a salient contour nearby.

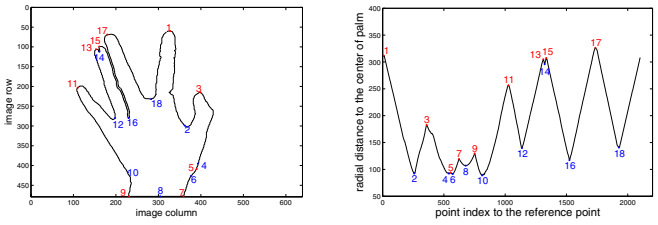
We model a finger gap as a curve with one end fixed. Since finger gap is usually darker than other part of hand, we use the grayscale image as energy image, namely, image intensity is set as energy. The initial contour is a straight line with one end fixed in one of those candidate separating points. The other end of the line is hard to be determined. So we search lines in a small range and the line with lowest cumulative gray-value is adopted as the initial contour.

In the example illustrated in Fig. 8(a), the seventh and eighth minima have relatively large radial distance, thus are chosen as the candidate separating points. The result of gap detection are shown in Fig. 6(b). The line starting from seventh minimum (the green one) is not straight and neglected. The line starting from eighth minimum (the blue one) is adopted as the gap. The hand contour is then updated as shown in Fig. 6(c).

C. Finger Tips and Valleys Extraction

Given a hand contour, we first detect hand extremities (local minima and maxima), and then choose finger tips and valleys from them.

Pan et al. [11] detect the finger tips using curvature based detection techniques. Since this method is very sensitive to noise, we use another character of finger tips and valleys. Fig. 9



(a) Hand contour (b) Radial distance

Fig. 9. Extremities in the hand contour and the radial distance curve.

(a) (b)

Fig. 10. Postures with too many or too few numbers of extremities.

shows the contour and radial distance of the segmented hand (Fig. 6(c)). Maxima on the curve are regarded as candidate finger tips, and minima are regarded as candidate finger valleys. Let $L^t = \{n_1, \ldots, n_{N^t}\}$ be the list of extremities at tth iteration, where n_i denotes the extremity i, and N^t denotes the number of extremities in the list at tth iteration. Let $d_{i,j}$ be the index distance (the difference of index when counting from reference point) between extremity i and extremity j, and $d(L^t) = \{d^t_{i,i+1} | i = 1, 2, \ldots, N^t - 1\}$ be the index distance between all the adjacent pairs in L^t. For spatial distance, the notations are $D_{i,j}$ and $D(L^t)$ respectively, where $D_{i,j}$ denotes the spatial distance between n_i and n_j which are the nearest neighbor pair of the same type (maximum or minimum).

If there are too many extremities (Fig. 10(a)), the posture will be rejected directly without performing the subsequent steps. Too many extremities may be due to that the quality of hand contour is poor, or more than one hand appears in the image. Similarly, if less than 9 extremities are found (Fig. 10(b)), the posture will be rejected too.

We could observe that this technique is likely affected by two kinds of unwanted extremities, one is caused by irregular contour, the other is produced by the forearm. The first kind noisy extremities usually occur between two true extremities (namely, finger tips and valleys), and will decrease the index distance between the two extremities. On the contrary, extremities in the forearm region regularly have a large index distance and a large spatial distance (see Fig. 9) from other extremities.

For example, in Fig. 9, $L^0 = \{n_1, \ldots, n_{18}\}$. n_{14} and n_{15} in Fig. 9 are caused by irregular contour. Index distances between them $d_{14,15}$ is significantly less than $d_{12,13}$. Meanwhile, $n_4, \ldots, n_{10}$ belong to the latter kind of noisy extremities. They are far away from other finger tips, and near to or in the boundary of image. Those extremities in the bottom boundary of image (n_7,n_8,n_9) can be excluded by a simple rules. But this method is not robust enough since the threshold is difficult to determine (n_4,n_5,n_6 and n_{10} have relatively long distance to the boundary). So we define an energy function to evaluate the extremities as follows.

