BlockRFC: Real-Time Rolled Fingerprint Construction and Distortion Rectification

YONGLIANG ZHANG, YIFAN WU, MINGHUA GAO, SHENGYI PAN, ZIRUI SHAO, AND TIAN LUO
College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China
Corresponding author: Yongliang Zhang (titanzhang@zjut.edu.cn)
This work was supported in part by the Public Welfare Technology Research Program of Zhejiang Province under Grant LGF18F030008, and in part by the Hangzhou Jinglianwen Technology Company Ltd.

ABSTRACT Compared with a flat fingerprint, the rolled fingerprint has a larger fingerprint area and can be extracted more minutiae. It has high requirements in many fields, not only in the military environment or the police field but also in many civil application fields. The challenge that has been troubled for a long time is that contact-based rolled fingerprint registration is easy to cause obvious distortion without human experts’ supervision, which has a negative impact on fingerprint recognition performance. Due to the elastic deformation of fingertips, the mosaicking gaps in the rolled fingerprint are usually visible but challenging to locate. To address these problems, we propose a novel rolled fingerprint construction algorithm called BlockRFC (Block-based Rolled Fingerprint Construction) in this article. BlockRFC’s core idea is to use the fingerprint image block as a processing unit for mosaicking and distortion rectification. BlockRFC is based on a real-time mosaicking framework, which makes it possible to construct a rolled fingerprint while collecting fingerprint images. One distinctive advantage of BlockRFC is that it does not require minutiae or ridge information, but fully utilizes the gray-scale information and foreground area in the fingerprint image block. In this article, we first propose a metric called Mosaicking Gap Rate (MGR), which can effectively quantify the mosaicking gaps in the rolled fingerprints. Experimental results show that the proposed method can not only effectively locate and eliminate the mosaicking gaps but also have better recognition accuracy than previous methods.

INDEX TERMS Fingerprint recognition, rolled fingerprint construction, distortion rectification, integral image, multi-objective optimization, dynamic programming.

I. INTRODUCTION
With the popularization of digital authentication technology, biometrics has become the most reliable authentication method, and fingerprint authentication is the most widely used method [1]. The fingerprint scanners on the market usually use the plane imprinting method to collect the small-size fingerprint. However, it is challenging for the scanner to acquire the same fingerprint region every time, which makes the false rejection rate higher. This shortcoming can be overcome by rolling the finger to capture a series of fingerprint images and mosaic them into a complete rolled fingerprint. Compared with the flat fingerprint, the foreground area of the rolled fingerprint is larger, which provides more fingerprint feature information (i.e., minutiae and ridges) and makes the identity authentication more accurate [2]. Law enforcement agencies usually use rolled fingerprints to help identify the fingerprints found at the crime scene. The Federal Bureau of Investigation (FBI) has experimented with evaluating the accuracy of matching latent fingerprints against flat and rolled fingerprints in the FBI’s Integrated Automated Fingerprint Identification System (IAFIS) is 38.8% and 54.4%, respectively [3]. Therefore, it is valuable work to develop a convenient and high-precision rolled fingerprint construction method.

A few years ago, fingerprint scanners equipped with large sensors and an automatic roll synthesis function came out and replaced ink-based fingerprint acquisition methods [4]. However, whether it is an ink-based method or a fingerprint scanner, in order to construct a high-quality rolled fingerprint, the whole process of fingerprint acquisition can not be...
separated from the supervision of human experts. To mitigate the need for manual supervision, researchers have proposed many rolled fingerprint construction methods for improving matching accuracy [5]. Undeniably, accuracy is an essential issue but not the only one. In fact, the primary purpose of manual supervision is to prevent obvious skin deformation due to the user’s improper finger rolling. That is to say, once the algorithm can detect and rectify the distortion during the finger rolling process, then there is no need for manual supervision at all.

Another problem that has been troubled for a long time is whether the construction of rolled fingerprints must use minutiae or ridges. Skin deformation can lead to the change of ridge pattern and produce some fake minutiae, which is more serious when dealing with rolled fingerprints. However, our previous works [6], [7] have shown that good results can be achieved by relying only on image gray-scale information and fingerprint foreground area.

In this article, a rolled fingerprint construction algorithm called BlockRFC is proposed. BlockRFC can not only support real-time mosaicking of rolled fingerprints but also detect distortion and provide two methods to rectify it. Experimental results show that the algorithm is better than the existing algorithms in improving fingerprint recognition accuracy. Our main contributions are as follows:

- We propose a novel real-time mosaicking algorithm BlockRFC for rolled fingerprints. When the user rolls his finger on a live-scanner, BlockRFC can evaluate the distortion in the currently rolled fingerprint and then rectify it, which mitigates human experts’ supervision.
- To the best of our knowledge, MGR proposed by us is the first metric to quantify the rolled fingerprint’s mosaicking gaps. It plays an indispensable role in the location and optimization of mosaicking gaps.
- Different from conventional methods, BlockRFC uses the fingerprint image block as a processing unit. It does not use traditional fingerprint biological information (i.e., minutiae and ridges), but makes full use of gray-scale information and fingerprint foreground area of blocks. Besides, BlockRFC combines coordinate with frame index to uniquely describe the image block, which makes the rolled fingerprint construction problem be modeled as an optimization problem.
- We evaluate BlockRFC from three aspects: visualization, fingerprint matching performance, and effects of optimization. Experiments show that it can not only locate and eliminate mosaicking gaps efficiently but also achieve higher matching accuracy than previous methods.

The rest of the paper is organized as follows: Section II reviews the related work on rolled fingerprint construction and distortion rectification. The general process of the proposed method is explained in Section III. Section IV describes two different optimization methods, which are the focus of this article. In Section V, various comparative experiments are introduced to verify the effectiveness of the proposed method. Finally, the conclusion and future work are presented in Section VI.

