Walking to Singular Points of Fingerprints[1]

En Zhu[a], Xifeng Guo[a,*], Jianping Yin[b]

[a] *College of Computer, National University of Defense Technology, Changsha, 410073, China*
[b] *State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha, 410073, China*

Abstract

Singular point is an essential global feature in fingerprint images. Existing methods for singular points detection generally visit each pixel or each small image block to determine the singular point. That is to say, existing methods require scanning the image to compute a quantity at each pixel or block, and hence they are inevitably time-consuming. We propose a fast algorithm for detecting singular points by walking directly to them instead of scanning the image. Walking Directional Fields (WDFs) are established from the orientation field. Then following the walking directions on WDFs, we can rapidly walk to the singular points. The walking algorithm is extremely fast and easily implemented with acceptable accuracy. Further more, its accuracy can also be improved by combining with state-of-the-art methods: we can rapidly walk to a candidate singular point, then refine its location using existing more accurate method in the local area. Experimental results on datasets of SPD2010 and FVC validate the high efficiency and satisfactory accuracy of the proposed algorithm.

Keywords: walking, singular point, walking directional field

1. Introduction

Fingerprint singular points (upper core, lower core and delta), defined as where the orientation field is discontinuous or the ridge curvature is the highest, are essential for registration and identification (specially for image-based approach instead of minutiae-based approach) [1, 2]. Singular points detection methods have been well studied, including Poincaré index (PI) technique [3, 4, 5, 6, 7, 8], model-based technique [9, 10, 11, 12], complex filter technique [13, 14, 15, 16, 17, 18] and others [19, 20, 21, 22]. However, these methods are based on scanning process which consumes a lot of processing time as shown below.

The Poincaré index of a point is defined as the cumulative orientation differences counterclockwise along a simple closed path surrounding this point. For core, delta and

[1]**This is a preprint version. The final version can be accessed from** `http://dx.doi.org/10.1016/j.patcog.2016.02.015`

*Corresponding author. Tel.: +8673184573603; fax: +8673184575992.

Email address: `guoxifeng1990@163.com` (Xifeng Guo) non-singular point, the values of Poincaré index are π, $-\pi$ and 0 respectively. The Poincaré index method, first proposed by Kawagoe and Tojo [3], calculates Poincaré index for each point in the orientation field to determine whether it is a singular point. The Poincaré index method is the most classical way to detect singular points, because it can detect singular points accurately if the closed path is not too long. However it's sensitive to the noise of fingerprint image and easy to detect many spurious singular points. To improve the robustness, different strategies are used, such as replacing closed line integral with surface integral [4], fusing global features [5, 6, 7] and combining with other local characteristics [8]. They surely need extra processing time compared with the original Poincaré index method.

Model-based methods use mathematical formula to represent the orientation field and detect singular points by analysing the corresponding model. The most popular model is the zero-pole model, first introduced by Sherlock and Monro [9], which reveals that the orientation at a point is determined by the number and positions of singular points (cores as zeros and deltas as poles) plus a constant correction term. According to this constraint, special relationship between singular points and their neighbor points can be derived, then a Hough transform method [11] or a modified convergence index filter [12] is utilized to detect singular points. Although the accuracies of these methods are satisfactory, the efficiencies are not promising because Hough transform needs to scan all pixels to fill the parameter space and the modified convergence index filter needs to be applied to each pixel to find maxima and minima. Wang et al. [10] proposed a singular point detection method using fingerprint orientation model based on 2D Fourier series expansions (FOMFE). The FOMFE detects singular point by analysing the attributes of the characteristic matrix $\mathbf{A}$, while the construction of $\mathbf{A}$ dominates the computation as it is conducted at each point on the orientation field, which constrains the efficiency of this method.

Complex filter technique [13, 14, 15, 16, 17, 18] design two complex filter to capture the symmetry properties of core and delta respectively. Then the convolution of the complex orientation field image with each complex filter is computed and the point with high filter response is taken as the singular point. Similar to Poincaré index method, complex filter technique can be accurate when the size of complex filter is small but will report more spurious singular points. If the filter size is too large, the accuracy and efficiency degrade. Nilsson and Bigun [15] makes a tradeoff between detection rate and false alarm rate by applying complex filter to the orientation field in multiple resolution scales, but more time is needed.

Other methods like sine-map-based technique [19], shape analysis method [20], multi-scale Gaussian filter method [21] and multi-scale orientation entropy method [22] all need to visit every pixel or block to extract enough information to detect singular points.

To sum up, almost all of state-of-the-art methods for detecting fingerprint singular point can hardly avoid visiting every pixel or small block to determine whether it is a candidate singular point, which fundamentally constrains the efficiency of these methods from being improved extraordinarily.

Based on the analysis of orientation fields derived from Zero-pole Model and those estimated from real fingerprint images, in this paper we propose a novel fast algorithm termed *walking* algorithm which can directly walk to the singular points on some defined Walking Directional Fields (WDFs) without scanning the fingerprint image. The rotation characteristics of WDFs are used to make the walking algorithm insensitive to the rotation of fingerprint image. We also introduce a simple strategy to combine the walking algorithm with state-of-the-art method to improve the accuracy. The code of the walking algorithm can be available at `http://cn.mathworks.com/matlabcentral/fileexchange/54588-walking-algorithm-for-sp-detection` or at github: `https://github.com/XifengGuo/Walking-to-SP.git`. The rest of this paper is organized as follows. Section 2 introduces the WDFs which are the basis of the proposed algorithm. Then the walking algorithm is described in Section 3. Section 4 shows some experimental results. Finally we give conclusions in Section 5.

2. Walking Directional Fields

2.1. Zero-Pole Model

Zero-pole Model for orientation field estimation was first proposed by Sherlock and Monro [9]. This model considers core as zero and delta as pole in the complex plane, and uses the argument of complex function $p(z)$ to approximate the orientation $o(z)$ of a point z in the fingerprint. $p(z)$ and $o(z)$ are defined as follows:

$$\begin{aligned} p(z) &= \sqrt{e^{2jo_\infty}\frac{(z-z_{c1})(z-z_{c2})\cdots(z-z_{cm})}{(z-z_{d1})(z-z_{d2})\cdots(z-z_{dn})}}, & (1)\\ o(z) &= \arg(p(z)) \mod \pi. & (2)\end{aligned}$$

where z_{ci} and z_{dj} are the ith core and jth delta of fingerprint respectively, and o_∞ is a constant correction term. According to the knowledge of complex function, orientation at point z is the sum of effects of all cores and deltas, so Eq. (2) can be rewritten as

$$o(z) = o_\infty + \frac{1}{2}\left(\sum_{i=1}^{m}\arg(z-z_{ci}) - \sum_{j=1}^{n}\arg(z-z_{dj})\right) + k\times\pi, k\in\mathbb{N}. \tag{3}$$

According to [11], orientation field in area Ω is mainly determined by its closest singular point, while the other singular points just have a constant influence on Ω. So the orientation of point z near core point z_c and that near delta z_d can be respectively approximated by Eq. (4) and (5).

$$\begin{aligned} o_c(z) &= o_{c\infty} + \frac{1}{2}\arg(z-z_c) + k\times\pi, k\in\mathbb{N}, & (4)\\ o_d(z) &= o_{d\infty} - \frac{1}{2}\arg(z-z_d) + k\times\pi, k\in\mathbb{N}. & (5)\end{aligned}$$

Fig. 1 shows the orientation fields around a core and delta generated by the above two equations with $z_c = (50,50), z_d = (50,50)$ and $o_{c\infty} = o_{d\infty} = 0$. We define the directions of a core and a delta as the positive directions of the x-axis in Fig. 1. Then when o_∞ changes, the directions of a core and a delta will equal $2o_{c\infty}$ and $(2o_{d\infty}/3)$ respectively. The proof is as follows:

