ELSEVIER

Image and Vision Computing 16 (1998) 677–687

image AND vision COMPUTING

Projection for pattern recognition

Chiou-Shann Fuh*, Horng-Bin Liu

Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, ROC

Received 10 February 1997; accepted 13 October 1997

Abstract

This paper presents a method of pattern recognition whereby patterns can be enlarged, shrunk, rotated, sheared or mirrored. The program generates a series of projections at different angles, then uses these projections to match images in the pattern database. Our algorithm performs satisfactorily in a large pattern database and is also robust under noise.

Keywords: Pattern recognition; Projections

1. Introduction

Projection [2,6], first used in optical character recognition [8], is important in binary image recognition because it can be computed in real time [10]. Computer tomography [4,14] also uses one-dimensional projections at different angles to reconstruct the original two-dimensional image. When a pattern is enlarged, shrunk, rotated or mirrored, human beings can easily determine what happened to that pattern. But this task is not trivial for a computer. We present a method that can easily detect if the pattern has been enlarged, shrunk or mirrored, and by how many degrees it has been rotated [9]. On the other hand, our method can also recognize the pattern from a set of patterns and is robust to noise and shear.

There are other methods for shift, rotation and scale-invariant pattern recognition [1]. Traditional machine vision systems for pattern recognition are based on feature extraction followed by classification. Shepherd et al. [11] use feature-based pattern recognition, using the shapes and arrangement of local features to recognize patterns. Spirkovska and Reid [12] use higher-order neural networks, which reduces the training time significantly. Flusser and Suk [5] use affine moment invariants for pattern recognition. Stein and Medioni [13] proposed structural indexing whereas Lamdan and Wolfson [7] proposed geometric hashing for invariance recognition.

We know the computational time for feature extraction is dependent on input image: the more complex the image, the more computational time is required. Furthermore, the recognition time is not always fixed, and the descriptors of features also vary. With neural networks, each modification of the database requires additional training time and the recognition time is not fixed. With the moment method, there is the problem of sensitivity to noise. Our method is superior to the preceding two methods because it fixes recognition time, can fix the size of feature descriptors, requires no training time and is robust to noise.

2. Projection

Projection is an important feature in this method. A projection is a histogram of the nonzero pixels of the image in some parallel lines. For image u, the image size is H-pixel $\times$ W-pixel, which is always fixed, and the pattern size is $M_u(\theta)$-pixel $\times$ $N_u(\theta)$-pixel, which depends on nonzero pixels and is equal to the size of the bounding box after the pattern rotates θ degrees counterclockwise from the x-axis as shown in Fig. 1. In Fig. 1, our pattern is a rectangle of size m-pixel $\times$ n-pixel. When $\theta = 0°$, we have $M_u(0°) = m$ and $N_u(0°) = n$. When θ is nonzero, we have $M_u(\theta) = n|\sin\theta| + m|\cos\theta|$ and $N_u(\theta) = n|\cos\theta| + m|\sin\theta|$. We assume $M_u(0°) \le H$, $N_u(0°) \le W$, and thus we have $M_u(\theta) \le \sqrt{H^2 + W^2}$, $N_u(\theta) \le \sqrt{H^2 + W^2}$, for any image u.

Projection $P_u(\theta)$ is the projection along the line of $\theta°$ counterclockwise from the x-axis. Degree θ is the projection angle and is defined from 0° to 180°. The number of projection cells is always equal to $N_u(\theta)$ rounded to the nearest integer. The order of projection cells is defined from lower left to upper right when $0° \le \theta \le 90°$, and from lower right to upper left when $90° < \theta \le 180°$, as shown in Fig. 2. Projection cell $p_u(\theta, i)$ is the projection in the index i, as

* Corresponding author.

0262-8856/98/$19.00
PII S0262-8856(98)00080-8

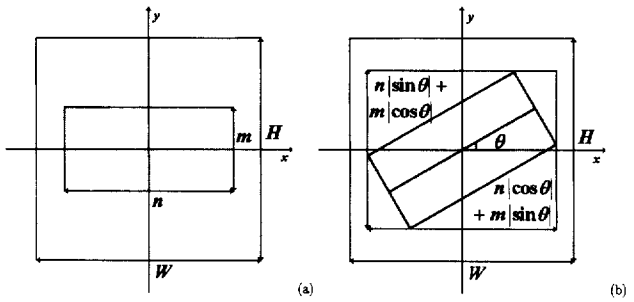

Fig. 1. The image size is always H-pixel × W-pixel, and the pattern size is affected by image u and projection angle θ. (a) $\theta = 0°$; (b) $\theta \neq 0°$.

shown in Fig. 3.

$$P_u(\theta) = \{p_u(\theta, 1), p_u(\theta, 2), p_u(\theta, 3), \ldots, p_u(\theta, N_u(\theta))\} \quad (1)$$

2.1. Property 1

Let $\overline{P_u(\theta)}$ be the projection with the same projection cells of $P_u(\theta)$ but with the order of the projection cells reversed:

$$\overline{P_u(\theta)} = \{p_u(\theta, N_u(\theta)), p_u(\theta, N_u(\theta) - 1), p_u(\theta, N_u(\theta) - 2), \ldots, p_u(\theta, 1)\} \quad (2)$$

Property 1 is that when an image is rotated by 180° counterclockwise from the x-axis we have

$$P_u(\theta + 180°) = \overline{P_u(\theta)}. \quad (3)$$

Fig. 4 shows this property.

2.2. Property 2

As shown in Figs. 5–8, when an image is symmetric to another image with respect to the y-axis (mirrored at the y-axis), horizontal projections remain the same (E = e), vertical projections have a reverse order (A = ā), and diagonal projections are switched (B = h, C = g, D = f, F = d, G = c, H = b). The equivalence relation is shown in Fig. 9.

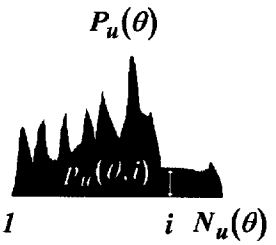

Fig. 3. Projection cell $p_u(\theta, i)$ is the projection of image u in the index i at angle θ.

Let circular projections, $Q_u(a, \phi)$, be defined as a set of projections of image u in equally spaced angles. Degree a is the initial angle and degree ϕ is the angle step. Also, we let ρ be the size of the circular projections. Size ρ is always an integer.

$$\phi = \frac{180°}{\rho} \quad (4)$$

$$Q_u(a, \phi) = \{P_u(a + 0°), P_u(a + \phi), P_u(a + 2\phi), \ldots, P_u(a + (\rho - 1)\phi)\} \quad (5)$$

Now let $Q_u'(a, \phi)$ be $Q_u(a, \phi)$ with its elements in decreasing order except for the first projection, and the order of the first projection cells is in reversed order:

$$Q_u'(a, \phi) = \{\overline{P_u(a + 0°)}, P_u(a + (\rho - 1)\phi), P_u(a + (\rho - 2)\phi), P_u(a + (\rho - 3)\phi), \ldots, P_u(a + \phi)\} \quad (6)$$

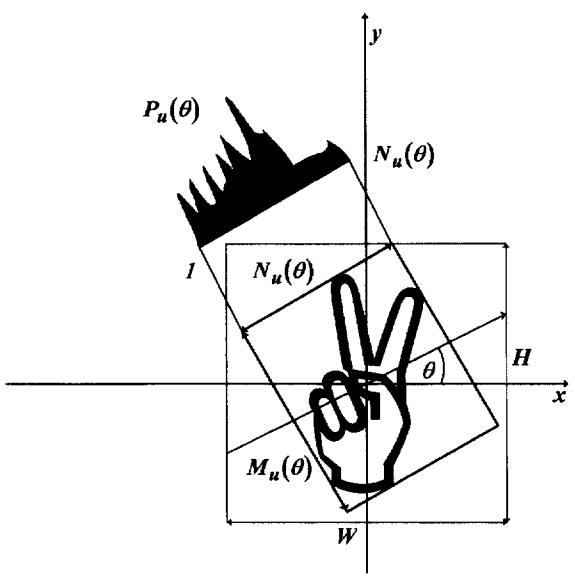

Fig. 2. The projection of an image along the line of θ degrees counterclockwise from the x-axis.