II. RELATED WORK

Rolled fingerprint construction is a special issue of fingerprint mosaicking. Due to skin deformation, low image quality, and a large number of images, there is little research on it. Fingerprint mosaicking is broadly categorized into feature-level mosaicking and image-level mosaicking. For feature-level rolled fingerprint mosaicking, a nonrigid image registration method based upon a Markov Random Field (MRF) energy model [5] was proposed. It has achieved excellent results in matching accuracy, but the time complexity is high, and it is unable to construct rolled fingerprints in an online manner. In this article, we mainly study the image-level rolled fingerprint mosaicking method.

Image-level mosaic mainly refers to combining the Region of Interest (ROI) of several fingerprint images into a composite image. Some image-level mosaicking methods were proposed to construct rolled fingerprints in the literatures [6]–[9]. To sum up, two main factors affect the quality of rolled fingerprints: the integrity of the rolled fingerprint and the mosaicking gaps. The integrity of rolled fingerprints has been studied in most methods, but there is little mention of eliminating mosaicking gaps. In [8], a cover-based mosaicking algorithm was proposed to synthesize the image by calculating the confidence of each pixel. This method can get a complete fingerprint contour, but the ridges of the rolled fingerprint are staggered and disordered. In [9], the local affine transformations between two neighboring image sequences were estimated. Its purpose is to avoid mosaicking gaps, but some mosaicking gaps still exist. The emphasis of this article is to reduce the degree of mosaicking gaps based on avoiding them. In [6], a real-time rolled fingerprint mosaicking algorithm based on key columns was proposed, which took the dislocation of the fingertip region as the principal object to be optimized for the first time. However, this method has limitations in selecting mosaicking units, resulting in dense and obvious mosaicking lines in the image. In [7], a rolled fingerprint construction algorithm based on a block scale was proposed. The algorithm is more flexible in mosaicking and can meet the real-time mosaicking requirements, but there is still room to optimize the mosaicking gaps. Based on a real-time mosaicking framework, the rolled fingerprint construction method proposed in this article can obtain a complete fingerprint area and less mosaicking gaps.

III. PROPOSED METHODOLOGY

In this article, it is assumed that the finger does not slide during the rolling process. The finger needs to roll horizontally from one end to another in the same direction (from left to right or from right to left) on the interface of the optical acquisition instrument. In particular, when the finger is roll-back, the current mosaicking is terminated, and the new mosaicking starts. At the beginning of the acquisition process, an image of the acquisition interface without finger
TABLE 1. Nomenclature.

| Notation | Description |
|----------|-------------|
| $I_i$ | the $i^{th}$ frame image |
| $I_i'$ | the $i^{th}$ frame after image segmentation |
| $I_i(x, y)$ | the gray-scale value of a pixel whose coordinate is $(x, y)$ in the $i^{th}$ frame image |
| $I_i'(a, b)$ | the gray-scale sum of the $b^{th}$ image block in the $a^{th}$ row from $I_i'$ |
| $\Psi_{k-1}$ | the mosaicking result before the $k^{th}$ frame |
| $FG_i$ | the area of fingerprint foreground in the $i^{th}$ frame |
| $FG_i'$ | the area of fingerprint foreground after image segmentation in the $i^{th}$ frame |
| $H$ or $W$ | the height or width of the fingerprint image |
| $h$ or $w$ | the height or width of the image block |
| $H'$ or $W'$ | the number of blocks cut in the vertical or horizontal direction |
| $L(a, b)$ | the set of pixel coordinates and all of its pixels are from the $b^{th}$ image block in the $a^{th}$ row of $I_i'$ |
| $SAT(x, y)$ | the gray-scale value calculated by the integral image for the point $(x, y)$ |
| $SAT^*(X_i, Y_i)$ | the gray-scale value calculated by the integral image for the $Y_i^{th}$ image block in the $X_i^{th}$ row |
| $CR(k)$ | the index of the fingerprint center in the $k^{th}$ frame |
| $J_i(a, b)$ | the frame index of the $b^{th}$ image block in $a^{th}$ row of $I_i'$ |
| $J_{k-1}(a, b)$ | the frame index of the $b^{th}$ image block in $a^{th}$ row of $\Psi_{k-1}$ |
| $R(k)$ | the finger rolling direction of the current frame $I_k$ |
| $M_i(a, b)$ | the Boole after binarization in the $b^{th}$ image block in the $a^{th}$ row and it is from $I_i'$ |
| $F(a, b)_i(i, j)$ | the deviation degree of the $b^{th}$ image block in the $a^{th}$ row in $I_i'$ and $I_j'$ |
| $Q$ | the set of direction constants |

FIGURE 1. Image sequence of rolled fingerprints.

is captured and recorded as a background frame $I_0$. Next, when the fingerprint in the image is detected for the first time, the image is marked as $I_1$. Image acquisition does not stop when fingerprints are detected in subsequent images, and the frame index increases sequentially. Once the image is identified as a non-fingerprint, it will end immediately. Figure 1 shows an example of a rolled fingerprint sequence $I = \{I_1, I_2, \ldots, I_{12}\}$, where $I_i$ is the $i^{th}$ frame image. Table 1 summarizes the symbols frequently used throughout this article.

The construction of a rolled fingerprint is performed in the real-time mosaicking manner, which can dynamically display the fingerprint mosaicking process. Compared with the offline model, it can cooperate with scanners to automatically correct errors during fingerprint collection. Assume that the frame acquired at the current time is $I_k$, and the mosaicking result before the $k^{th}$ frame is recorded as $\Psi_{k-1}$. Real-time mosaicking is equivalent to mosaicking $I_k$ and $\Psi_{k-1}$ to generate $\Psi_{k}$ when each frame is acquired. Finally, the final mosaicking image $\Psi$ can be constructed.

The rolled fingerprint construction algorithm is divided into the following steps:

1) Based on fingerprint segmentation of foreground and background to accurately estimate the appearance of
the fingerprint image, some low-quality fingerprint images can be removed to improve the mosaicking speed and quality.