PROOF. Suppose the image in Fig. 1a is rotated counterclockwise by θ round the core, then the direction of the core should be θ. Choose a point z in the origin image and the counterpart z' in the rotated image, then we have

$$o'_c(z') - o_c(z) = \arg(z'-z_c) - \arg(z-z_c) = \theta, \tag{6}$$

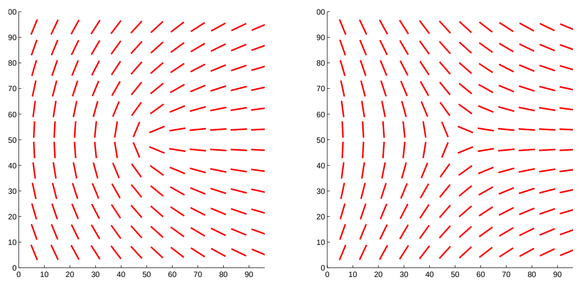

(a) (b)

Figure 1: Orientation field around (a) a core and (b) a delta, simulated by Zero-pole Model.

where $o'_c(z')$ denotes the orientation of z' in the rotated image. According to Eq. (4), z' satisfies

$$o'_c(z') = o'_{c\infty} + \frac{1}{2}\arg(z' - z_c) + k' \times \pi, k' \in \mathbb{N}, \tag{7}$$

then it subtracts Eq. (4) to get

$$\begin{aligned} o'_c(z') - o_c(z) = \; & o'_{c\infty} - o_{c\infty} + \frac{1}{2}(\arg(z' - z_c) - \arg(z' - z_c)) \\ & + (k' - k) \times \pi, (k' - k) \in \mathbb{N}. \end{aligned} \tag{8}$$

Substitute Eq. (6) into Eq. (8), we can get

$$\theta = 2(o'_{c\infty} - o_{c\infty}). \tag{9}$$

As we know, for the origin image there is $o_{c\infty} = 0$, so the direction of the core z_c is $2o'_{c\infty}$. We can prove the direction of the delta z_d is $(2o'_{d\infty}/3)$ in a similar way.

For cores, when $-\pi \leq 2o_{c\infty} < 0$, we call them upper cores, and when $0 \leq 2o_{c\infty} < \pi$, we call them lower cores. When $2o_{c\infty} = -\pi/2$ and $2o_{c\infty} = \pi/2$, we obtain the orientation fields for ideal upper core and lower core respectively, as shown in Fig. 2a and 2d. For deltas, the ideal orientation field is obtained when $2o_{d\infty}/3 = \pi/2$, as shown in Fig. 2g. The orientations of points around the ideal upper core z_{uc}, lower core z_{lc} and delta z_d are given by Eq. (10), (11) and (12) respectively.

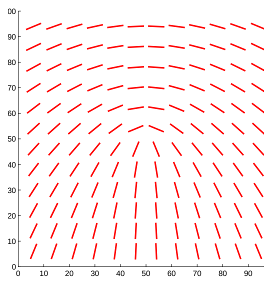
$$o_{uc}(z) = \frac{1}{2}\arg(z - z_{uc}) - \frac{\pi}{4} + k \times \pi, k \in \mathbb{N}, \tag{10}$$

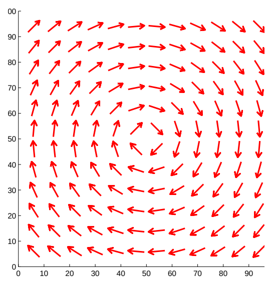
$$o_{lc}(z) = \frac{1}{2}\arg(z - z_{lc}) + \frac{\pi}{4} + k \times \pi, k \in \mathbb{N}, \tag{11}$$

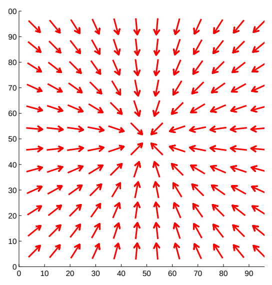
$$o_d(z) = -\frac{1}{2}\arg(z - z_d) + \frac{3\pi}{4} + k \times \pi, k \in \mathbb{N}, \tag{12}$$

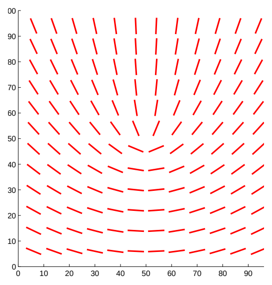
2.2. Analysis of Directional Fields

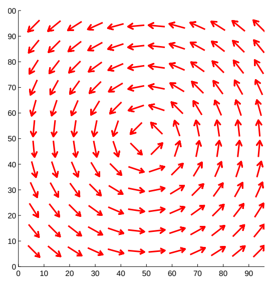
In the local area around singular points we obtained orientation field estimation equations (see Eq. (10), (11) and (12)) according to Zero-pole Model, now we calculate

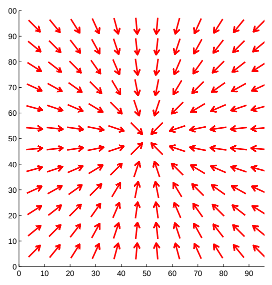

(a)

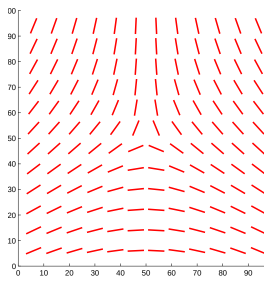

(b)

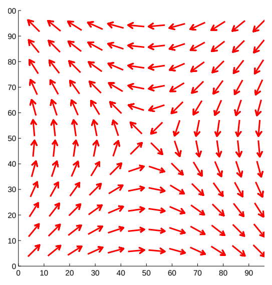

(c)


(d)


(e)


(f)


(g)


(h)

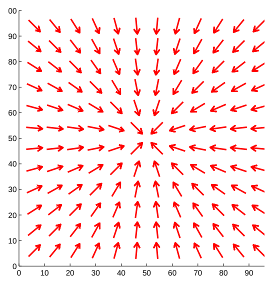

(i)

Figure 2: Orientation fields, SDFs and WDFs of (a)-(c) ideal upper core, (d)-(f) ideal lower core and (g)-(i) ideal delta simulated by Zero-pole Model.

the Squared Directional Field (SDF) [4] by doubling $o(z)$ at each pixel, i.e.

$$\begin{aligned} SDF_{uc}(z) &= 2o_{uc}(z) = \arg(z - z_{uc}) - \frac{\pi}{2}, \quad (13) \\ SDF_{lc}(z) &= 2o_{lc}(z) = \arg(z - z_{lc}) + \frac{\pi}{2}, \quad (14) \\ SDF_{d}(z) &= 2o_{d}(z) = -\arg(z - z_{d}) + \frac{3\pi}{2}. \quad (15) \end{aligned}$$

These SDFs are illustrated in Fig. 2b, 2e and 2h. By observing these pictures, we can notice that the upper core is surrounded by some clockwise circles in SDF_{uc}. Therefore, if each direction in SDF_{uc} minus $\pi/2$, as shown in Fig. 2c, the upper core will be pointed to by all surrounding directions in the new directional field. And the lower core is surrounded by counterclockwise circles in SDF_{lc}, so adding $\pi/2$ to each direction in SDF_{lc} can make the lower core be pointed to by its surrounding directions. However, the characteristics of SDF_d are not so obvious.

On the other hand, by analysing the above equations, we can get more clear and understandable inferences. We all know the argument, $\arg(z - z_{uc})$, has the direction from z_{uc} to z, so $(\arg(z - z_{uc}) \pm \pi)$ makes the direction of each z point to z_{uc}. Rewrite Eq. (13) as

$$2o_{uc}(z) - \frac{\pi}{2} = \arg(z - z_{uc}) - \pi, \quad (16)$$

then from any point z near z_{uc}, we can always walk along the above direction to the upper core z_{uc}. This direction is called *walking direction*, and the direction field is defined as Walking Directional Field (WDF) of upper core, represented by WDF_{uc},

$$WDF_{uc}(z) = 2o_{uc}(z) - \frac{\pi}{2}. \quad (17)$$

Analogically, we can define WDF of lower core and delta as

$$\begin{aligned} WDF_{lc}(z) &= 2o_{lc}(z) + \frac{\pi}{2}, \quad (18) \\ WDF_{d}(z) &= -2o_{d}(z) + \frac{\pi}{2}. \quad (19) \end{aligned}$$

These three kind of WDFs are illustrated in Fig. 2c, 2f and 2i.