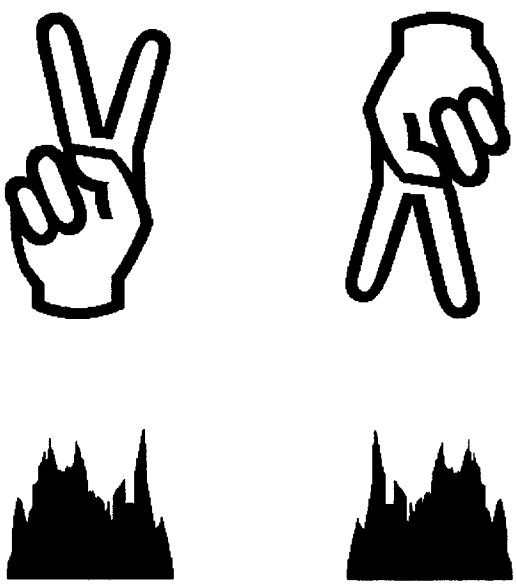

Fig. 4. When an image is rotated by 180°, the projection cells are the same but the order of the projection cells is reversed.

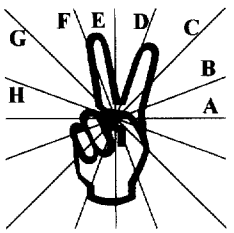

Fig. 5. An example image in eight projection directions: A, B, C, D, E, F, G and H.

Thus, when the initial angle $a = 0°$,

$$Q_u'(0°, \phi) = \{\overline{P_u(0°)}, P_u((\rho-1)\phi), P_u((\rho-2)\phi), P_u(a+(\rho-3)\phi), \ldots, P_u(\phi)\}. \quad (7)$$

If image v is symmetric to image u with respect to the y-axis, the following equation is true:

$$Q_v(0°, \phi) = \{P_v(0°), P_v(\phi), P_v(2\phi), \ldots, P_v((\rho-1)\phi)\} = \{\overline{P_u(0°)}, P_u((\rho-1)\phi), P_u((\rho-2)\phi), P_u((\rho-3)\phi), \ldots, P_u(\phi)\} = Q_u'(0°, \phi) \quad (8)$$

Now we can arrive at Property 2: if image v is symmetric to image u with respect to the y-axis, it is true that

$$Q_v(0°, \phi) = Q_u'(0°, \phi), \text{ and } Q_v(a, \phi) = Q_u'(-a, \phi).$$

The situation $Q_v(a, \phi) = Q_u'(-a, \phi)$ is shown in Fig. 10. If image v is rotated by a degrees, equivalently the symmetric image u is rotated by $-a$ degrees, and they are still symmetric with respect to the y-axis.

Define $\overline{Q_u(a, \phi)}$ to be

$$\overline{Q_u(a, \phi)} = \{\overline{P_u(a+0°)}, \overline{P_u(a+\phi)}, \overline{P_u(a+2\phi)}, \ldots, \overline{P_u(a+(\rho-1)\phi)}\}. \quad (9)$$

Using Property 1, we also have

$$Q_u(a+180°, \phi) = \overline{Q_u(a, \phi)}. \quad (10)$$

If image r is symmetric to image u with respect to the x-axis, then we have

$$Q_r(0°, \phi) = \overline{Q_u'(0°, \phi)}. \quad (11)$$

If image u is symmetric to image r with respect to the x-axis, every pixel (x, y) in image u is equal to pixel $(-x, y)$ in image r. Let image s be image u rotated by 180°. Now every pixel (x, y) in image s is equal to pixel $(-x, -y)$ in image u and pixel $(x, -y)$ in image r. Image s is symmetric

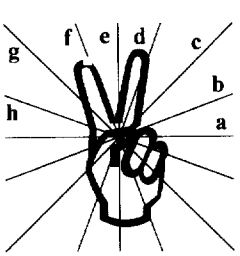

Fig. 6. This image is symmetric to the image in Fig. 5 with respect to the y-axis. The eight projection directions are a, b, c, d, e, f, g and h.

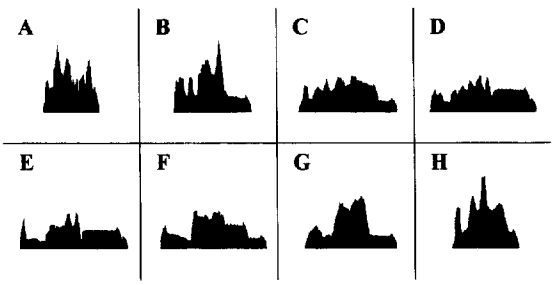

Fig. 7. The eight projections (A, B, C, D, E, F, G and H) of the image in Fig. 5.

to image r with respect to the y-axis. We get

$$Q_r(0°, \phi) = Q_s'(0°, \phi) = \overline{Q_u'(0°, \phi)}. \quad (12)$$

2.3. Property 3

If $\phi \le 2\sin^{-1}\{1/(\sqrt{H^2+W^2})\}$ and $0° \le a \le \phi/2$, then $Q_u(0°, \phi) = Q_u(a, \phi)$. When two circular projections of the same image are taken at different initial angles, the maximum difference of these two circular projections will occur at $a = \phi/2$ for a fixed angle step ϕ. Fig. 11 shows this condition, and the details will be explained in this section.

For comparison, let us look at the difference between two vertical projections. The other projections have the same following characteristic: after a pattern of width W and height H rotates with respect to its center it will be within a radius of $(\sqrt{H^2+W^2})/2$ pixels. Let A be the image pixels projected to the projection cell at $x = 0$ and let the cell have domain $[-0.5, 0.5]$. If this image has an initial angle of $\phi/2$, we know that pixel set A will be rotated to pixel set B. If we want these two pixel sets, A and B, to have the same projection value at $x = 0$, we must have

$$\frac{0.5}{\sin\frac{\phi}{2}} \ge \frac{\sqrt{H^2+W^2}}{2} \quad (13)$$

Then we get

$$\phi \le 2\sin^{-1}\frac{1}{\sqrt{H^2+W^2}}. \quad (14)$$

If $\phi > 2\sin^{-1}\{1/\sqrt{H^2+W^2}\}$, some pixels of pixel set B will not be projected to the cell at $x = 0$ after pixel set A is

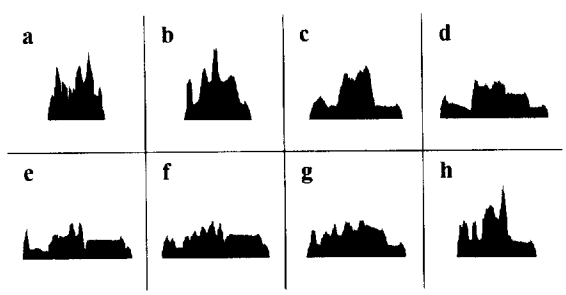

Fig. 8. The eight projections (a, b, c, d, e, f, g and h) of the image in Fig. 6.