2) The fingerprint region is divided into image blocks, and the integral image is used to calculate the gray-scale information in the image blocks.

3) The central region of a fingerprint is located by binary search, and then the partition mosaicking strategy is adopted to eliminate the blank fragments in the rolled fingerprint.

4) A non-decreasing sequence based on dynamic programming is proposed to eliminate the mosaicking gaps caused by the difference of frame indexes between adjacent image blocks.

5) Finally, we introduce the MGR to describe the degree of gap between adjacent image blocks, and then the mosaicking gaps are optimized in the horizontal and vertical directions, respectively.

As shown in Figure 2, a real-time rolled fingerprint construction algorithm is proposed to construct a composed fingerprint impression. The optimization of mosaicking gaps is a crucial step to improve the accuracy of fingerprint recognition.

A. FINGERPRINT DETECTION

Define \( I_i(x, y) \) as the gray-scale value of a pixel whose coordinate is \((x, y)\) in the \(i\)th frame image. Then, the area of fingerprint foreground \( FG_i \) is calculated according to the following formula:

\[
FG_i = \{(x, y)|I_i(x, y) - I_0(x, y) > T_1\}
\]

where \(|\cdot|\) represents the number of elements in the set, and \(T_1\) is the threshold value. \(H\) and \(W\) are the height and width of the image, respectively. If \(FG_i\) of the \(i\)th frame is larger than a certain threshold \(T_2\), the image is considered as a fingerprint image.

B. IMAGE BLOCK SEGMENTATION

The image-level fingerprint mosaicking task can be regarded as extracting the ROI in a series of fingerprint images, and then recombining these regions into a complete rolled fingerprint image. However, extracting the ROI from overlapping regions of multiple fingerprint images is a challenging problem to be solved urgently in constructing rolled fingerprints. In [10], a block segmentation method was proposed to extract the overlapped regions of two latent fingerprints. Moreover, it proposed a novel idea to judge the source of image blocks based on their feature information. Similarly, a rolled fingerprint construction method based on block scale was proposed in the literature [7].

The fingerprint image with height \(H\) and width \(W\) is divided into image blocks in the form of rectangular blocks. The height and width of the segmented block are \(h\) and \(w\), respectively. If \(W'\) is the number of blocks cut in the horizontal direction and \(H'\) is the number of blocks cut in the vertical direction, then the number of total segmented blocks is \(W' \times H'\). The image block segmentation expression is as follows:

\[
\begin{align*}
W' &= \frac{W}{w} \\
H' &= \frac{H}{h}
\end{align*}
\]  

(2)

For image segmentation of raw image sequence, \(I'_i\) represents the \(i\)th frame after image segmentation. \(I'_i(a, b)\) represents the gray-scale sum of the \(b\)th image block in the \(a\)th row from \(I'_i\). \(FG'_i(a, b)\) is defined as the foreground area of this block, and the formulae are as follows:

\[
I'_i(a, b) = \sum_{(x,y) \in L(a,b)} I_i(x, y)
\]

(3)

\[
FG'_i(a, b) = \{(x, y)|I'_i(x, y) - I_0(x, y) > T_1\}
\]

(4)

\[
L(a, b) = \{(x, y)|(a-1) \times h + 1 \leq x \leq a \times h, \quad (b-1) \times w + 1 \leq y \leq b \times w\}
\]

(5)

where \(L(a, b)\) represents the set of pixel coordinates, and all of its pixels are from the \(b\)th image block in the \(a\)th row of \(I'_i\).
In the following sections, a column or row of $I'_r$ refers to a set of image blocks rather than a set of pixels.

**C. FINGERPRINT CENTER POSITIONING**

The central region of the fingerprint usually has the best image quality and the least distortion. In the process of finger rolling, the location of the central region continually changes, so that the direction of finger rolling can be reflected by the movement of the central region. In [6], the column with the smallest gray-scale value in the fingerprint image was regarded as the central region of the fingerprint. Under normal circumstances, it is impossible to remove stains and residual fingerprints entirely on the interface of the scanners, so the column with the smallest gray-scale value may not be fingerprints but stains. To solve this problem, a block-based method [7] divided $I'_r$ into left and right sub-regions, and then the sub-region with a lower gray-scale value was selected. When the sub-region was narrowed to a column after the binary search, the central region of the fingerprint was located. This article adopts this method to locate the central region of the fingerprint, and the integral image [11] is used to reduce the computational complexity.

In the integral image, the gray-scale value of each point is the sum of the gray-scale values of its upper left region, which is expressed as follows:

$$SAT(x, y) = \sum_{x_i \leq x, y_i \leq y} I(x_i, y_i)$$  \hspace{1cm} (6)

where $SAT(x, y)$ refers to the gray-scale value calculated by the integral image for the point $(x, y)$.

The image block is regarded as a unit, so the improved integral image formula is as follows:

$$SAT'(X_i, Y_i) = \sum_{a \leq x_i, b \leq y_i} I'_r(a, b)$$  \hspace{1cm} (7)

where $SAT'(X_i, Y_i)$ refers to the gray-scale value calculated by the integral image for the $I'_r$ image block in the $X'_i$ row, which is equal to the gray-scale value of all image blocks at the upper left of $(X_i, Y_i)$.