Although the characteristics of WDFs derived from Zero-pole Model are so perfect, they will not be so ideal for real fingerprint images because of multiple singular points and low quality. In the following part, we will analyse WDFs estimated from real fingerprint images.

First of all, the orientation field, Θ, of fingerprint image should be estimated. There are many methods for orientation field estimation, such as gradient-based methods [23, 24, 25], model-based method [26, 27, 28] and others [29, 30]. In this paper, we estimate Θ using the publicly available Matlab source codes [31], which follows the basic algorithm in [23].

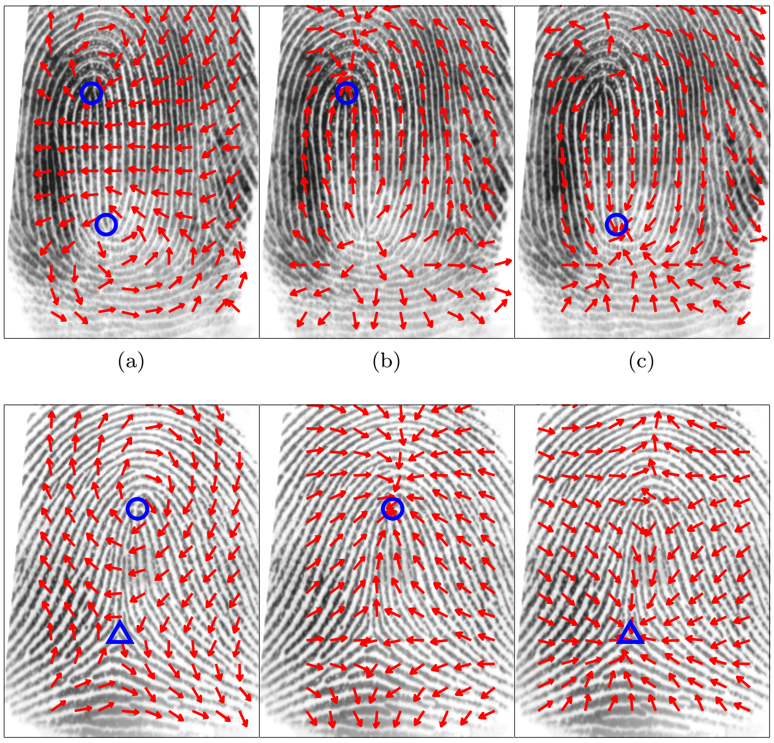
After estimating the orientation field Θ, the $o_{uc}(z)$, $o_{lc}(z)$ and $o_d(z)$ in Eq. (17), (18) and (19) are replaced by $\Theta(i,j)$ to derive the WDFs of real fingerprint image:

$$\begin{aligned} WDF_{uc}(i,j) &= 2\Theta(i,j) - \frac{\pi}{2}, & (20)\\ WDF_{lc}(i,j) &= 2\Theta(i,j) + \frac{\pi}{2}, & (21)\\ WDF_{d}(i,j) &= -2\Theta(i,j) + \frac{\pi}{2}. & (22) \end{aligned}$$

As same as [4], SDF is defined as

$$SDF(i,j) = 2\Theta(i,j). \tag{23}$$

Some examples of SDF and WDFs estimated from real fingerprint images are shown in Fig. 3, from which we can validate that the characteristics of SDF and WDFs in the



(a) (b) (c)

(d) (e) (f)

Figure 3: Examples of different directional fields. (a)-(c) SDF, WDF_{uc} and WDF_{lc} derived from the same image. (d)-(f) SDF, WDF_{uc} and WDF_{d} derived from another image.

corresponding singular area comply with the analysis of the directional fields derived from Zero-pole Model (see Fig. 2). Further observations for Fig. 3 reveal the following interesting phenomena.

1) From any point on WDF_{uc}, we can always walk to either the upper core or the outside of the fingerprint foreground. Analogically, we can walk to the lower core on WDF_{lc} and to the delta on WDF_d.
2) If the lower core is detected, we can walk to the upper core from the upper area of the lower core on WDF_{uc} with highest probability (see Fig. 3b) and vice versa (see Fig. 3c).
3) If the delta is detected, we can walk to the upper core from the upper area of the delta on WDF_{uc} with highest probability (see Fig. 3e and vice versa (see Fig. 3f).

Therefore, it is feasible to design an algorithm of detecting singular points by directly walking from some points to them on WDFs, which should be extremely fast because this algorithm only need to visit tens of points without scanning the image.

2.3. Rotation Analysis of WDFs

Intuitionally, the proposed WDFs are sensitive to rotation of fingerprint image, we analyse this characteristic in this subsection.

Section 2.1 has proved the directions of core and delta are $2o_{c\infty}$ and $(2o_{d\infty}/3)$, and the directions of ideal upper core, lower core and delta are $2o_{c\infty} = -\pi/2$, $2o_{c\infty} = \pi/2$ and $2o_{d\infty}/3 = \pi/2$ respectively. Hence the rotation angles of the ideal upper core, lower core and delta are

$$\begin{aligned}\phi_{uc} &= 2o_{c\infty} + \frac{\pi}{2}, &(24)\\ \phi_{lc} &= 2o_{c\infty} - \frac{\pi}{2}, &(25)\\ \phi_d &= \frac{2}{3}o_{d\infty} - \frac{\pi}{2}. &(26)\end{aligned}$$

Some examples of WDFs derived from Zero-pole Model with different rotation angles are shown in Fig. 4. Let $2T$ be the period of the directions of singular points, then $T = \pi$ and $T = \pi/3$ for core and delta respectively. Fig. 4 indicates $T/2$ is the critical rotation angle between convergence and divergence. While for real fingerprint images, as shown in Fig. 5, it is hard to ensure the convergence when the rotation angle is near $T/2$. So we conservatively choose $T/4$ as the critical rotation angle for stable convergence. In other words, if the rotation angle is in range $[-T/4, T/4]$, we can walk along the walking direction to the singular point stably. Specifically, the stable range is $[-\pi/4, \pi/4]$ for core and $[-\pi/12, \pi/12]$ for delta. Note that the rotation angle mentioned here is relative to the direction of singular point, not necessarily to be consistent with the angle of fingerprint image rotation. Therefore, our WDFs may achieve better performance in singular points detection if the fingerprint images are pre-aligned to the rotation range $[-\pi/12, \pi/12]$ relative to deltas or $[-\pi/4, \pi/4]$ relative to cores.

However, if the rotation angle ϕ respect to a certain singular point is estimated or chosen by sampling, the WDFs can be easily modified to

$$\begin{aligned}WDF_{uc}(i,j) &= 2\Theta(i,j) - \frac{\pi}{2} - \phi_{uc}, &(27)\\ WDF_{lc}(i,j) &= 2\Theta(i,j) + \frac{\pi}{2} - \phi_{lc}, &(28)\\ WDF_d(i,j) &= -2\Theta(i,j) + \frac{\pi}{2} + 3\phi_d. &(29)\end{aligned}$$

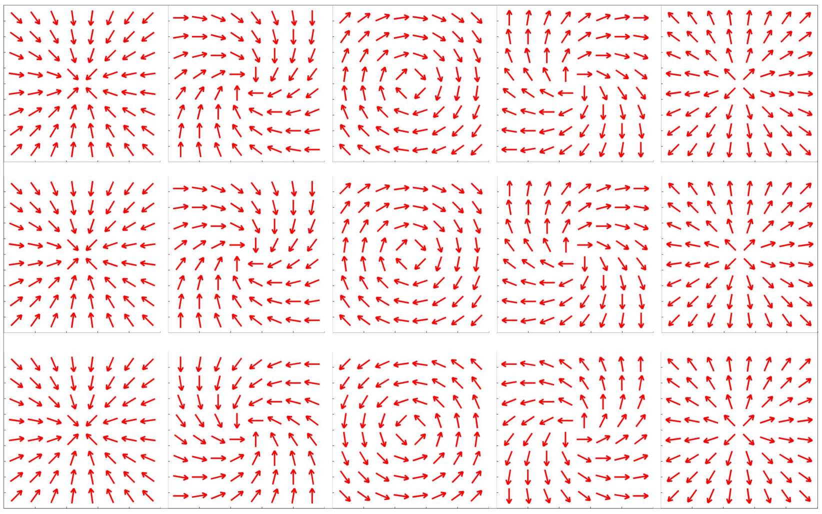


Figure 4: WDF_{uc}, WDF_{lc} and WDF_d (top to bottom) derived from Zero-pole Model rotated by 0, $T/4$, $T/2$, $3T/4$ and T (left to right).