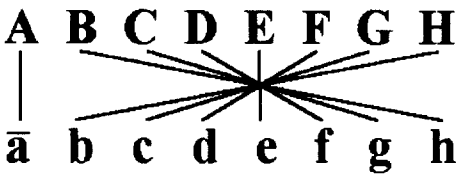

Fig. 9. The equivalence relation between the projections in Figs. 5 and 6.

rotated to pixel set B. We define the difference rate $e\%$ to be the number of these pixels that are not projected to the cell at $x = 0$ after the rotation divided by the total number of pixels in pixel set A. If we can have $e\%$ difference rate, we have

$$\left(\frac{0.5}{\sin\frac{\phi}{2}}\right)/\left(\sqrt{H^2+W^2}/2\right) \geq (1-e\%). \tag{15}$$

The equation becomes

$$\phi \leq 2\sin^{-1}\frac{1}{\sqrt{H^2+W^2}(1-e\%)}. \tag{16}$$

On the other hand, if we fix angle step ϕ, we get the following characteristic. The pixels near the rotation center have a low difference rate, and the pixels far from the rotation center have a high difference rate. This phenomenon is shown in Fig. 12. When the projection cell is far off the origin, the value of the projection cell is always less than $\sqrt{H^2+W^2}$, and we can have a larger angle step ϕ.

2.4. Scaling, shearing and projection

When two patterns have the following feature, we say these two patterns have different scales. The feature is: for every pixel (x_u, y_u) in image u, there is one and only one pixel $(x_v, y_v) = (S_x x_u + C_x, S_y y_u + C_y))$ in image v which will match, where S_x, S_y, C_x and C_y are the constants. (The same holds true if images u and v are interchanged.) If we calculate the projections of images u and v, the projections are so different in size and amplitude that it is difficult to match their projections. A better way is to normalize these two patterns into a fixed and predefined size, after

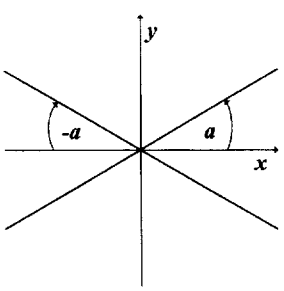

Fig. 10. If image v is symmetric to image u with respect to the y-axis, we get $Q_v(a, \phi) = Q_u'(-a, \phi)$. If image v is rotated by a degrees, equivalently, the symmetric image u is rotated by $-a$ degrees, and they are still symmetric with respect to the y-axis.

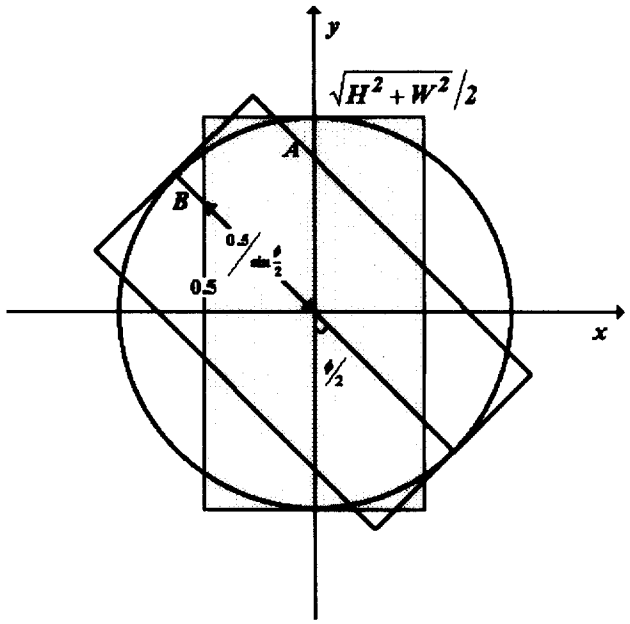

Fig. 11. When two circular projections of the same image are taken at different initial angles, the maximum difference of these two circular projections will occur at $a = \phi/2$.

which it is easy to match their projections. If the normalization process is perfect, we can get the same projections. In practice, we always get some difference in their projections due to quantization and noise, but their difference is trivial and will not affect the recognition.

When two patterns have the following feature, we say these two patterns have different sheared degrees. The feature is: for every pixel (x_u, y_u) in image u, there is one and only one pixel $(x_v, y_v) = (x_u + y_u \tan\theta_y, y_u + x_u \tan\theta_x)$ in image v which will match, where θ_x and θ_y are the constants. (The same holds true if images u and v are interchanged.) If

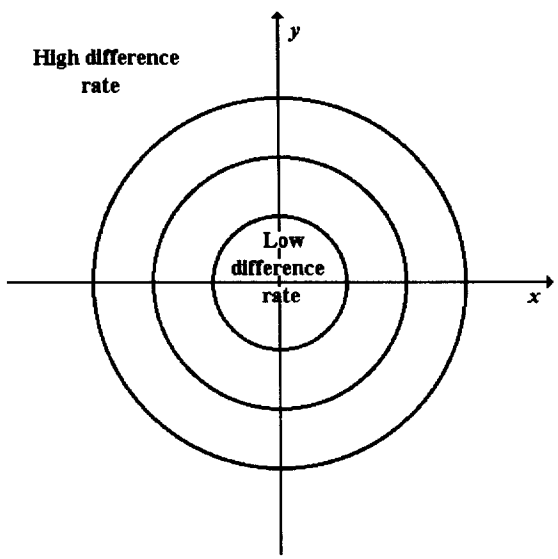

Fig. 12. When we fix angle step ϕ, the difference rate is affected by the distance from the pixel to the rotation center.

$\theta_x = -\theta_y = \theta$, we have

$$(x_v, y_v) = (x_u + y_u \tan\theta_y,\ y_u + x_u \tan\theta_x)$$
$$= \left(\frac{x_u \cos(-\theta) + y_u \sin(-\theta)}{\cos(-\theta)},\ \frac{y_u \cos\theta + x_u \sin\theta}{\cos\theta}\right)$$
$$= \frac{1}{\cos\theta}(x_u \cos\theta - y_u \sin\theta,\ y_u \cos\theta + x_u \sin\theta). \quad (17)$$

When $\theta_x = -\theta_y = \theta$, we know that image v is the image u rotated by a degree θ and scaled by a factor of $1/\cos\theta$. When we calculate the projection of a pattern, we care about the displacement of the x-coordinate but not the y-coordinate. If $-10° \le \theta_x \le 10°$, we have $x_u - 0.1763y_u \le x_v \le x_u + 0.1763y_u$, where $\tan 10° = 0.176327$. When we recognize a pattern, we do not use each individual projection cell, but average some projection cells into a new projection cell. The displacement of shearing does not greatly affect our recognition approach. On the other hand, when a pixel moves outside this projection cell, another pixel may move into this projection cell. Small shearing does not affect the recognition accuracy much.

3. Method

Fig. 13 is our algorithm. After inputting an image u, there are four main steps to manipulate input image u. They are image generation, normalization process, weighted projection, and matching process. They are explained in the following paragraphs.