Generally speaking, the two kinds of noisy extremities mentioned above will both increase the variance of $d(L^t)$ and $D(L^t)$, thus extremities decreasing the variance most are more likely to be a outlier. That is, the difference between variance of original list and obtained list will contain much information about the energy of an extremity. We designate the difference between index variance of the two list by $\delta^t_i = var(d(L^t)) - var(d(L^t - \{n_i\}))$ when deleting extremity i. For spatial distance, that is $\Delta^t_i = var(D(L^t)) - var(D(L^t - \{n_i\}))$. When n_i is deleted from L^t, the obtained list is designated by $L^t - \{n_i\}$.

The energy of n_i can be computed as:

$$e1^t(i) = \frac{\delta^t_i - \min_{k=1}^{N^t}(\delta^t_k)}{\max_{k=1}^{N^t}(\delta^t_k) - \min_{k=1}^{N^t}(\delta^t_k)}, \quad (1)$$

$$e2^t(i) = \frac{\Delta^t_i - \min_{k=1}^{N^t}(\Delta^t_k)}{\max_{k=1}^{N^t}(\Delta^t_k) - \min_{k=1}^{N^t}(\Delta^t_k)}, \quad (2)$$

$$e^t(i) = e1^t(i)^2 + e2^t(i)^2. \quad (3)$$

Generally, maximum and minimum extremities appear alternately in the list. However, this may not always be the case due to noise. For those extremities whose adjacent extremities are of the same type (maximum or minimum), the one with higher energy will be deleted iteratively until all maximum and minimum extremities appear alternately in the list. After that, adjacent maximum and minimum extremities will only be deleted in pair. The definition of energy of a pair is similar to that of a single extremity. For instance, the energy $e(i, i+1)$ of two extremities n_i and n_{i+1} can be computed by the following equation (without lose of generality, n_k and n_{k+1} are thought to be adjacent):

$$\delta^t_{k,k+1} = var(d(L^t)) - var(d(L^t - \{n_i, n_j\})), \quad (4)$$

$$e1^t(i,j) = \frac{\delta^t_{i,j} - \min_{k=1}^{N^t-1}(\delta^t_{k,k+1})}{\max_{k=1}^{N^t-1}(\delta^t_{k,k+1}) - \min_{k=1}^{N^t-1}(\delta^t_{k,k+1})}, \quad (5)$$

$$e2^t(i,j) = (e2^t(i) + e2^t(j))/2, \quad (6)$$

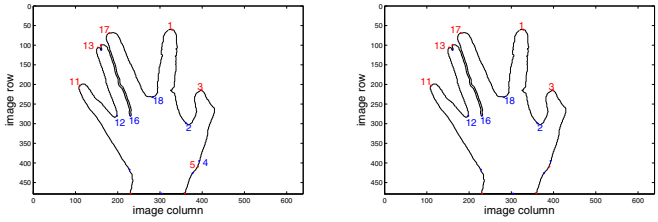
$$e^t(i,j) = e1^t(i,j)^2 + e2^t(i,j)^2. \quad (7)$$

In the case illustrated in Fig. 9, the extremities are deleted in this order. Contour where hand meets the bottom boundary are regarded as forearm region. Thus n_7,n_8,n_9 are deleted firstly. Minimums next to forearm region are usually in palm (the n_6 and n_{10}), and can be ignored, too. Then we get a list of extremities: $L^1 = \{n_{11}, n_{12}, n_{13}, n_{14}, n_{15}, n_{16}, n_{17}, n_{18}, n_1, n_2, n_3, n_4, n_5\}$. As all the extremities are paired, we need to delete them pairwise. Table I gives the the energy of the pairs in L^1 and L^2. As n_1 is our target finger tip that cannot be deleted, we do not take into account pair n_{18}, n_1 and pair n_1, n_2. According to our rules, pair n_{14}, n_{15} will be deleted from L^1.