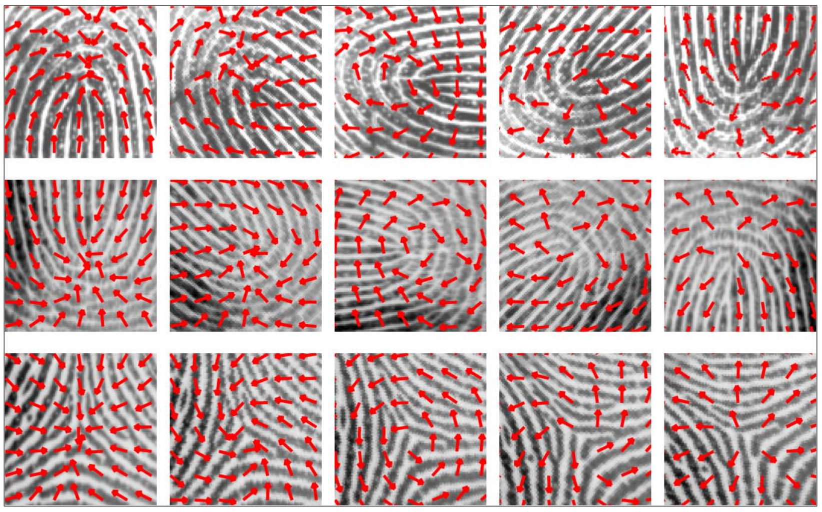


Figure 5: WDF_{uc}, WDF_{lc} and WDF_d (top to bottom) estimated from real fingerprint image rotated by 0, $T/4$, $T/2$, $3T/4$ and T (left to right).

Here we give the proof for Eq. (27), the other two equations can be proved in the same way. Substitute Eq. (24) into Eq. (4), we can get

$$o_{uc}(z) = \frac{1}{2}\arg(z - z_{uc}) + \frac{\phi_{uc}}{2} - \frac{\pi}{4} + k \times \pi, k \in \mathbb{N}, \tag{30}$$

then both sides of equal sign are multiplied by 2 and reform the result to

$$2o_{uc}(z) - \frac{\pi}{2} - \phi_{uc} = \arg(z - z_{uc}) - \pi. \tag{31}$$

Compare it with Eq. (16) and Eq. (17), we can get

$$WDF_{uc}(z) = 2o_{uc}(z) - \frac{\pi}{2} - \phi_{uc}. \tag{32}$$

Transform from complex plane to image plane and replace o_{uc} with Θ, we can obtain Eq. (27) finally.

These modified equations can help deal with large rotated fingerprint images. The details will be discussed in Section 3.4.

3. Walking to Singular Points on WDFs

From a starting point, we may walk to the singular point on WDFs and the details of walking strategy will be discussed in Section 3.1 (Algorithm 1). However, the detected singular point can be spurious, so we propose an approach to discriminate the spurious singular point in Section 3.2. On the other hand, we cannot always walk to a singular point from any starting point. To improve the detection rate, more than one starting point are tried using Algorithm 1. This progress is summarized in Section 3.3 (Algorithm 2). At last, in Section 3.4 we handle the detection failure caused by fingerprint rotation using WDF with different modified term ϕ (see Eq. (27), (28) and (29)) and give the whole walking algorithm in Algorithm 3. The block diagram of the walking algorithm is illustrated in Fig. 6.

3.1. Walking From a Given Starting Point

In Section 2.2, we established three WDFs (WDF_{uc}, WDF_{lc} and WDF_d) and noticed that from certain point we may walk directly to the corresponding singular point. Here the details of walking scheme are described.

Without loss of generality, we take WDF_{uc} as an example. Suppose the starting point is $P_0 = (x_0, y_0)$, the length of walking step is d ($d = 7$ pixels in our study) and the direction of the starting point is $\alpha_0 = WDF_{uc}(y_0, x_0)$. Then the new position $P_1 = (x_1, y_1)$ after walking one step can be calculated by

$$\begin{aligned} x_1 &= x_0 + [d \cdot \cos(\alpha_0)], \\ y_1 &= y_0 + [d \cdot \sin(\alpha_0)], \end{aligned} \tag{33}$$

where $[\cdot]$ is the round operator. Keep walking to the kth point $P_k = (x_k, y_k)$, calculated by

$$\begin{aligned} x_k &= x_{k-1} + [d \cdot \cos(\alpha_{k-1})], \\ y_k &= y_{k-1} + [d \cdot \sin(\alpha_{k-1})], \end{aligned} \tag{34}$$

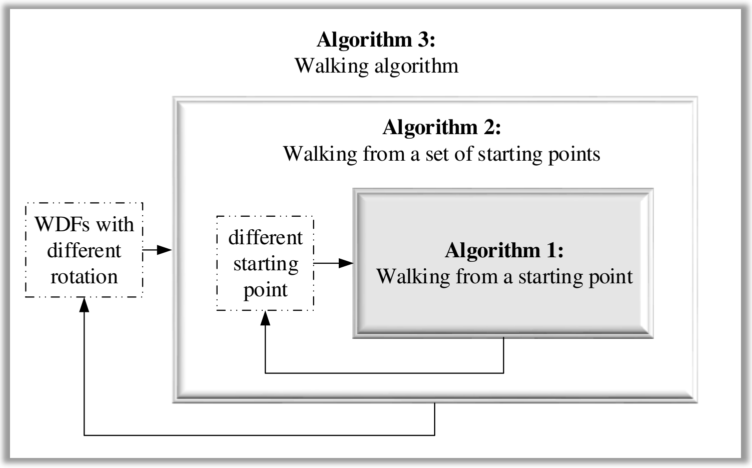

Algorithm 3:
Walking algorithm
Algorithm 2:
Walking from a set of starting points
WDFs with different rotation
different starting point
Algorithm 1:
Walking from a starting point

Figure 6: The block diagram of our walking algorithm.

where $k \geq 1$, (x_{k-1}, y_{k-1}) is the coordinate of the $(k-1)$th point P_{k-1} and α_{k-1} is the walking direction at P_{k-1}, given by

$$\alpha_{k-1} = WDF_{uc}\left(y_{k-1}, x_{k-1}\right). \tag{35}$$

Let $Path = \{P_0, P_1, \cdots, P_k\}$ denote the walking path and δ_k denote the least distance between the end point of $Path$, P_k, and the other points in $Path$, i.e.

$$\delta_k = \min\left\{\sqrt{(x_i - x_k)^2 + (y_i - y_k)^2} \;\middle|\; i = 0, 1, \cdots, k-1\right\}. \tag{36}$$

As we can see in Fig. 3b and 3e, when we walk into the singular region of the upper core, we can not walk out, which leads to the gradual decrease of δ_k. Give a threshold T_δ ($T_\delta = d/2$ in our study), then we will stop walking once $\delta_k < T_\delta$, where we say a loop occurs on the walking path. The center point of the detected loop serves as a candidate for the upper core, P_{uc}, obtained by

$$\begin{aligned} x_{uc} &= \frac{1}{k - i_0 + 1}\sum_{j=i_0}^{k} x_j, \\ y_{uc} &= \frac{1}{k - i_0 + 1}\sum_{j=i_0}^{k} y_j, \end{aligned} \tag{37}$$

where (x_{uc}, y_{uc}) gives the position of P_{uc}, (x_j, y_j) is the coordinate of P_j, which is on the detected loop of $Path$, and (x_{i_0}, y_{i_0}) is the coordinate of point P_{i_0} which is the beginning point of the detected loop on $Path$, satisfying

$$\delta_k = \sqrt{(x_{i_0} - x_k)^2 + (y_{i_0} - y_k)^2}. \tag{38}$$

However, it is possible to walk to the outside of fingerprint foreground, in which case we declare that the attempt to walk to singular point from the given starting point P_0 comes into a failure. The process of walking from a given starting point is summarized in Algorithm 1 which is also applicable to walking on WDF_{lc} and WDF_d.