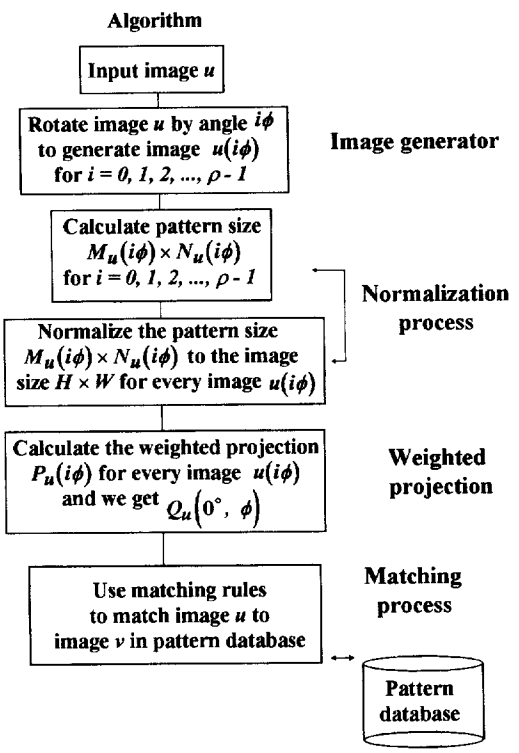

Fig. 13. Our algorithm.

3.1. Image generation

It is hard to calculate the projections directly from the input image from different angles. Using the equivalent method, we rotate the image through different angles [3] and then calculate the vertical projections. After we have rotated the input image, the pattern size may be larger than the image size. So we must put the rotated pattern in a larger frame. After this step, we can get image $u(0°)$ to image $u((\rho-1)\phi)$. Let $u(j\phi)$ be the image rotated by $j\phi$ degrees counterclockwise from the x-axis, where j ranges from 0 to $\rho - 1$. An example is shown in Fig. 14.

3.2. Normalization process

If we now calculate the vertical projections for image $u(0°)$ to image $u((\rho-1)\phi)$, the number of projection cells may vary, making the matching process very difficult. A better method is to normalize the size of the pattern, $M_u(\theta)$-pixel $\times$ $N_u(\theta)$-pixel, to some fixed size. We set this fixed size to equal the image size, H-pixel $\times$ W-pixel. The algorithm is shown below.

For every pixel (i, j) in the normalized image calculate the pixel value at $\left(\frac{i(M_u(\theta)-1)}{H-1}, \frac{j(N_u(\theta)-1)}{W-1}\right)$ in the pattern where the normalized image coordinate is in $(0, 1, 2, \ldots, H-1) \times (0, 1, 2, \ldots, W-1)$ and the pattern coordinate is in $(0, 1, 2, \ldots, M_u(\theta)-1) \times (0, 1, 2, \ldots, N_u(\theta)-1)$.

Let $\hat{u}(j\phi)$ be the normalized image $u(j\phi)$ where j ranges from 0 to $\rho - 1$. We can get images $\hat{u}(0°)$, $\hat{u}(\phi)$, $\hat{u}(2\phi)$, ..., $\hat{u}((\rho-1)\phi)$. After the normalization process, the number of projection cells is W. In our experiments, the image size is H-pixel $\times$ W-pixel = 128-pixel $\times$ 128-pixel. After the normalization process, we always get a fixed number of projection cells. When we want to compare two projections, having the same number of projection cells will make the comparison easy. But when the pattern is enlarged or shrunk, the normalization process will leave the pattern unchanged. Thus we can get the same projections. Although partial occlusion affects our normalization method, our recognition method is robust to partial occlusion. If the occlusion is not great, say 5% of the pattern area disappears, we consider the occlusion as noise. After the normalization process, the pattern will have the same scale and be invariant in scaling. An example is shown in Fig. 15. Property 3 still holds true after the normalization process.

3.3. Weighted projection

From Property 3, we can define different weights of pixels when we calculate vertical projections. The pixels near the rotation center have a high weight, and the pixels far from the rotation center have a low weight. We choose the origin (i.e. the center of the image) to be the rotation

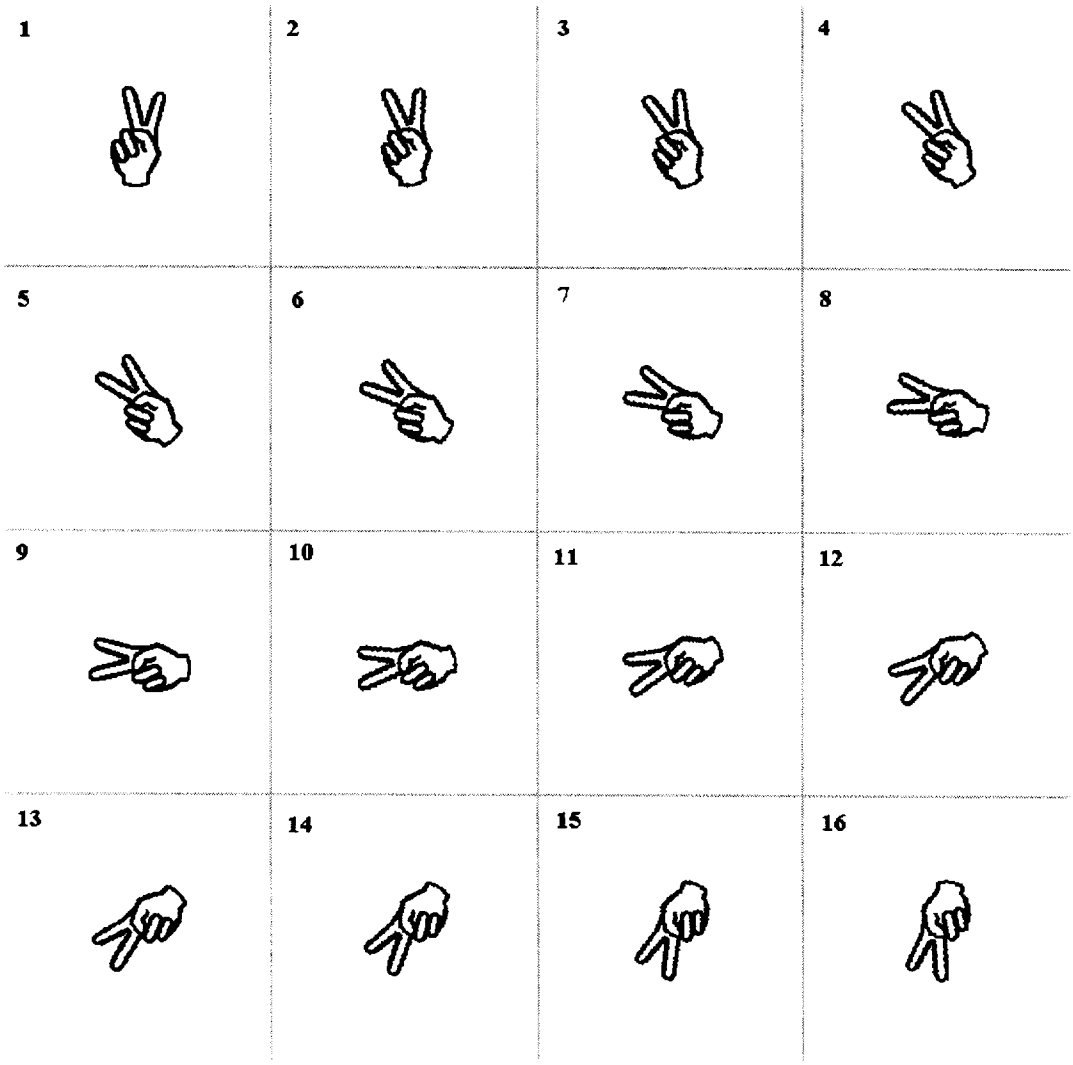

Fig. 14. An example is shown after image generation in which $\rho = 16$ and $\phi = 11.25°$.

center. The weight values in our experiments are shown in Table 1.