The image is divided into blocks, and then the gray-scale value of each block is recalculated by the integral image. After these steps, the location of the fingerprint center can be performed as follows. In the initial state, the whole image is defaulted to the exploring region, and four parameters are defined as follows: $l$ is the index of the leftmost column in the exploring region, with an initial value of 1; $r$ is the index of the rightmost column in the exploring region, with an initial value of $W'$; $m$ is the index of the middle column of the exploring region, which is equal to $[(l + r)/2]$; $d$ is the width of the exploring region, which is equal to $r - l + 1$. Define the gray-scale value of the left half of the exploring region as $G_l$, and another part as $G_r$. The above two parameters are calculated as follows:

$$G_l = SAT'(H', m) - SAT'(H', l)$$  \hspace{1cm} (8)

$$G_r = SAT'(H', r) - SAT'(H', m + 1 - (d \mod 2))$$  \hspace{1cm} (9)

If $G_l \geq G_r$, then the right half region is selected as the next exploring region. Otherwise, the left half region is selected. After the exploring region has been determined, six parameters (including $l, r, m, d, G_l, G_r$) need to be recalculated until $d$ is equal to 1, then $m$ is the index of the fingerprint center. The fingerprint center is actually a column which consists of image blocks, and each block is defined as a reference block. If the fingerprint image is the $k^{th}$ frame, the index of the fingerprint center is $CR(k)$. For example, the location of a reference block in the $i^{th}$ row is $(i, CR(k))$.

The integral image can avoid recalculating the gray-scale value of the same area, and the time complexity of this step is $\mathcal{O}(H \times W)$, where $H$ and $W$ are the height and width of the fingerprint image. In more detail, calculating the gray-scale value of all blocks requires $\mathcal{O}(H \times W)$, and constructing an integral image requires $\mathcal{O}(2 \times H' \times W')$, and then calculating any region only requires $\mathcal{O}(1)$.

**D. REAL-TIME ROLLED FINGERPRINT CONSTRUCTION**

Each image block in the image sequence can be uniquely determined by coordinate and frame index, so the original problem of rolled fingerprint construction can be regarded as updating the frame index of image block in the same position. As shown in Figure 3, each image block marked in red is represented by a three-dimensional vector. The first two dimensions represent position information, and the third dimension represents the frame index. Each image can be transformed into a frame index matrix with a size of $W' \times H'$. Each element in the matrix corresponds to the frame index of the block in the current position. Set $J_l(a, b)$ to represent the frame index of the $b^{th}$ image block in $a^{th}$ row of $I'_r$. The principle of updating the frame index is to select the block with a smaller gray-scale value and larger foreground area. The condition expression is as follows:

$$J'_k(a, b) < J'_{k-1}(a, b) \land FG'_k(a, b) > FG'_{k-1}(a, b)$$  \hspace{1cm} (10)

where $J'_{k-1}(a, b)$ represents the frame index of the $b^{th}$ image block in $a^{th}$ row of $\Psi_{k-1}$, and $FG'_{k-1}(a, b)$ represents the foreground area of this block. $\Psi'_{k-1}$ represents $\Psi_{k-1}$ after image segmentation. If the above conditions are met, $J'_k(a, b)$ should be updated to $k$.

**FIGURE 3.** Description of image blocks.
If the finger rolling direction is from left to right during image acquisition, it is defined as: \((1, 2, \ldots, CR(k) - 1)\) is the spliced region, \((CR(k) + 1, CR(k) + 2, \ldots, W')\) is the non-spliced region. The rolled fingerprint construction algorithm employs different mosaicking strategies for these two regions, which can effectively avoid the blank fragments in the fingerprint. Figure 4(a) shows that some blank fragments appear inside the rolled fingerprint, but they are avoided in (b) because of the partition mosaicking strategy. For the spliced region, the strategy of avoiding coverage as much as possible is adopted, while for the non-spliced region, the strategy of full coverage is adopted. The mosaicking strategy of each row is consistent, so the algorithm only needs to execute from the first row to the last row, and the mosaicking starts from the reference block of each row.

The execution sequence of the mosaicking strategy is as follows: for the spliced region, start from the reference block to the leftmost block; for the non-spliced region, start from the reference block to the rightmost block. If a block of the current frame \(I_k\) and another block at the same position in \(\Psi_{k-1}\) meet the condition of updating the frame index, the frame index of this block in \(\Psi_k\) needs to be updated to \(k\). The difference between the mosaicking strategy of the two regions is that the mosaicking in the spliced region will stop immediately in case of failing to meet the update conditions.

Algorithm 1 shows the process of constructing a rolled fingerprint when the finger rolling direction is from left to right.

The time complexity of this step is \(O(H' \times W' \times N)\), which \(N\) refers to the frame index of the last image acquired. In order to speed up the mosaicking, when the center of the newly acquired fingerprint image does not change, it can be skipped directly.

IV. OPTIMIZATION METHOD

Skin deformation makes it difficult to match the ridges of the same part from a finger in two adjacent frames. At present, some mosaicking methods utilize image transformation to eliminate deformation [12]. Although these methods have achieved some results, they have also produced some pseudo minutiae. In this article, two methods based on dynamic programming and multi-objective optimization are proposed to optimize the mosaicking gaps. The first method is to combine the monotonic continuity of fingerprints in image sequence with dynamic programming to optimize some serious mosaicking gaps. The second method is to describe the degree of mosaicking gap between adjacent blocks according to the position relationship and the frame index association. Then, the multi-objective optimization method is carried out to minimize the degree of gaps between the four adjacent directions of the block. The experimental results show that the second method can find the mosaicking gaps which are difficult to be observed by human eyes and optimize them.