Algorithm 1: Walking from a given starting point.

```
   Input: $WDF_{uc}$, $P_0$
   Output: $P_{uc}$ or $Failure$
1  $Path = \{P_0\}$;
2  for $k = 1$ to $\infty$ do
3      Get $P_k$ according to Eq. (34);
4      if $P_k$ is not in the foreground then
5          return $failure$;
6      $Path = Path \cup P_k$;
7      Get $\delta_k$ using Eq. (36);
8      if $\delta_k < T_\delta$ then
9          return $P_{uc}$ (calculated by Eq. (37));
```

Some examples of the process of walking from some given points to the corresponding singular points on WDFs are shown in Fig. 7. Apparently, if we can walk to the singular point from a proper starting point, the walking path will not be too long. While each step just need very simple calculation as described above, the whole process will be significantly fast.

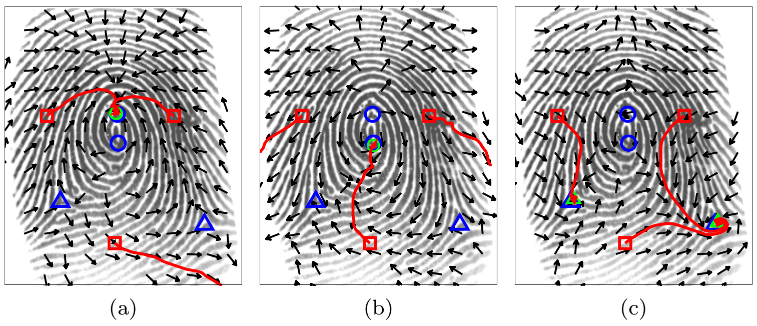


Figure 7: Walking from the given starting points on (a) WDF_{uc}, (b) WDF_{lc} and (c) WDF_d. Starting point and detected singular point are marked with red "□" and green "◯" ("△") respectively, and blue markers are ground truth.

3.2. Removing Spurious Singular Points

We described the process of walking from a given starting point to the singular point on the corresponding WDF in Section 3.1. Although we can walk to the genuine singular point in most cases, spurious singular points still can stop the walking process in noisy

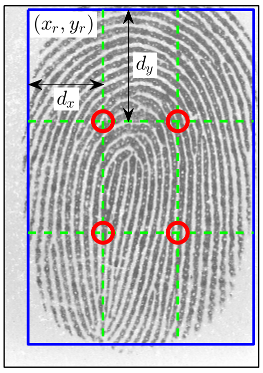


Figure 8: Sampling starting points with $n = 2$.

image. By analysing the local area of spurious singular points, we notice that from some neighbor points we can not walk back to the same local area. According to this characteristic, we define the neighborhood of a candidate singular point as a circular area with radius $R = 2d$ (length of two walking steps) and neighbors as four quadrantal points. Therefore, from each neighbor a new candidate singular point can be found using Algorithm 1. If all of the four new candidate singular points locate in the defined neighborhood, the original candidate singular point is deemed as genuine singular point; otherwise, it is deemed as spurious one.

3.3. Choosing Starting Points

In Section 3.1, we described the process of walking to the candidate singular point from a given starting point, then proposed an approach to determine whether a candidate singular point is spurious one in Section 3.2. Unfortunately, we can not make sure that from any starting point Algorithm1 always outputs a genuine singular point. It is a natural thought to improve the detection rate by selecting more starting points. A simple selection strategy is used in this section, i.e. choose starting points by sampling in fingerprint foreground and collect them into a set S. Suppose the left top corner point and the size of the bounding rectangle of fingerprint foreground are (x_r, y_r) and $w_r \times h_r$ respectively, then we sample points with step length along x-axis, $d_x = \lfloor w_r/(n+1) \rfloor$, and step length along y-axis, $d_y = \lfloor h_r/(n+1) \rfloor$, where $\lfloor \cdot \rfloor$ is floor operator and n ($n = 2$ in our study) denotes the number of points to be sampled along x-axis or y-axis. So

$$S = \left\{ (x, y) \middle| x = x_r + i \cdot d_x, y = y_r + j \cdot d_y, M(y, x) = 1, i, j = 1, 2, \cdots, n \right\} \tag{39}$$

where $M(y, x) = 1$ denotes that (x, y) is in the foreground. Therefore the number N of points in S is not more than n^2, i.e. $N \leq n^2$. Fig. 8 shows an example of sampling points with $n = 2$.

Therefore for each starting point $P_i \in S$, a result can be obtained using Algorithm 1 and then be determined whether it is a spurious singular point according to Section 3.2. For cores, at most one upper core and one lower core are detected. While more than one delta may exist, so for each starting point in S a candidate delta is tried to detect. If there are many deltas in a circular region with radius 10 pixels, these deltas will be replaced by their average point. The progress is summarized in Algorithm 2, which is also applicable to lower core and delta detection.

Algorithm 2: Walking from a set of starting points.

```
    Input: $WDF_{uc}$ (Walking Directional Field of upper core)
    Output: $P_{uc}$ (upper core)
1   for each starting point $p \in S$ do
2       Get a candidate $P_{uc}$ by Algorithm 1 with $p$ and $WDF_{uc}$;
3       if $P_{uc}$ is not spurious (see Section 3.2) then
4           return $P_{uc}$;
```

3.4. Dealing with Rotation of Fingerprints

If the fingerprint images are in the rotation range $[-T/4, T/4]$ with respect to the direction of singular point, where $T = \pi$ for core and $T = \pi/3$ for delta, our Algorithm 1 can work well. But large rotated images maybe exist, in which case Algorithm 1 might fail to detect the singular point for any starting point. This section describes how to use the rotation properties of WDFs (see Section 2.3) to deal with large rotated fingerprints.

As the period of direction of singular point is $2T$, so any rotation angle ϕ can be converted to $[-T, T]$. And for any rotation angle $\phi \in [-T, T]$, we can always find a number $k \in \mathbb{N}$ to satisfy $(\phi + kT/2) \in [-T/4, T/4]$. So we let correction term ϕ (ϕ_{uc}, ϕ_{lc} and ϕ_d) in Eq. (27), (28) and (29) successively be 0, $\frac{T}{2}$, T and $\frac{3T}{2}$, then no singular point in theory can be missed.

However, these trials not only costs more processing time but also tends to detect more spurious singular points. By observing fingerprints, we believe the following statements are always true:

1. At most 2 cores and 2 deltas can be found in a fingerprint image.
2. If there are 2 cores in a fingerprint image, their directions are always approximately opposite
3. If there are 2 deltas in a fingerprint image, their directions are always approximately equal.

Therefore, for a certain value of correction term ϕ, if the upper core is detected on WDF_{uc}, the lower core (if exists) can be detected on WDF_{lc} meanwhile, and vice versa. Deltas are also detected simultaneously. So we need not try to detect cores (deltas) for the new ϕ if cores (deltas) have been detected by using last ϕ.