We select the projection size ρ to be 16, so we can get the angle step ϕ equal to 11.25°. When the number of projections is a power of two, we can always divide the projections into two sections and then divide each section into two smaller sections and so on. A number of projections which is a power of two is more convenient than a number of projections which is not a power of two. Eight projections are not sufficient, 16 projections are sufficient, but 32 projections are too time consuming. So we selected 16 projections.

Table 1
The weights in our experiments

Distance from center (d pixels)	Weight
$0 \le d \le 5$	1.45
$5 < d \le 15$	1.40
$15 < d \le 25$	1.35
...	...
$85 < d \le 95$	1.00

From Property 3, we have 0.5/(sin 5.625°) = 5.10 and 1.0/(sin 5.625°) = 10.20. So the innermost part has a radius of 5 pixels, whereas the other parts are increased by 10 pixels from inner to outer. We made the weight of the pixels farthest from the center equal to 1.00, and we did not lose much information from these pixels. We also did not want to make the pixels nearest the center have too high a weight; otherwise, these pixels would dominate the weight projections. A weight of 1.45 or 1.50 is a good choice. But if the weight is 1.50, then the increase in volume will be 0.5/9. So we chose the pixels nearest to the center to have a weight of 1.45, with the increase in volume being 0.05.

The number of projection cells was equal to W, and this was so large that the matching process resulted in data-heavy calculations. Averaging several projection cells into one new projection cell was a good method to speed up the computation. On the one hand it shortened the calculation time, and on the other hand it suppressed the noise in the image. In our experiments, W was equal to 128, and we chose to average every 8 projection cells and got 16 new

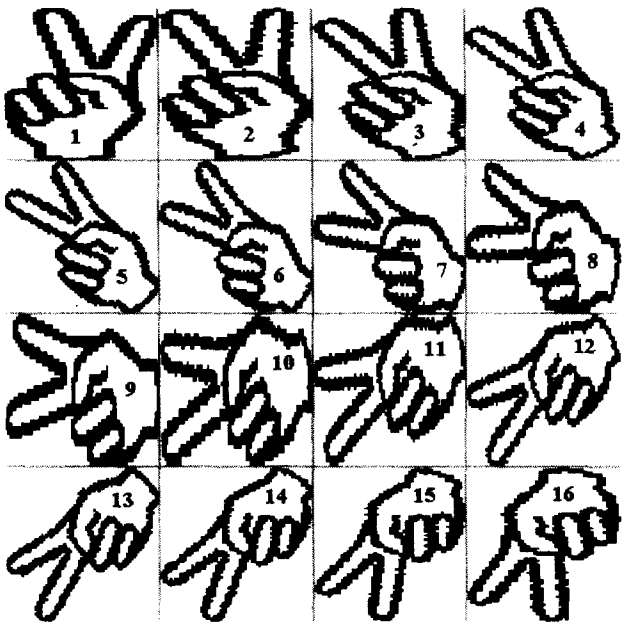

Fig. 15. An example of the normalization process is shown in which $H = W = 128$.

projection cells. Sixteen projection cells were sufficient. We also tried to average 16 projection cells into 8 new projection cells, but this performed poorly because the number of projection cells was too few to distinguish similar patterns. In a weighted projection, projection $p_u(\theta, k)$ was the sum of the weights of nonzero pixels instead of the number of nonzero pixels in column k at $\theta°$. After a weighted projection the shearing operation does not affect projection much, so our recognition method is robust to shearing. An example is shown in Fig. 16.

3.4. Matching process

The main rule is least-square difference, but there are other rules to enhance the matching process. Define the circular projections difference $Q_u(0°, \phi) - Q_v(\phi i, \phi)$ to be

$$Q_u(0°, \phi) - Q_v(\phi i, \phi) = \sum_{j=0}^{\rho-1} \sum_{k=1}^{W} (|p_u(\phi j, k) - p_v(\phi i + \phi j, k)|)^2. \quad (18)$$

Let the projection difference, $\mathrm{pd}_{uv}(\theta_1, \theta_2)$, between image u at $\theta_1°$ and image v at $\theta_2°$ be

$$\mathrm{pd}_{uv}(\theta_1, \theta_2) = \sum_{k=1}^{W} (|p_u(\theta_1, k) - p_v(\theta_2, k)|)^2. \quad (19)$$

Then the image difference $E(u, v)$ is the least-square difference of images u and v:

$$\begin{aligned} E(u, v) = \min\{ & \min_{i=0,1,\ldots,\rho-1} [Q_u(0°, \phi) - Q_v(\phi i, \phi)], \\ & \min_{i=0,1,\ldots,\rho-1} [Q_u(0°, \phi) - \overline{Q_v(\phi i, \phi)}], \\ & \min_{i=0,1,\ldots,\rho-1} [Q_u(0°, \phi) - Q_v'(\phi i, \phi)], \\ & \min_{i=0,1,\ldots,\rho-1} [Q_u(0°, \phi) - \overline{Q_v'(\phi i, \phi)}]\}. \end{aligned} \quad (20)$$

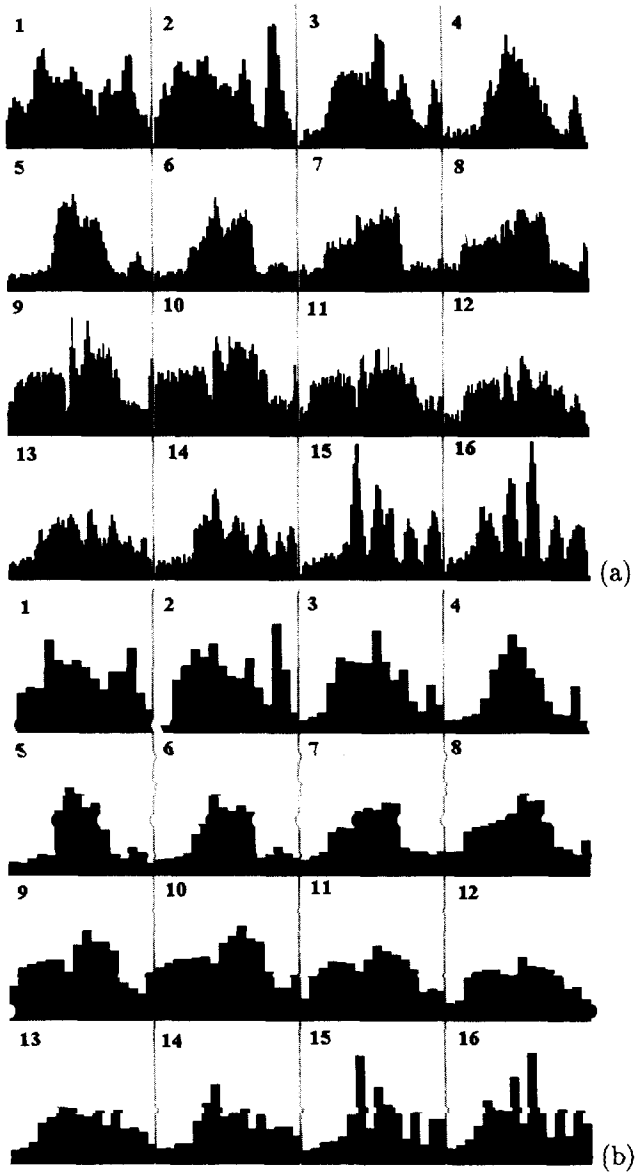

Fig. 16. An example is shown after the weighted projection. (a) Weighted projections with 128 projection cells. (b) Average weighted projections with 16 projection cells.