A. OPTIMIZATION BASED ON DYNAMIC PROGRAMMING

The movement of the central region of the fingerprint can be used to judge whether the finger roll-back occurs. In the same way, the movement of the reference block can determine the finger rolling direction. If the current frame is \(I_k\), the direction

\[
\begin{align*}
J_{\Psi_k}(a, b) &= J_{\Psi_{k-1}}(a, b); \\
&\text{if } I_k(a, b) < \Psi_{k-1}(a, b) \text{ then} \\
&\quad \text{if } FG_k(a, b) > FG_{k-1}'(a, b) \text{ then} \\
&\quad\quad J_{\Psi_k}(a, b) = k; \\
&\text{if } J_{\Psi_k}(a, b) == J_{\Psi_{k-1}}(a, b) \text{ then} \\
&\quad \text{break;} \\
&\text{for } b = CR(k) \rightarrow W' \text{ do} \\
&\quad J_{\Psi_k}(a, b) = J_{\Psi_{k-1}}(a, b); \\
&\quad \text{if } I_k(a, b) < \Psi_{k-1}(a, b) \text{ then} \\
&\quad\quad \text{if } FG_k(a, b) > FG_{k-1}'(a, b) \text{ then} \\
&\quad\quad\quad J_{\Psi_k}(a, b) = k;
\end{align*}
\]

\(n\) refers to the frame index of the last image acquired. In order to speed up the mosaicking, when the center of the newly acquired fingerprint image does not change, it can be skipped directly.
of finger rolling can be defined as follows:

\[
R(k) = \begin{cases} 
0, & CR(k - 1) > CR(k); \\
1, & CR(k - 1) < CR(k); \\
R(k - 1), & CR(k - 1) = CR(k).
\end{cases}
\] (11)

where \(R(k)\) indicates the finger rolling direction of the current frame \(I_k\), and \(R(k) = 0\) indicates that the finger rolls from right to left, \(R(k) = 1\) indicates that the finger rolls from left to right.

If the finger rolls from left to right, the frame index of all image blocks in each row must be monotonically increasing. This finding means that the frame index of blocks in each row is a non-decreasing sequence. When the mosaicking process meets the regular rolling (i.e., no roll-back), the following condition is met:

\[
R(k - 1) = R(k), \quad k \in [2, N]
\] (12)

In each row, in order to keep the monotonic increase in the frame index of the image blocks, the frame index of the current block is larger than that of the left adjacent block, and the frame index of the current block is smaller than that of the right adjacent block. Maintaining this relationship between the frame indexes of adjacent image blocks is not an independent problem, and each of its stages has an impact on the next stage. Therefore, the state transition equation can be expressed through dynamic programming in Algorithm 2.

**Algorithm 2 Optimization Based on Dynamic Programming**

```plaintext
while R(k - 1) == R(k) do
  if R(k - 1) == 1 then
    for a = 1 → H' do
      for b = 1 → W' do
        \[ J_{Ψk}(a, b + 1) = \max(J_{Ψk}(a, b), J_{Ψk}(a, b + 1)); \]
  if R(k - 1) == 0 then
    for a = 1 → H' do
      for b = 1 → W' do
        \[ J_{Ψk}(a, b + 1) = \min(J_{Ψk}(a, b), J_{Ψk}(a, b + 1)); \]
end while
```

The time complexity of this step is \(O(H' \times W' \times N)\). Whenever the \(k^{th}\) frame real-time mosaicking is completed, Algorithm 2 is executed to reduce some serious mosaicking gaps.

### B. MULTI-OBJECTIVE OPTIMIZATION

Define \(F_{a,b}(i,j)\) as the deviation degree of two image blocks. Specifically, two image blocks have the same position (the \(b^{th}\) image block in the \(d^{th}\) row), but they are from \(I'_j\) and \(I'_j\) respectively. The lower the deviation degree, the more similar the two image blocks are. The calculation expression is as follows:

\[
F_{a,b}(i,j) = SUM(M_i(a, b) \oplus M_j(a, b))
\] (13)

where \(SUM(X)\) means the sum of all elements in the matrix \(X\). \(M_i(a, b)\) and \(M_j(a, b)\) are both Boolean with the size of \(b \times w\) after binarization in the \(b^{th}\) image block in the \(d^{th}\) row. Moreover, they are from \(I'_i\) and \(I'_j\) respectively. The symbol \(\oplus\) indicates the exclusive or operator.

To describe the positional relationship between adjacent image blocks, the set of direction constants \(Q = \{a, β, γ, δ\}\) is defined to represent four adjacent directions of the image block: up, down, left and right. \(Ψ'\) represents the final mosaicking image \(Ψ\) after image segmentation. The definition of Mosaicking Gap Rate \(MGR_{(a,b)}(i,j, δ)\) indicates the degree of mosaicking gap between the \(b^{th}\) image block in the \(d^{th}\) row and the right adjacent block in \(Ψ'\), and the two image blocks are from frame \(i^{th}\) and \(j^{th}\) respectively. In the same way, the MGR in the other three directions can be obtained. Based on the permutation and combination of image blocks, the mosaicking gaps must be caused by the inconsistent frame index between adjacent image blocks. If the frame indexes of adjacent blocks are the same, it means that two image blocks come from the same image and are adjacent to each other, so there is no mosaicking gap. The above findings can be summarized as the following relational expression:

\[
\begin{align*}
MGR_{(a,b)}(i,j, α) &= 0, \quad i = j \\
MGR_{(a,b)}(i,j, α) &= 0, \quad i \neq j
\end{align*}
\] (14)

where \(φ \in Q\).

It is assumed that the image block \(A\) from the \(ψ^{th}\) frame in \(Ψ'\) is right adjacent to the image block \(B\) from the \(ψ^{th}\) frame, while the image block \(A\) is right adjacent to the image block \(C\) at the \(ψ^{th}\) frame. It can be found that there is no mosaicking gap between \(A\) and \(C\). To a certain extent, the degree of deviation between \(B\) and \(C\) can express the degree of mosaicking gap between \(A\) and \(B\). Similarly, if the image block \(D\) is right adjacent to \(B\) in the \(ψ^{th}\) frame, the degree of deviation between \(A\) and \(D\) can also express the degree of mosaicking gap between \(A\) and \(B\). The position relationship between the above image blocks is shown in Figure 5. Based on these ideas, the calculation formula for MGR is as follows:

\[
\begin{align*}
MGR_{(a,b)}(i,j, α) &= (F_{a,b}(i,j) + F_{a+1,b}(i,j))/2 \\
MGR_{(a,b)}(i,j, β) &= (F_{a,b}(i,j) + F_{a-1,b}(i,j))/2 \\
MGR_{(a,b)}(i,j, γ) &= (F_{a,b}(i,j) + F_{a,b-1}(i,j))/2 \\
MGR_{(a,b)}(i,j, δ) &= (F_{a,b}(i,j) + F_{a,b+1}(i,j))/2
\end{align*}
\] (15)