When the fingerprint image is rotated, there will be no obvious difference between the upper core and lower core. This is also reflected in Fig. 4 where the first row and the second row have the same shapes. Therefore we use WDF_{c1} and WDF_{c2} represent two kinds of WDFs of cores. Then rewrite Eq. (27), (28) and (29) into (40).

$$\begin{aligned} WDF_{c1}(i,j) &= 2\Theta(i,j) + \frac{k\pi}{2}, \\ WDF_{c2}(i,j) &= 2\Theta(i,j) + \frac{k\pi}{2} - \pi, \\ WDF_d(i,j) &= -2\Theta(i,j) + \frac{k\pi}{2}. \end{aligned} \tag{40}$$

According to the above analysis, we summarize the whole *walking algorithm* in Algorithm 3.

Algorithm 3: Walking algorithm.

```
    Input: $\Theta$ (Orientation field)
    Output: $P_c$ (cores) and $P_d$ (deltas)
1   for $k = 1$ to $4$ do
2       Get $WDF_{c1}$, $WDF_{c2}$ and $WDF_d$ using Eq. (40);
3       if no core is detected then
4           For each starting point $p \in S$, run Algorithm1 with inputs $p$ and $WDF_{c1}$
            until the result is a genuine core $P_{c1}$ (i.e. Algorithm 2);
5           Get a genuine core $P_{c2}$ using $WDF_{c2}$ in the same way;
6       if no delta is detected then
7           For each starting point $p \in S$, get a genuine delta using $WDF_d$ and add it
            to $P_d$;
8   Replace points in $P_d$ which are too close with their average point;
9   Output $P_c = \{P_{c1}, P_{c2}\}$ and $P_d$;
```

4. Experimental Results

4.1. Datasets and Parameters

The proposed walking algorithm is tested on the subset of FVC database and the benchmark database of the first fingerprint singular points detection competition (SPD2010). The former one, FVC database, is extensively used in evaluating the performance of different fingerprint verification methods. We choose the first imprint of each finger in FVC2000 DB2 [32], FVC2002 DB2 [33] and FVC2004 DB2 [34], total 330 fingerprints. Then divide them into two parts: those belongs to DB2A, total 300 fingerprints, are added to the testing set, and those in DB2B, total 30 fingerprints, are added to the training set. The singular points of these fingerprints are manually labeled beforehand to obtain ground truth.

The SPD2010 database has 500 fingerprint images with size 355×390 pixels, captured by an optical scanner, Microsoft Fingerprint Reader C model 1033, without any restrictions on the poses of fingers. The fingerprint images in SPD2010 database vary largely in quality, type, affine transformation and nonlinear distortion. The ground truths of cores and deltas are publicly available at `http://paginas.fe.up.pt/~spd2010/`, obtained by hand according to the definition of singular points in [35]. We sample 50 fingerprints starting from the first image with a step length of 10 in the whole SPD2010 dataset and add them into the training set. The rest of fingerprints in SPD2010 are put into the testing set. So there are 80 fingerprints in our training set and 750 in the testing set.

The performance of singular point detection method is evaluated according to the instructions from [36] and [5], summarized as follows:

- A singular point with coordinate (x, y) and type t ($t = 1$ for core and $t = 2$ for delta) is represented by (x, y, t). For a ground truth singular point, (x_0, y_0, t_0), if a detected singular point (x, y, t), satisfies $(t = t_0)$ and $\sqrt{(x - x_0)^2 + (y - y_0)^2} < \delta$ (termed "allowed distance error", $\delta = 10$ in SPD2010), it is said to be truly detected and, otherwise, it is called a miss.

- The **detection rate** is defined as the ratio of truly detected singular points to all ground truth singular points.
- The **miss rate** is defined as the ratio of the number of missed singular points to the number of all ground truth singular points.
- The **false alarm rate** is defined as the number of falsely detected singular points versus the number of all ground truth singular points.
- If all singular points are detected and there are no spurious singular points in a fingerprint, the fingerprint is considered to be **correctly detected**.

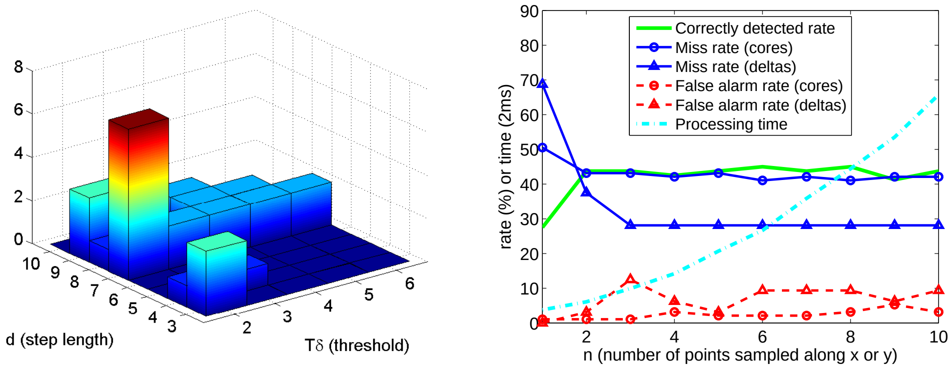
All output results have been generated on a PC with Intel(R) Core(TM) Processor (i5-3470, 3.2 GHz), 4 GB of RAM, implemented in Matlab language. The preprocessing steps like segmentation and orientation field estimation are implemented directly by using the public source codes [31] which follows the basic algorithm in [23].

We use the training set to tune the parameters of our algorithm. There are mainly 4 parameters in our algorithm: the length d of one walking step, the threshold T_δ for stopping the walking process, the radius R of the neighborhood of the candidate singular point and the number n of the chosen starting points. A grid search for theses parameters is applied. To avoid overfitting, we collect all values which make the correctly detected rate larger than 90% of the highest rate. when $R = 16$ the performance remains in a higher level, so we choose $R = 16$. Then for different n, the times of achieving higher performance at (d, T_δ) are draw in Fig. 9a and hence $(d, T_\delta) = (7, 2)$ is chosen. When $R = 16$, $d = 7$ and $T_\delta = 2$, the performance varying with n is shown in Fig. 9b. After making a tradeoff between corrected detected rate and processing time, we set $n = 2$. Fig. 9b also indicates when $n \leq 2$ the correctly detected rate remains relatively stable, which implies the efficiency is an intrinsic property of the proposed algorithm.

4.2. Comparison with Existing Methods

The walking algorithm is compared with three state-of-the-art methods: Poincaré index method [6], complex filter method [18] and angle matching index combined with convergence index filter (AMF-based, also a model-based method) method [12]. The Poincaré index method uses a square curve with length 25 pixels as the integrating path, which is proved optimal by Hong and Jain [37]. The fingerprint foreground is divided into non-overlapping blocks with block size 3×3 pixels, only the Poincaré index of center pixel in each block is calculated. The postprocessing steps are: 1) removing the cores or deltas who are too close to the background (the distance is smaller than 16 pixels); 2) if a delta is too close to a core (the distance between them is smaller than 12 pixels), remove both of them; and 3) in a very small region (a circular region with a radius of 12 pixels), if there are more than one core or delta, an average core or delta is computed instead.

Since the walking algorithm is designed to be fast, not too accurate. The other methods can be more accurate but substantially time-consuming because they can not



(a) (b)

Figure 9: Performance of the walking algorithm with different parameters on the training set. (a) Times of achieving higher performance for different n at (d, T_δ). (b) Performance varying with n when $d = 7$ and $T_\delta = 2$.

Table 1: Performance of state-of-the-art methods, walking algorithm and combined methods on the testing set. CD: correctly detected rate.