TABLE I

ENERGY OF PAIRS OF ADJACENT EXTREMITIES AT THE 1ST AND 2ND ITERATIONS. THE PAIR REMOVED AT EACH ITERATION IS SHOWN IN RED.

pairs at 1st iteration	n_{11}, n_{12}	n_{12}, n_{13}	n_{13}, n_{14}	n_{14}, n_{15}	n_{15}, n_{16}	n_{16}, n_{17}	n_{17}, n_{18}	n_2, n_3	n_3, n_4	n_4, n_5
$e1^1(i,j)$	0.847	0.834	1	0.999	0.673	0	0	0.217	0.820	0.907
$e2^1(i,j)$	0.651	0.754	0.987	1	0.808	0.740	0.840	0	0.0896	0.538
$e^1(i,j)$	1.14	1.26	1.97	2.00	1.06	0.55	0.705	0.0470	0.681	1.11
pairs at 2nd iteration	n_{11}, n_{12}	n_{12}, n_{13}	–	–	n_{13}, n_{16}	n_{16}, n_{17}	n_{17}, n_{18}	n_2, n_3	n_3, n_4	n_4, n_5
$e1^2(i,j)$	0.901	0.421	–	–	0.0663	0	0.0965	0.330	0.981	1
$e2^2(i,j)$	0.861	0.971	–	–	1	0.974	0.794	0	0.419	0.842
$e^2(i,j)$	1.55	1.12	–	–	1.00	0.949	0.640	0.109	1.14	1.71



(a) Extremities in L^1 (b) Extremities in L^2

Fig. 11. Remaining extremities after 1st and 2nd iterations for the hand contour in Fig. 9.

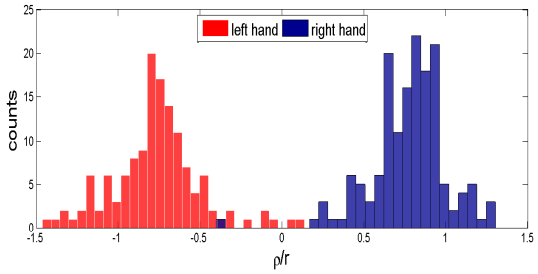


Fig. 12. Histograms of ρ/r values for left hand and right hand samples.

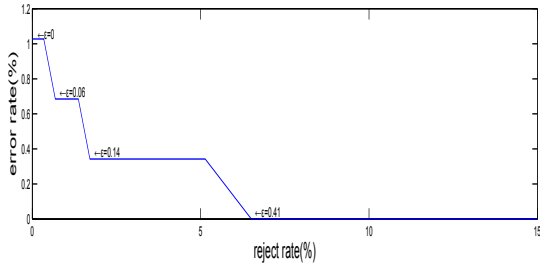


Fig. 13. The curve of reject rate vs. left/right classification error rate for different values of threshold ϵ.

Then $L^2 = \{n_{11}, n_{12}, n_{13}, n_{16}, n_{17}, n_{18}, n_1, n_2, n_3, n_4, n_5\}$ is obtained. Pair n_4, n_5 will be delted from L^2, so the final list of extremities is $L^3 = \{n_{11}, n_{12}, n_{13}, n_{16}, n_{17}, n_{18}, n_1, n_2, n_3\}$. which has 5 maxima and 4 minima. Fig. 11 shows the remaining extremities after 1st and 2nd iterations for the hand contour in Fig. 9.

D. Finger Position Estimation

After the processes mentioned above, we obtain 5 finger tips and 4 finger valleys, which are ordered from left to right. For convenience, the list of extremities is denoted as $L = \{n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9\}$. By the rank of target finger tip (namely, the finger tip pressed on the fingerprint scanner), we can easily figure out which finger it is. For example, in Fig. 11(b), the target finger tip is the fourth maximum in the obtained list, so the target finger may be a left index or right ring finger.