With the definition of these concepts, a comprehensive optimization model based on multi-objective can be established to eliminate the mosaicking gaps. The image block \(A\) in \(Ψ'\), which is located at \((a, b)\) and adjacent to \(A_α, A_β, A_γ,\) and \(A_δ\). Their frame indexes are \(J(A), J(A_α), J(A_β), J(A_γ),\) and \(J(A_δ)\). The optimization of mosaicking gaps is divided into two directions, horizontal and vertical. The image block \(A\) has two adjacent image blocks in each direction.
If replacing $A$ with other image blocks can reduce its MGR with adjacent blocks, then the performance of mosaicking can be improved. Due to the direction of finger rolling is horizontal, eliminating the horizontal mosaicking gaps can get better results. Therefore, reducing the horizontal MGR is the first objective function, and reducing the vertical MGR is the second objective function. According to a non-decreasing sequence, the value range of the updated block frame index $J'(A)$ belongs to $[J(A_1), J(A_3)]$.

To sum up, a multi-objective optimization model is established as follows:

$$\begin{align*}
\min Z_1 &= MGR_{(a,b)}(J'(A), J(A_3), \delta) \\
&\quad + MGR_{(a,b)}(J'(A), J(A_1), \gamma) \\
\min Z_2 &= MGR_{(a,b)}(J'(A), J(A_3), \alpha) \\
&\quad + MGR_{(a,b)}(J'(A), J(A_1), \beta) \\
J(A_1) &\leq J'(A) \leq J(A_3) \\
MGR_{(a,b)}(J'(A), J(A_1), \phi) &\leq 1
\end{align*}$$

(16) (17) (18) (19)

where $Z_1$ and $Z_2$ represent the first objective function and the second objective function, respectively. Equations (18) and (19) are the constraints of the model.

This step is most appropriate to be performed after the entire mosaicking is completed. However, in order to achieve real-time correction of finger rolling, it can also be performed after each mosaicking. The time complexity of this step is $O(H \times W \times N \times 8)$. In more detail, calculating the MGR of a block requires $O(H' \times W' \times 8)$, as shown in Equation (15).

In the worst case, it needs to calculate the MGR of all blocks, and the number of these blocks is $H' \times W' \times N$. However, due to the limitation of Equation (18), the number of blocks we need to calculate is much smaller than the theoretical value.

V. EXPERIMENT

In this section, we first evaluate the impact of different block sizes on the performance of the proposed method. Then, we design various experiments to prove the effectiveness of the proposed method and compare it with the existing methods. Finally, we design an experiment to analyze the effectiveness of MGR on the location and elimination of mosaicking gaps.

A. SETUP & DATASETS

In this experiment, the optimization method based on dynamic programming is applied after each mosaicking, and multi-objective optimization is performed after the entire mosaicking. In Section III-A, the parameters T1 and T2 are set to 25 and 5185 in this experiment, respectively. To evaluate BlockRFC, the Green Bit DS84C rolled fingerprint sensor (800 × 800 pixels per frame, 500 DPI) is used [13]. When the finger is rolling on the scanner, the scanner continuously collects images. A typical rolled fingerprint sequence is shown in Figure 1. In this experiment, four hundred rolled fingerprint sequences are captured and recorded as dataset G1. They are from 80 different fingers, and each finger rolls five times. We randomly select one of each finger’s five rolled fingerprint sequences and record these rolled fingerprint sequences as dataset G2.

The number of images in each rolled fingerprint sequence is between 40 and 80. By using the method mentioned in Section III-A, some fingerprint images with low quality are filtered. In G1, we construct a complete rolled fingerprint for each finger’s rolled fingerprint sequences as a template database. Then, all high-quality images in the rolled fingerprint sequences are used as the sample database. The number of images in the sample database is 8316, and the number of images in the template database is 400.

B. EVALUATION METHODS

The performance evaluation of rolled fingerprint construction is often done indirectly, by computing metrics related to fingerprint matching performance: the better the performance of the fingerprint matching algorithm, the better (presumably) the construction method. In this article, a fingerprint matching algorithm [14] is used for fingerprint matching between the template database and sample database. Each template fingerprint is matched to each sample fingerprint, and a score as the matching result is the similarity between a template and a sample. The following metrics are used to evaluate performance: EER, FMR100, FMR1000, FMR10000, and ZeroFMR. The equal error rate (EER) represents the false non-matched ratio (FNMR) value is equal to the false matched ratio (FMR) [15]. FMR100 indicates that the FNMR value at FMR = 1%. FMR1000 indicates that the FNMR value at FMR = 0.1%. FMR10000 indicates that the FNMR value at FMR = 0.01%. ZeroFMR indicates that the FNMR value at FMR = 0%.

In [8], [9], [16], the most telling indicator of rolled fingerprint quality is the number of minutiae extracted. In this experiment, we use FingerNet [17] to extract the minutiae in rolled fingerprints. Besides, we provide some mosaicking results to observe the effect of distortion rectification visually.