Algorithm	CD(%)	Detection rate (%)		Miss rate (%)		False alarm rate (%)		Ave. Time (ms)
		cores	deltas	cores	deltas	cores	deltas	
AMF-based [12]	**47.20**	61.56	72.01	38.44	27.99	11.38	**3.80**	8616.6
Complex filter [18]	42.40	56.53	**75.27**	43.47	**24.73**	9.10	26.09	119.5
Poincaré index [6]	32.67	**64.67**	72.28	**35.33**	27.72	58.68	103.8	101.6
Walking	43.33	54.85	69.84	45.15	30.16	4.91	6.79	**12.2**
Walking+PI	46.13	58.68	69.84	41.32	30.16	**4.67**	6.79	14.4
Walking+AMF	46.80	59.52	69.84	40.48	30.16	**4.67**	6.79	173.9

avoid investigating each pixel or block in the foreground. Hence we can take a trade off between accuracy and efficiency by combining our walking algorithm and the other accurate methods. We use the walking algorithm to detect all singular points, then in the local area (a square region with side length 32 pixels) of each singular point, another accurate method (Poincaré index or AMF-based method in our experiment) is utilized to refine the position of the singular point. If no singular point can be found by Poincaré index or AMF-based method, the corresponding singular point detected by walking algorithm is removed. As the walking algorithm has been already accurate for detecting deltas, we do not refine the position of deltas.

Table 1 shows the effectiveness and average processing time (excluding the time of estimating orientation field) of complex filter method, Poincaré index method, AMF-based method and our walking algorithms. Our walking algorithm costs 12.2 ms in average to detect all singular points in a fingerprint image, while the existing methods spend about 8 times or even 700 times more than ours. The proposed algorithm has a low false alarm rate but the correctly detected rate and miss rate are not so superior. This result validates the above statement: the walking algorithm is designed to be fast, but not too accurate. By combing the other methods, the accuracies (evaluated by correctly

Table 2: Performance of state-of-the-art methods, walking algorithm and combined methods on the subsets of FVC datasets. CD: correctly detected rate.

Algorithm	CD(%)	Detection rate (%)		Miss rate (%)		False alarm rate (%)	
		cores	deltas	cores	deltas	cores	deltas
AMF-based [12]	62.00	76.90	81.08	23.10	18.92	9.01	**2.03**
Complex filter [18]	59.33	73.52	**85.14**	26.48	**14.86**	5.92	31.08
Poincaré index [6]	38.00	**82.54**	83.11	**17.46**	16.89	71.27	143.24
Walking	59.00	69.86	77.03	30.14	22.97	1.13	6.08
Walking+PI	62.00	72.96	77.03	27.04	22.97	**0.56**	6.08
Walking+AMF	**62.67**	73.52	77.03	26.48	22.97	**0.56**	6.08

Table 3: Scores of different methods evaluated by best-take-all protocol.

Methods	Scores	Methods	Scores
AMF-based [12]	213	Walking	187
Complex filter [18]	196	Walking+PI	243
Poincaré index [6]	**299**	Walking+AMF	286

detected rate and miss rate) of walking+PI and walking+AMF method approximate that of AMF-based method which is the most accurate among these three existing methods. The walking+PI method uses about 2 ms to exchange 3% improvement in the correctly detected rate, so it has the best performance in general. Table 2 shows the effectiveness of different methods on the subsets of FVC datasets where our proposed Walking+AMF method outperforms the others in terms of correctly detected rate.

We also report the performance of different methods by using best-take-all protocol. Concretely, for each ground truth of singular point, if the singular point detected by method A is nearest to the ground truth, method A gets one score and the others get zero. The results are shown in Table 3, which indicates the Poincaré index method has the highest score followed by our combined methods.

Furthermore, the allowed distance error δ (the default value in SPD2010 is 10 pixels) can be varied with different applications. The correctly detected rates of different methods varying with δ are plotted in Fig. 10. To be concrete, Fig. 10a illustrates the absolute values of correctly detected rates of existing methods and the walking algorithm. When $\delta \geq 14$ the walking algorithm approaches the AMF-based method and when $\delta > 26$ our algorithm achieves the highest correctly detected rate. As the curves of the AMF-based method, walking algorithm and combined methods (walking+PI and walking+AMF) are very close to each other, we draw the relative correctly detected rate curves with respect to AMF-based method in Fig. 10b to sharpen the difference between them. It shows the combined methods outperform the AMF-based method in general. Therefore, we recommend our walking algorithm for real-time applications with larger accuracy tolerance, our walking+AMF method for applications which care more about accuracy and our walking+PI method for applications requiring both accuracy and efficiency.

There's no doubt that a better estimation of orientation field will improve the performance of the walking algorithm. In our work the gradient-based method [31] is used to estimate the pixel-wise orientation field. But its performance degrades quickly for images with low quality and thus affects the followed singular points detection process. To compare the impacts of orientation fields with different quality, we implement the MCO method [30] which is more robust to noise to estimate a block-wise orientation field with

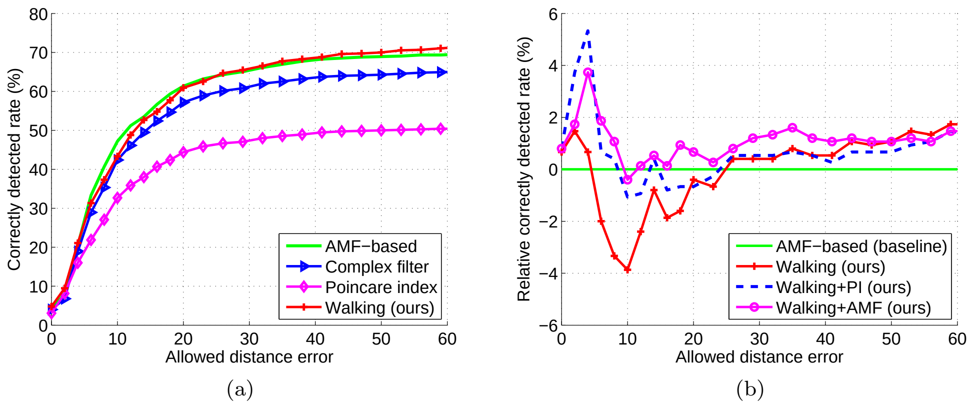


Figure 10: Performance comparison of different methods.

block size 8×8. The results illustrated in Fig. 11 validate that a better estimation of orientation field will improve the performance of the proposed walking algorithm and so it is with other singular point detection methods.

5. Conclusions

This paper proposed a novel fast algorithm which can directly walk to the singular points, succeeding to avoid scanning the fingerprint image. By analysing the orientation field generated by Zero-pole Model and that estimated from real fingerprint image, walking directions and WDFs are designed and derived from the real fingerprint orientation field. Then on WDFs we can rapidly walk to singular points following the walking directions. Besides the efficiency, our walking algorithm is quite easy to be implemented and can deal with large rotated fingerprint images. It can also be combined with state-of-the-art method to improve the accuracy: we can rapidly walk to a singular point and then fine-tune the location of the singular point using the complicated process proposed by existing scanning-based method. Experimental results on SPD2010 datasets and subset of FVC databases validated the amazing efficiency of the walking algorithm and the comparable accuracy. We believe the walking algorithm can be used in real time Automatic Fingerprint Identification System (AFIS), including large scale AFIS and embedded AFIS.

Future works include: 1) designing a heuristic function for starting points selection according to the relation between cores and deltas, and 2) designing more effective combination way to improve the accuracy of the walking algorithm.

Acknowledgement

This work was sponsored by the National Natural Science Foundation of China (Project no. 61170287, 61232016) and National Science Foundation of Hunan Province (Grant no. 2jj3069).