The first term is the difference of images u and v with the initial angle of image v less than 180°. The second term is the difference of images u and v with the initial angle of image v more than 180°, so $\overline{Q_v(\phi i, \phi)}$ is used. The third term is the difference of images u and v' which is symmetric to image v with respect to the y-axis with the initial angle less than 180°, so $Q_v'(\phi i, \phi)$ is used. The fourth term is the difference of images u and v' which is symmetric to image v with respect to the y-axis with the initial angle more than 180°, so $\overline{Q_v'(\phi i, \phi)}$ is used.

3.4.1. Rule 1

Let $A(\hat{u}(\theta))$ be the area of pattern in a normalized image u

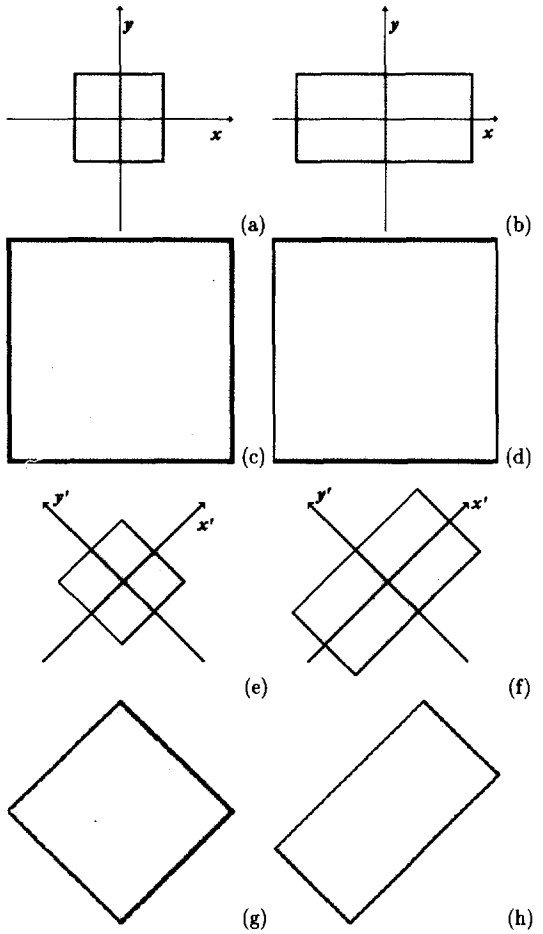

Fig. 17. When the pattern is a square, and we enlarge the square × 2 in the x-axis direction and do not change it in the y-axis direction, then the square becomes a rectangle. Now we normalize the square and the rectangle, and they are similar. But when we rotate them and normalize them, they are very different. (a) The input image is a square. (b) The square becomes a rectangle. (c) Normalized square. (d) Normalized rectangle. (e) Rotated square. (f) Rotated rectangle. (g) Normalized rotated square. (h) Normalized rotated rectangle.

at $\theta°$. For input image u and any image v in our pattern database, if $|A(\hat{u}(\theta_1)) - A(\hat{v}(\theta_2))| \leq A(\hat{v}(\theta_2)) \times 10\%$, then we calculate the projection difference $\mathrm{pd}_{uv}(\theta_1, \theta_2)$ between image u at $\theta_1°$ and image v at $\theta_2°$. Otherwise, we do not calculate the projection difference.

The weighted projections are projections with position information. The position near the rotation center has a high weight and the position far from the rotation center has a low weight. It is possible that the weighted projections are similar, but the patterns are very different. Thus we add another feature to distinguish them. We select the feature to be the area of the pattern. For input image u and any image v in our pattern database, if $|A(\hat{u}(\theta_1)) - A(\hat{v}(\theta_2))| \leq A(\hat{v}(\theta_2)) \times 10\%$, then we calculate their projection difference, $\mathrm{pd}_{uv}(\theta_1, \theta_2)$. Otherwise, we do not have to calculate the projection difference. If the difference of the areas of these two images is too large, then these two images should never be matched together.

We do not use every area of the input image at different rotation angles for matching. We choose the image at 0° and 90°, because the areas of the image at 0° and 90° stay unchanged even if the image is scaled in either the x or the y direction, thanks to normalization in both directions. For example, when the input image is a square, and we enlarge the square × 2 in the x direction and do not make any changes in the y direction, then the square becomes a rectangle. Now we normalize the square and the rectangle at 0°, and they are similar. But when we rotate them $\theta°$, $\theta \neq 90°$, and normalize them, they are very different, as shown in Fig. 17. The changed scales are in the x'-axis and the y'-axis, and the normalization process changes the scales in the x-axis and the y-axis. When the rotation angle is equal to 90°, the area of the pattern also stays unchanged.

3.4.2. Rule 2

From Rule 1, we know the projections at 0° and 90° stay unchanged. Thus, when we accumulate projection differences, the projection differences at 0° and 90° have higher weights. In our experiments, the projection differences at 0° and 90° have a weight of 4, and the others have a weight of 1. We fix the projection differences that are not at 0° and 90° to have a weight of 1, and change the weight of the projection differences at 0° and 90°. Weight 4 is better and is chosen from the experimental results.

4. Experimental results

Fig. 18 is our pattern database with 387 patterns. The input image size is 128 pixels × 128 pixels. We enlarge or shrink, rotate, mirror and shear our patterns, and then match these distorted patterns to our pattern database. There are seven parameters affecting the input patterns: scale x, scale y, shear x, shear y, rotation angle θ, mirror, and noise. Scales x and y are the scales in the x-axis and the y-axis respectively. When scale $x = 2$ and scale $y = 0.5$, the size of the pattern is enlarged × 2 in the x direction and is shrunk × 2 in the y direction. Shears x and y are the sheared angles in the x direction and the y direction respectively. When shear $x = 10°$ and shear $y = 5°$, every pixel (x, y) in the pattern will move to pixel $(x + y \tan 5°, y + x \tan 10°)$. For example, when the pattern is a rectangle and shear $x = 10°$ and shear $y = 5°$, the rectangle will become a parallelogram, as shown in Fig. 19. Our experiments show that our algorithm can tolerate a shear of 10°.

Rotation angle θ means to rotate the pattern $\theta°$ counterclockwise from the x-axis. Mirror means whether the pattern is mirrored with respect to the y-axis. When mirror is 'Y', it means the pattern is mirrored with respect to the y-axis. When mirror is 'N', it means the pattern is not mirrored at all. Noise indicates whether noise is added to the pattern. In some experiments, we add a square of 5% area of the pattern in the frame of the pattern size to be the noise.