C. SELECTION OF BLOCK SIZE

The length of the contact between adjacent blocks affects the degree of the mosaicking gap, so this part includes...
TABLE 2. Test of ten different block sizes.

| Block Size | EER (%) | FMR100 (%) | FMR1000 (%) | FMR10000 (%) | ZeroFMR (%) |
|------------|---------|------------|-------------|--------------|-------------|
| 10 × 10    | 3.18    | 4.12       | 5.95        | 8.35         | 11.54       |
| 20 × 10    | 2.08    | 2.35       | 3.57        | 5.36         | 8.23        |
| 20 × 20    | 1.59    | 1.84       | 3.06        | 4.66         | 7.33        |
| 40 × 10    | 1.51    | 1.57       | 2.33        | 3.59         | 5.94        |
| 40 × 20    | 1.69    | 1.87       | 2.80        | 3.73         | 7.34        |
| 40 × 40    | 1.94    | 2.26       | 3.15        | 4.20         | 7.27        |
| 80 × 10    | **0.99**| **0.99**   | **1.77**    | **2.89**     | **4.69**    |
| 80 × 20    | 1.46    | 1.57       | 2.51        | 3.90         | 6.06        |
| 80 × 40    | 1.45    | 1.62       | 2.78        | 4.46         | 9.87        |
| 80 × 80    | 1.83    | 2.04       | 2.89        | 4.05         | 6.35        |

FIGURE 6. The fingerprint images from left to right are the results of the algorithm-A, algorithm-B, algorithm-C and BlockRFC on the same rolled fingerprint image sequence respectively.

image blocks of different sizes and different aspect ratios to achieve comparative observation. The method mentioned in Section IV-B is mainly to optimize the mosaicking gap of adjacent image blocks in the horizontal direction, so the optimization of image blocks with large height and small width is more effective. In Table 2, ten image blocks with different sizes are used to compare the matching performance, and the size of the block is expressed by $\text{Height} \times \text{Width}$. There are three kinds of aspect ratios: 1:1, 2:1, and 4:1.

Table 2 shows that when the height of the image block is 80 pixels and the width is 10 pixels, the EER is 0.99%, which is better than the others. Therefore, 80 × 10 is determined as the block size of the proposed method.

D. EFFECTS OF OPTIMIZATION METHOD

In this part, the proposed method is compared with some representative image-level mosaic algorithms in recognition performance. The algorithms in literature [8] (called algorithm-A), literature [6] (called algorithm-B), literature [7] (called algorithm-C) are used for mosaicking. In particular, algorithm-A was the minimum compositing method, and its performance was best in literature [8]. Algorithm-A is often used as a benchmark for performance comparison with the latest rolled fingerprint construction method.

Figure 6 is a mosaic of the rolled fingerprint sequence in Figure 1. The reprint phenomenon is more obvious in Figure 6 (a), and there are a lot of dense mosaicking gaps in Figure 6 (b). Figure 7 is the magnified view of the fingertip region in Figure 6. Through careful observation, it can be found that there are some small mosaicking gaps in Figure 7 (c), but they are repaired in Figure 7 (d). Overall, BlockRFC is superior to the other three algorithms visually.

In terms of fingerprint recognition performance, Table 3 shows that BlockRFC has more significant advantages than the other three algorithms. Algorithm-A is a method similar to covering all fingerprints, which retains the complete contour of the fingerprint. But the ridges in the rolled fingerprint mosaicked by algorithm-A are chaotic, which makes it perform the worst. Algorithm-B gets a rolled fingerprint quickly, but it inevitably produces some dense mosaicking gaps. It is because of these mosaicking gaps that significantly affect its performance. BlockRFC has an improvement over algorithm-C because BlockRFC detects and optimizes many mosaicking gaps in the rolled fingerprint. As shown in Figure 7 (c) and (d), BlockRFC successfully repaired some mosaicking gaps that are difficult for human eyes to find.

Figure 8 shows the comparison of the number of minutiae in rolled fingerprints mosaicked by the four algorithms in the dataset $G_2$. The number of minutiae in the rolled fingerprint constructed by BlockRFC and algorithm-C significantly exceeds that of algorithm-B and algorithm-A. Although the results obtained by BlockRFC and algorithm-C are similar, we can judge from the matching performance and visual effects that the ridges obtained by BlockRFC are more accurate than the algorithm-C. Since algorithm-C does
not properly rectify the distortion, it produces some fake minutiae.

**E. EXPLORATION OF MGR**

According to the description in Section IV-B, MGR is used to indicate the degree of mosaicking gap between adjacent blocks. Therefore, the sum of MGR of all adjacent blocks in an image can be indirectly regarded as a metric to evaluate the quality of a rolled fingerprint. When the metric is quantified, its meaning needs to be defined artificially. Combined with experimental observations, the value of MGR is divided into the following five levels: [0, 8) indicates excellent, [8, 80) indicates good, [80, 160) indicates general, [160, 320) indicates bad, and [320, 800] indicates worse.

Figure 9 shows the comparison of mosaicking gaps between optimized and non-optimized. As shown in 9 (a), the gaps caused by mosaicking can be identified accurately and marked with different colors. Green indicates good, brown is general, and pink-purple indicates bad. As shown in 9 (b), after using the optimization methods to optimize the gaps, the optimization of the mosaicking gaps is visible, which shows that the original gaps have reduced, or even no gap. In past researches, the mosaicking gaps of the rolled fingerprints were found by the naked eye. It is of considerable significance to accurately find the mosaicking gaps and provide a direction for further eliminating it.

This part uses the rolled fingerprint construction method in Section III to obtain 80 rolled fingerprints from dataset G2, and records them as group A. Then, all the images in group A are optimized by using the method in Section IV to obtain new 80 rolled fingerprints, which are recorded as group B. Next, MGR is summed for each rolled fingerprint in group A and B to characterize its image quality indirectly. Figure 10 shows that the mosaicking gaps of all rolled fingerprints in group B are optimized compared to group A. Figure 11 shows the effect of the optimization method to group A.
Experimental results show that the average optimization rate of the proposed optimization method for all rolled fingerprints is 22.04%. The highest optimization rate of mosaicking gaps reaches 52%. Moreover, there are only 13 rolled fingerprints, whose optimization rate is less than 10%.