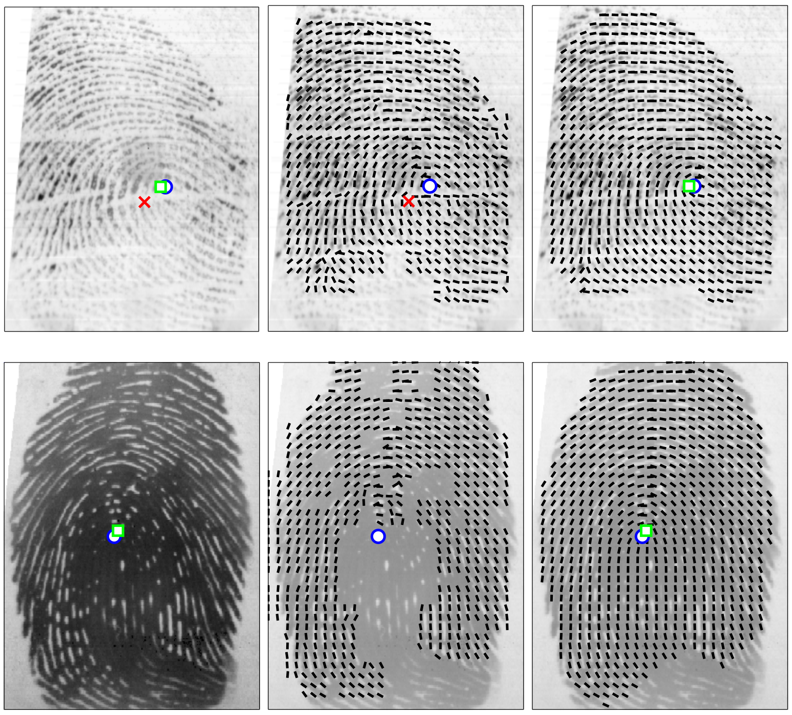


Figure 11: Performance of the walking algorithm using the orientation fields estimated by gradient-based method [31] (second column) and MCO method [30] (third column). The ground truth and detected cores are marked with “◯” and “□” (“×”)

References

[1] L. Nanni, A. Lumini, A deformation-invariant image-based fingerprint verification system, NeuroComputing 69 (16) (2006) 2336–2339.

[2] L. Nanni, S. Brahnam, A. Lumini, Biohashing applied to orientation-based minutia descriptor for secure fingerprint authentication system, Electronics letters 47 (15) (2011) 851–853.

[3] M. Kawagoe, A. Tojo, Fingerprint pattern classification, Pattern Recognition 17 (3) (1984) 295–303.

[4] A. M. Bazen, S. H. Gerez, Systematic methods for the computation of the directional fields and singular points of fingerprints, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 905–919.

[5] J. Zhou, F. Chen, J. Gu, A novel algorithm for detecting singular points from fingerprint images, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (7) (2009) 1239–1250.

[6] D. Maltoni, D. Maio, A. K. Jain, S. Prabhakar, Handbook of fingerprint recognition, Springer Science & Business Media, 2009.

[7] D. Weng, Y. Yin, D. Yang, Singular points detection based on multi-resolution in fingerprint images, Neurocomputing 74 (17) (2011) 3376–3388.

[8] F. Belhadj, S. Akrouf, S. Harous, S. A. Aoudia, Efficient fingerprint singular points detection algorithm using orientation-deviation features, Journal of Electronic Imaging 24 (3) (2015) 033016–033016.

[9] B. G. Sherlock, D. M. Monro, A model for interpreting fingerprint topology, Pattern recognition 26 (7) (1993) 1047–1055.

[10] Y. Wang, J. Hu, D. Phillips, A fingerprint orientation model based on 2d fourier expansion (fomfe) and its application to singular-point detection and fingerprint indexing, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (4) (2007) 573–585.

[11] L. Fan, S. Wang, H. Wang, T. Guo, Singular points detection based on zero-pole model in fingerprint images, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (6) (2008) 929–940.

[12] J. Qi, S. Liu, A robust approach for singular point extraction based on complex polynomial model, in: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, 2014, pp. 78–83.

[13] K. Nilsson, J. Bigun, Complex filters applied to fingerprint images detecting prominent symmetry points used for alignment, in: Biometric authentication, Springer, 2002, pp. 39–47.

[14] K. Nilsson, J. Bigun, Prominent symmetry points as landmarks in fingerprint images for alignment, in: Proceedings of the 16th International Conference on Pattern Recognition (ICPR), Vol. 3, IEEE, 2002, pp. 395–398.

[15] K. Nilsson, J. Bigun, Localization of corresponding points in fingerprints by complex filtering, Pattern Recognition Letters 24 (13) (2003) 2135–2144.

[16] S. Chikkerur, N. Ratha, Impact of singular point detection on fingerprint matching performance, in: Fourth IEEE Workshop on Automatic Identification Advanced Technologies, IEEE, 2005, pp. 207–212.

[17] H. Fronthaler, K. Kollreider, J. Bigun, Local feature extraction in fingerprints by complex filtering, in: Advances in Biometric Person Authentication, Springer, 2005, pp. 77–84.

[18] A. I. Awad, K. Baba, Singular point detection for efficient fingerprint classification, International Journal of New Computer Architectures and their Applications (IJNCAA) 2 (1) (2012) 1–7.

[19] A. K. Jain, S. Prabhakar, L. Hong, S. Pankanti, Filterbank-based fingerprint matching, IEEE Transactions on Image Processing 9 (5) (2000) 846–859.

[20] C.-H. Park, J.-J. Lee, M. J. Smith, K.-H. Park, Singular point detection by shape analysis of directional fields in fingerprints, Pattern Recognition 39 (5) (2006) 839–855.

[21] C. Jin, H. Kim, Pixel-level singular point detection from multi-scale gaussian filtered orientation field, Pattern Recognition 43 (11) (2010) 3879–3890.

[22] H. Chen, L. Pang, J. Liang, E. Liu, J. Tian, Fingerprint singular point detection based on multiple-scale orientation entropy, IEEE Signal Processing Letters 18 (11) (2011) 679–682.

[23] L. Hong, Y. Wan, A. K. Jain, Fingerprint image enhancement: algorithm and performance evaluation, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 777–789.

[24] Y. Mei, G. Cao, H. Sun, R. Hou, A systematic gradient-based method for the computation of fingerprints orientation field, Computers & Electrical Engineering 38 (5) (2012) 1035–1046.

[25] E. Zhu, J. Yin, C. Hu, G. Zhang, A systematic method for fingerprint ridge orientation estimation and image segmentation, Pattern Recognition 39 (8) (2006) 1452–1472.

[26] S. Ram, H. Bischof, J. Birchbauer, Modelling fingerprint ridge orientation using legendre polynomials, Pattern Recognition 43 (1) (2010) 342–357.

[27] S. Jirachaweng, Z. Hou, W.-Y. Yau, V. Areekul, Residual orientation modeling for fingerprint enhancement and singular point detection, Pattern Recognition 44 (2) (2011) 431–442.

[28] W. Bian, Y. Luo, D. Xu, Q. Yu, Fingerprint ridge orientation field reconstruction using the best quadratic approximation by orthogonal polynomials in two discrete variables, Pattern Recognition 47 (10) (2014) 3304–3313.

[29] M. Oliveira, N. J. Leite, A multiscale directional operator and morphological tools for reconnecting broken ridges in fingerprint images, Pattern Recognition 41 (1) (2008) 367–377.

[30] E. Zhu, E. Hancock, J. Yin, J. Zhang, H. An, Fusion of multiple candidate orientations in fingerprints, in: Image Analysis and Recognition, Springer, 2011, pp. 89–100.

[31] P. D. Kovesi, MATLAB and Octave functions for computer vision and image processing, available from: <http://www.peterkovesi.com/matlabfns/>.

[32] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, Fvc2000: Fingerprint verification competition, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (3) (2002) 402–412.

[33] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, Fvc2002: Second fingerprint verification competition, in: Proceedings of 16th international conference on Pattern recognition, Vol. 3, IEEE, 2002, pp. 811–814.

[34] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, Fvc2004: Third fingerprint verification competition, in: Biometric Authentication, Springer, 2004, pp. 1–7.

[35] E. Henry, Classification and Use of Finger Prints, London: Routledge, 1900.

[36] F. Magalhães, H. P. Oliveira, A. Campilho, SPD2010 - Fingerprint Singular Points Detection Competition Database [Available: `http://paginas.fe.up.pt/~spd2010/`].

[37] L. Hong, A. Jain, Classification of fingerprint images, in: Proceedings of the Scandinavian Conference on Image Analysis, Vol. 2, 1999, pp. 665–672.