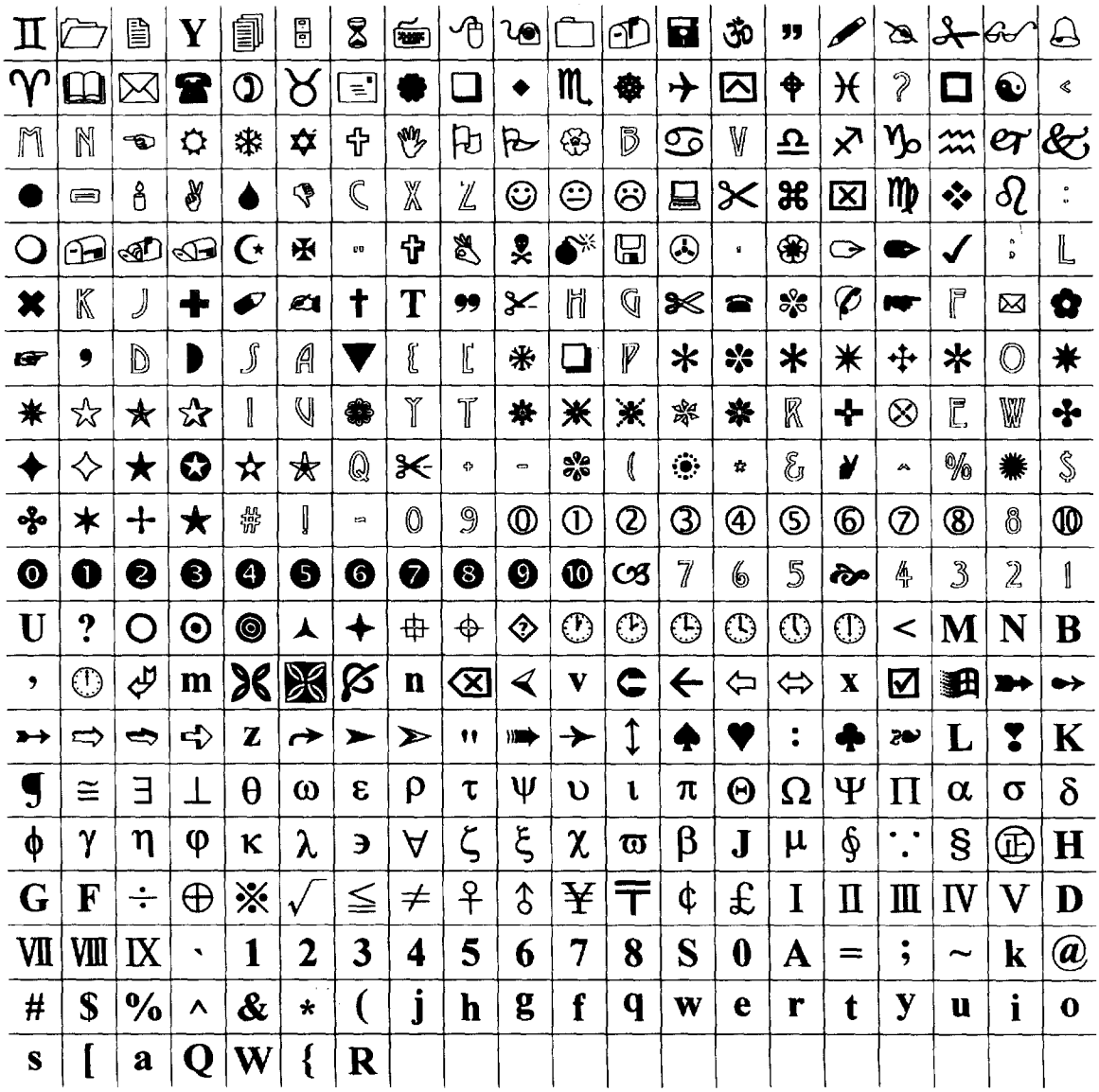

Fig. 18. The pattern database with 387 patterns.

We now show an example in Fig. 20. When we enlarge, shrink, rotate or mirror the pattern, we can still recognize the pattern, as shown in Fig. 20(a). But in some conditions we may recognize the pattern incorrectly, as shown in

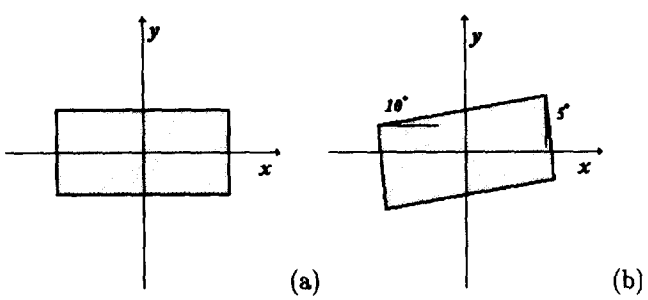

Fig. 19. When the pattern is a rectangle and shear $x = 10°$ and shear $y = 5°$, the rectangle will become a parallelogram. (a) A rectangle. (b) The sheared rectangle becomes a parallelogram.

Table 2
The parameters of patterns in Fig. 20. We first rotate, then scale, and finally shear the patterns

Label	Scale		Shear		Rotation angle θ	Mirror
	x	y	x	y		
1, C	Original pattern					
2	1.00	1.00	0°	0°	20°	N
3	1.00	1.00	10°	10°	40°	N
4	1.25	1.00	0°	0°	100°	N
5	1.00	1.25	100	10°	140°	N
6	1.00	1.00	0°	0°	0°	Y
7	1.00	0.75	10°	−10°	160°	Y
8	1.50	1.25	0°	0°	200°	Y
9	1.00	1.50	10°	10°	280°	N
A	1.00	1.25	10°	10°	40°	N
B	incorrectly matched pattern of label A					

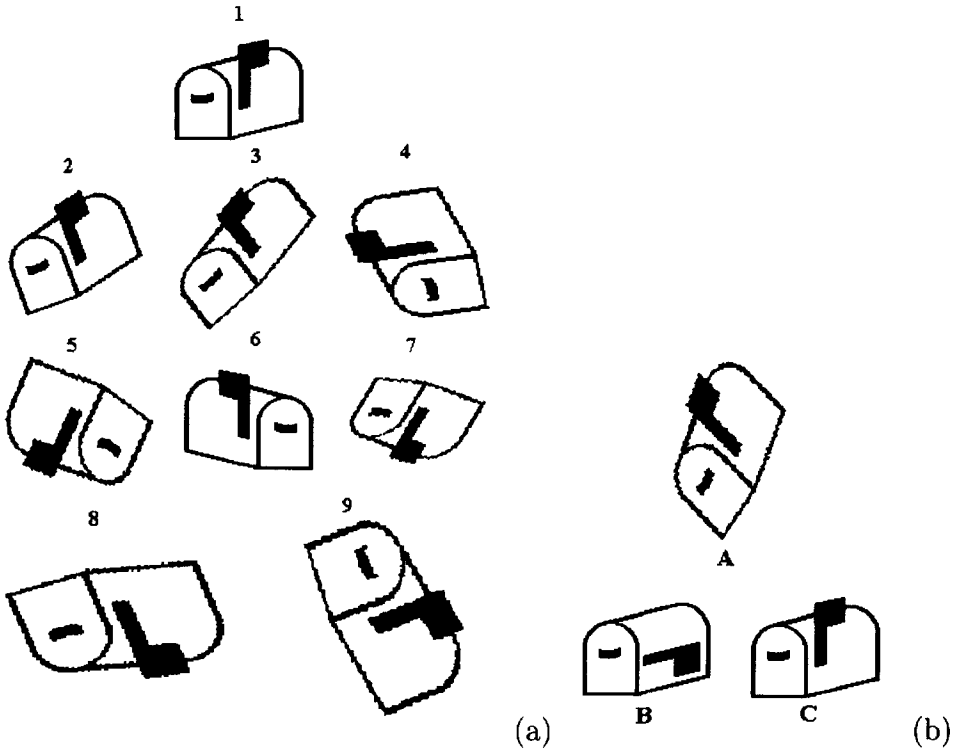

Fig. 20. (a) Some examples of correct recognition. (b) An example of incorrect recognition where A is incorrectly matched to B instead of C.