VI. CONCLUSION

We propose BlockRFC for real-time rolled fingerprint construction and distortion rectification. BlockRFC can evaluate and detect the mosaicking gaps during the finger rolling process, which makes it unnecessary for human experts to
supervise the whole process. Our experimental results show that BlockRFC can not only effectively detect the mosaicking gaps but also construct a high-quality rolled fingerprint image, which is of great help to improving the fingerprint matching performance. It is worth mentioning that this article originally puts forward MGR to evaluate the degree of the mosaicking gaps so that the mosaicking gaps can be accurately located and eliminated. In future work, we will try more optimization methods to eliminate fingerprint mosaicking gaps better. In terms of solving multi-objective optimization problems, many intelligent algorithms have advantages in performance, such as genetic algorithm and particle swarm optimization.

REFERENCES

[1] D. Peralta, M. Galar, I. Triguero, D. Paternain, S. García, E. Barrenechea, J. M. Benítez, H. Bustince, and F. Herrera, “A survey on fingerprint minutiae-based local matching for verification and identification: Taxonomy and experimental evaluation,” Inf. Sci., vol. 315, pp. 67–87, Sep. 2015.

[2] A. Kumat, “Introduction to trends in fingerprint identification,” in Contactless 3D Fingerprint Identification. Cham, Switzerland: Springer, 2018, pp. 1–15. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-67681-4_1

[3] J. Feng, S. Yoon, and A. K. Jain, “Latent fingerprint matching: Fusion of rolled and plain fingerprints,” in Proc. Int. Conf. Biometrics. Berlin, Germany: Springer, 2009, pp. 695–704. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-01793-3_71

[4] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition. London, U.K.: Springer, 2009.

[5] D. Kwon, I. Dong Yun, and S. U. Lee, “Rolled fingerprint construction based on key-column extraction,” IEEE Trans. Image Process., vol. 19, no. 12, pp. 3255–3270, Dec. 2010.

[6] Y. Zhang, S. Fang, Y. Bian, and Y. Li, “Real-time rolled fingerprint construction based on key-column extraction,” in Proc. Chin. Conf. Biometric Recognit. Cham, Switzerland: Springer, 2013, pp. 201–207. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-02961-0_25

[7] Y. Zhang, M. Gao, X. Zhan, Y. Wu, and S. Pan, “Rolled fingerprint mosaicking algorithm based on block scale,” in Proc. Chin. Conf. Biometric Recognit. Cham, Switzerland: Springer, 2019, pp. 55–62. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-31456-9_7

[8] N. K. Ratha, J. H. Connell, and R. M. Bolle, “Image mosaicing for rolled fingerprint construction,” in Proc. Iet. Int. Conf. Pattern Recognit., vol. 2, Brisbane, QLD, Australia: IEEE Press, Dec. 1998, pp. 1651–1653. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/712035, doi: 10.1109/ICPR.1998.712035.

[9] J. Zhou, D. He, G. Rong, and Z.-Q. Bian, “Effective algorithm for rolled fingerprint construction,” Electron. Lett., vol. 37, no. 8, pp. 492–494, Apr. 2001.

[10] B. Stojanovic, O. Marques, and A. Neskovic, “Deep learning-based approach to latent overlapped fingerprints mask segmentation,” IET Image Process., vol. 12, no. 11, pp. 1934–1942, Nov. 2018.

[11] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), vol. 1, Dec. 2001, p. 1.

[12] J. Chen, H. Zhao, Z. Cao, W. Zhao, and L. Pang, “Successive minutia-free mosaicking for small-sized fingerprint recognition,” IET Image Process., vol. 13, no. 7, pp. 1146–1151, May 2019.

[13] GreenBit. Ds9AC Rolled Fingerprint Sensor. [Online]. Available: http://www.greenbit-china.cn/index.php?m=content&c=index&a=show&catid=17&idx=10

[14] Y. Zhang, “Algorithm study on swipe fingerprint mosaicking and fingerprint matching,” M.S. thesis, Dept. Pattern Recognit. Innt. Syst., Shanghai Jiao Tong Univ., Shanghai, China, 2006.

[15] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain, “FV/C2000: Fingerprint verification competition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 3, pp. 402–412, Mar. 2002.

[16] K. Choi, H.-S. Choi, and J. Kim, “Fingerprint mosaicking by rolling and sliding,” in Proc. Int. Conf. Audio Video-Based Biometric Person Authentication. Berlin, Germany: Springer, 2005, pp. 260–269. [Online]. Available: https://link.springer.com/chapter/10.1007/11527923_27

[17] Y. Tang, F. Gao, J. Feng, and Y. Liu, “FingerNet: An unified deep network for fingerprint minutiae extraction,” in Proc. IEEE Int. Joint Conf. Biometrics (IJCB), Oct. 2017, pp. 108–116.

**Yongliang Zhang** received the B.S. and M.S. degrees from Jilin University, China, in 2000 and 2003, respectively, and the Ph.D. degree from Shanghai Jiao Tong University, China, in 2007. He is currently an Associate Professor with the College of Computer Science and Technology, Zhejiang University of Technology. His research interests include biometric identification, pattern recognition, and artificial intelligence.

**Yifan Wu** was born in Taizhou, China, in 1998. He is currently pursuing the B.S. degree in software engineering with the Zhejiang University of Technology, China. His research interests include biometric identification, pattern recognition, and machine learning.

**Shengyi Pan** was born in Hangzhou, China, in 1999. He is currently pursuing the B.S. degree in computer science with the Zhejiang University of Technology, China. His research interests include biometric identification, pattern recognition, and machine learning.

**Zirui Shao** was born in Hangzhou, China, in 1999. He is currently pursuing the B.S. degree in software engineering with the Zhejiang University of Technology, China. His research interests include biometric identification, pattern recognition, and machine learning.

**Tian Luo** was born in Guilin, China, in 2000. He is currently pursuing the B.S. degree in computer science with the Zhejiang University of Technology, China. His research interests include biometric identification, pattern recognition, and machine learning.