To determine which hand the target finger belongs to, we compare the relative height between little finger and ring finger to the relative height between thumb and index finger. Let $d_{left} = D_{3,2} - D_{1,2}$, $d_{right} = D_{7,8} - D_{9,8}$. For a left hand, $\rho = d_{left} - d_{right} < 0$; on the contrary, $\rho > 0$ if it is a right hand.

Moreover, if $d_{left} < 0$ or $d_{right} < 0$, there may be bent fingers. In such cases, the decision may not be reliable, and the posture will be rejected. To make the decision more reliable, we update the criterion as follows. If $\rho/r > \epsilon$, it is a right hand; if $\rho/r < -\epsilon$, it is a left hand; otherwise the posture will be rejected. r is the distance from the palm center to the nearest boundary point and used for normalization. Threshold $\epsilon \geq 0$ can be modified according to requirement. A trade off should be made between the reject rate and the error rate when changing ϵ.

IV. EXPERIMENT

Our database consisted of 424 samples from five persons using a webcam and a U.are.U 4500 fingerprint scanner. All the associated fingerprint images have passed the quality evaluation of this fingerprint scanner. Note that the proposed system is not limited to a specific fingerprint scanner. Each user provided several samples of each finger in both valid and invalid postures. There are 297 positive samples (valid postures), and 127 negative samples (invalid postures). We did a survey on the positive samples about value ρ/r. The histogram is represented in Fig. 12. We can see that two left hand samples have positive ρ/r value, and one right hand sample has a negative ρ/r value. By changing the value of ϵ, we can get the curve of reject rate vs. left/right classification error rate shown in Fig. 13.

All invalid postures except 3 of them are successfully rejected. See Fig. 14 for some successful examples. Three invalid postures which the algorithm did not reject are shown in Fig. 15. Cameras from multiple viewpoints may be needed to detect such invalid postures.

The finger positions of 287(96.6%) valid postures are cor-

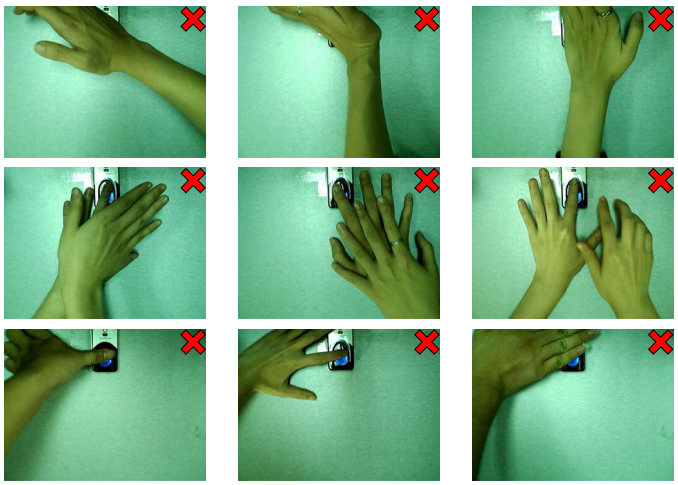


Fig. 14. Invalid postures which are successfully rejected.

Fig. 15. Invalid postures which are not rejected. While it looks like that the left hand presses on the fingerprint scanner, it is in fact a finger of the right hand (covered by the left hand) that presses on the fingerprint scanner.

rectly recognized (see Fig. 16 for some successful examples), 5(1.68%) of them are rejected, and the finger positions of the other 5(1.68%) are wrong. Among the 5 misclassified samples (see Fig. 17), 3 were caused by mistake in judgement of left or right hand, and the others were due to missing finger valleys (their fingers are too close to find the valleys).