Table 3
An example of rotation angle estimate

Rotation angle	0.0°	20.0°	40.0°	60.0°	80.0°	100.0°	120.0°	140.0°	160.0°
	180.0°	200.0°	220.0°	240.0°	260.0°	280.0°	300.0°	320.0°	340.0°
Estimated angle	0.00°	0.00°	45.00°	67.50°	78.75°	101.25°	123.75°	146.25°	157.50°
	180.00°	180.00°	225.00°	247.50°	258.75°	281.25°	303.75°	326.25°	337.50°

Fig. 20(b). The parameters of the patterns in Fig. 20 are shown in Table 2. It is hard for the matching process to distinguish patterns A and B in Fig. 20(b), because they are similar in areas and projections. In addition, the shapes are similar. We also use only 16 projections and 16 projection cells, and the rotation angle is not a multiple of angle steps.

We also show an example for estimating rotation angle and scale. The rotation angle estimate is shown in Table 3. The input image is in our pattern database at the first row and the tenth column. We can see that when the rotation angle is equal to 20° and 200° our algorithm performs poorly. The circular projections on initial angles 0° and 20° are similar and difficult to distinguish. We also show some examples of scale estimates in Table 4. The rotation angle estimate will affect the scale estimate. The better the rotation angle estimates, the better the scale estimates.

Table 5 shows our experimental results on the first 29 patterns in the pattern database. The output result of recognition is shown in three columns: No. of images, Correct, Incorrect. The first column, No. of images, is the number of test images. The second column, Correct, and third column, Incorrect, are the numbers and percentages of correct and incorrect matches.

In our experiments, we need 16 × 16 (the circular projections) +4 × 16 (the areas) +2 (the scales) bytes =

Table 4
Some examples of scale estimate

	Rotation angle		Rotation angle		Rotation angle		Rotation angle	
	Scale x	Scale y	Scale x	Scale y	Scale x	Scale y	Scale x	Scale y
Input	0.00°		140.00°		220.00°		20.00°	
	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.25
Estimate	0.00°		135.00°		225.00°		23.50°	
	1.00	1.00	1.10	1.11	1.09	1.12	1.12	1.26

Table 5
Experimental results

Scale		Shear		Rotation angle θ	Mirror	No. of images	Correct	Incorrect
x	y	x	y					
no noise								
1.00	1.00	0°	0°	0°, 20°, 40°, ..., 340°	N	29	506 (96.93%)	16 (3.07%)
no noise								
1.25	1.00	0°	0°	0°, 20°, 40°, ..., 340°	N	29	499 (95.59%)	23 (4.41%)
no noise								
1.00	1.25	10°	− 10°	0°, 20°, 40°, ..., 340°	N	29	473 (90.61%)	49 (9.39%)
no noise								
1.50	1.25	0°	10°	0°, 20°, 40°, ..., 340°	N	29	459 (87.93%)	63 (12.07%)
no noise								
0.75	1.00	5°	5°	0°, 20°, 40°, ..., 340°	Y	29	443 (84.87%)	79 (15.13%)
noise of square of 5% pattern area added								
1.00	1.00	0°	0°	0°, 20°, 40°, ..., 340°	N	29	466 (89.27%)	56 (10.73%)

322 bytes to store a pattern datum in a pattern database, and do not need to store the image of the pattern. On average we need about 36 s to recognize an input pattern on a 486/DX2-80 PC. In the sixth row of Table 5, we randomly add a square of 5% pattern area in the frame of pattern size to be the noise. The square noise may or may not change the patterns because the square is located randomly. In the sixth row, 27 patterns are changed, and two patterns are unchanged because they by chance overlap the noise square. If we take projections at 32 angles, we can get about 2–3% better accuracy and need about 57 s for recognition.

5. Conclusion

We present a method of pattern recognition and also estimate the parameters of the distorted pattern. We can estimate the rotation angle, scales in the x and the y directions, and mirror symmetry. We can also tolerate some noise on patterns. The number of projections and the number of projection cells will affect parameter estimate and recognition. When we add the number of projections and the number of projection cells, we can recognize more patterns and get a higher accuracy in estimating rotation angle and scale, but the matching process is more time consuming. On the other hand, we can rotate the input pattern by an initial angle that is larger than 0° and less than the angle step divided by 2, $\phi/2$, and we can also get a high accuracy of estimating the rotation angle. If we get a bad angle estimate, it will be difficult to obtain a scale estimate.

Acknowledgements

This research work was supported by National Science Council of Taiwan, ROC, under Grants NSC 85-2212-E-002-077 and NSC 86-2212-E-002-025, by Mechanical Industry Research Laboratories, Industrial Technology Research Institute under Grant MIRL 863K67BN2, by EeR-ise Corporation, ACME Systems, Mosel Vitelic and Foxconn Inc.

References

[1] D. Casasent, D. Psaltis, Position, rotation, and scale invariant optical correlation, Applied Optics 15 (1976) 1795–1799.
[2] S.K. Chang, The reconstruction of binary patterns from their projections, Communications of the Association for Computing Machinery 4 (1971) 21–25.
[3] P.E. Danielsson, M. Hammerin, High-accuracy rotation of images, Computer Vision, Graphics, and Image Processing 54 (1992) 340–344.
[4] D.E. Dudgeon and R.M. Mersereau, Multidimensional Digital Signal Processing, Prentice Hall, Englewood Cliffs, NJ, 1984.
[5] J. Flusser, T. Suk, Pattern recognition by affine moment invariants, Pattern Recognition 26 (1993) 167–174.
[6] W.C. Karl, G.C. Verghese, A.S. Willsky, Reconstructing ellipsoids from projections, Computer Vision, Graphics, and Image Processing 56 (1994) 124–139.
[7] Y. Lamdan and H. Wolfson, Geometric hashing: a general and efficient model-based recognition scheme, Proceedings of International Conference on Computer Vision, London, 1988, pp. 238–249.
[8] A.M. Landraud, Image restoration and enhancement of characters, using convex projection methods, Computer Vision, Graphics, and Image Processing 53 (1991) 85–92.
[9] H.B. Liu and C.S. Fuh, Pattern recognition using projection, Proceedings of International Conference on Control, Automation, Robotics, and Vision, Singapore, 1996, pp. 63–67.
[10] A.J. McCollum, C.C. Bowman, P.A. Daniels, B.G. Batchelor, A histogram modification unit for real-time image enhancement, Computer Vision, Graphics, and Image Processing 12 (1988) 337–398.
[11] T.S. Shepherd, W. Uttal, S. Dayanand, R. Lovell, A method for shift, rotation, and scale invariant pattern recognition using the form and arrangement of pattern specific features, Pattern Recognition 25 (1992) 343–355.
[12] L. Spirkovska, M.B. Reid, Robust position, scale, and rotation invariant object recognition using higher-order neural networks, Pattern Recognition 25 (1992) 975–985.
[13] F. Stein, G. Medioni, Structural indexing: efficient 3-D object recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1992) 125–145.
[14] Y. Wang, W. Lu, Multiobjective optimization approach to image reconstruction from projections, Signal Processing 26 (1992) 17–26.