V. Conclusion

Fingerprint acquisition in watch-list applications has to be closely monitored since the people in the watch-list have strong incentives to cheat the system. Currently, the task of monitoring fingerprint acquisition is performed by human. In this paper, we introduce a video surveillance system to monitor fingerprint acquisition. The proposed system can determine whether the posture of pressing fingers is valid or not and estimate the finger position if the posture is valid. With the proposed system, it is possible to reduce the manpower for monitoring fingerprint acquisition in watch-list applications. The proposed system may also be used to improve the security level of positive fingerprint recognition systems, since the adversary is now forced to use only a specific finger, rather than all one's fingers and palms.

The current system still has some limitations: 1) some common postures of pressing fingers (especially when hand is not stretched) are not supported now, 2) the system can still be cheated by some highly deceptive posture as shown in Fig. 15. We plan to overcome these limitations in the future.

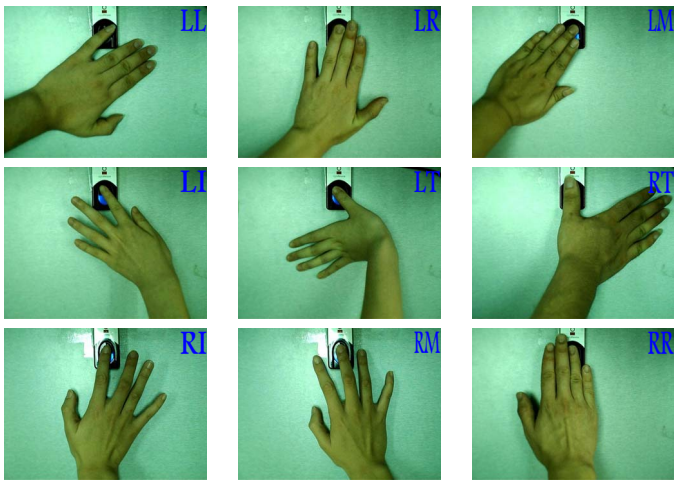


Fig. 16. Valid postures whose finger position estimates (shown on the top-right of the image) are correct.

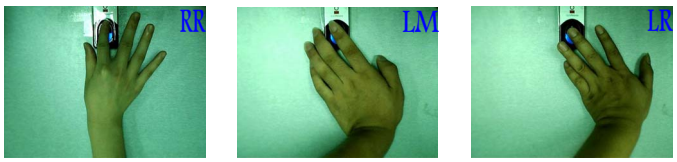


Fig. 17. Valid postures whose finger position estimates (shown on the top-right of the image) are wrong.

References

[1] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed. Springer, 2009.

[2] E. Tabassi, C. Wilson, and C. Watson, "Fingerprint image quality," NISTIR 7151, August 2004.

[3] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 52 – 73, 2007.

[4] A. Antonelli, R. Cappelli, D. Maio, and D. Maltoni, "Fake finger detection by skin distortion analysis," *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 3, pp. 360–373, 2006.

[5] S. Yoon, J. Feng, and A. K. Jain, "Altered fingerprints: Analysis and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012 (To Appear).

[6] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer, "Visual panel: Virtual mouse, keyboard and 3D controller with an ordinary piece of paper," in *Workshop on Perceptive User Interfaces (PUI)*, 2001, pp. 1–8.

[7] A. Sugiura and Y. Koseki, "A user interface using fingerprint recognition: Holding commands and data objects on fingers," in *Proceedings of ACM symposium on User Interface Software and Technology (UIST)*, 1998, pp. 71–79.

[8] J. Wang and J. Canny, "Fingersense: Augmenting expressiveness to physical pushing button by fingertip identification," in *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI)*, 2004, pp. 1267–1270.

[9] E. Yoruk, E. Konukoglu, B. Sankur, and J. Darbon, "Shape-based hand recognition," *IEEE Transactions on Image Processing*, vol. 15, pp. 1803–1815, 2006.

[10] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[11] Z. Pan, Y. Li, M. Zhang, C. Sun, K. Guo, X. Tang, and S. Zhou, "A real-time multi-cue hand tracking algorithm based on computer vision," in *IEEE Virtual Reality Conference (VR)*, March 2010, pp. 219 –222.