Double Line Image Rotation

Amir Hossein Ashtari, *Member, IEEE*, Md Jan Nordin, *Member, IEEE*, and Seyed Mostafa Mousavi Kahaki, *Member, IEEE*

***Abstract*— This paper proposes a fast algorithm for rotating images while preserving their quality. The new approach rotates images based on vertical or horizontal lines in the original image and their rotated equation in the target image. The proposed method is a one-pass method that determines a based-line equation in the target image and extracts all corresponding pixels on the base-line. Floating-point multiplications are performed to calculate the base-line in the target image, and other line coordinates are calculated using integer addition or subtraction and logical justifications from the base-line pixel coordinates in the target image. To avoid a heterogeneous distance between rotated pixels in the target image, each line rotates to two adjacent lines. The proposed method yields good performance in terms of speed and quality according to the results of an analysis of the computation speed and accuracy.**

***Index Terms*— Double-line rotation, DLR, image representation, image rotation, image transform, line rotation.**

I. Introduction

IMAGE rotation is considered a fundamental component of many computer vision applications and machine vision systems; thus, its speed and accuracy are important in most cases [1], [2]. Highly accurate image rotation is essential for certain image processing tasks such as feature extraction and matching. Furthermore, in many applications, especially real-time systems, a high speed of rotation is important [1], [3]. Hence, it is necessary and advantageous to define a method that provides both high accuracy and fast execution. Image rotation can be described as the movement of an image around a fixed point [2]. The best description of the rotation is based on polar coordinates, as shown in (1) [4].

$$\mathrm{T}(\mathrm{r},\theta) \rightarrow \mathrm{R}(\mathrm{r},\theta+\alpha), \tag{1}$$

where r is the radial coordinate, θ is the angular coordinate, and α is the arbitrary angle of rotation. The rotation can also be defined in Cartesian coordinates. An original input image T(x, y) is supposed to rotate into a target image R(x, y) with an angle of α. A pixel t with coordinates (x_t, y_t) in the image T(x, y) is rotated to position (x_r, y_r) in R(x, y). The basic rotation transformation has been expressed in (2). The direct determination of the rotated points is a one-pass method.

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \tag{2}$$

Determining the rotated point directly using a single function such as (2) is a one-pass technique, whereas calculating the rotated point using multiple functions such as (3), (4), and (5) is a multi-pass technique. Compared with multi-pass methods, one-pass methods are simple and usually more accurate [2], [5]. However, although one-pass methods are accurate, they are still inefficient because multiple high-level calculations and computations are required. As presented in (2), four floating-point multiplications and two floating-point additions are required to calculate each rotated pixel. For an image of many pixels, the one-pass method can actually take longer than multi-pass methods [2], [5].

After performing the primary operation for finding all rotated pixels using (2), an interpolation or resampling function must be used to assign values to the rotated pixels and to determine the best fit for the rotated value from (2). The image pixels use the grid coordinates, which are integers; therefore, floating-point values in (2) should be assigned to the grid coordinates using interpolation. Many interpolation methods have been applied to determine the coordinates of the rotated pixels, including bicubic, bilinear, midpoint, convolution-based, and nearest neighbor interpolation [6]–[9].

The basic transformation matrix (2) can be decomposed into several sub transformations [2], [10], thereby resulting in the development of multi-pass methods for image rotation. To obtain a rapid rotation, the sub-transformations should be changed to skewing transformations, which act on each coordinate separately and can thus be quickly performed as row-by-row or column-by-column parallel shifts. Two-pass and three-pass methods, which are based on different derivations of the basic rotation transformation, have been developed. Because flips are easy to implement visually, the process of rotation is reduced to two consecutive flips [4]. A rotation by $2(\beta-\alpha)$ has been accomplished, as shown in (3).

$$\begin{aligned} \theta_0 &= \theta_0 + 2(\alpha-\theta_0) = -\theta_0 + 2\alpha, \\ \theta_1 &= \theta_1 + 2(\beta-\theta_1) = \theta_0 + 2(\beta-\alpha). \end{aligned} \tag{3}$$

The initial two-pass method in Cartesian coordinates is presented in (4).

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} \cos\alpha & 0 \\ \sin\alpha & 1 \end{pmatrix} \begin{pmatrix} 1 & -\tan\alpha \\ 0 & 1/\cos\alpha \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \tag{4}$$

Manuscript received November 6, 2014; revised February 21, 2015 and May 8, 2015; accepted May 16, 2015. Date of publication June 3, 2015; date of current version June 25, 2015. This work was supported by the National University of Malaysia, Bangi, Malaysia under Grant DIP-2014-18 and Grant FRGS/1/2014/ICT07/UKM/02/2. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Vladimir Stankovic.

The authors are with the Center for Artificial Intelligence Technology, Pattern Recognition Research Group, National University of Malaysia, Bangi 43600, Malaysia (e-mail: ashtari@ieee.org; jan@ukm.edu.my; kahaki@siswa.ukm.edu.my).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2440763

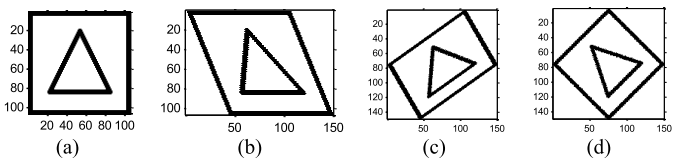


Fig. 1. Image rotation by $\alpha = \frac{\pi}{4}$ rad generated using three shearing processes. (a) Original image. (b) x-shear by $\begin{pmatrix} 1 & -\tan(\alpha/2) \\ 0 & 1 \end{pmatrix}$. (c) y-shear by $\begin{pmatrix} 1 & 0 \\ \sin\alpha & 1 \end{pmatrix}$. (d) x-shear by $\begin{pmatrix} 1 & -\tan(\alpha/2) \\ 0 & 1 \end{pmatrix}$.

Apparently, there is a serious error in this decomposition. Researchers have found that not all of the high-frequency contents of the original image can be preserved by the intermediate image that is yielded by the first sub-transformation [2], [10], [11]. A rotation by α causes a size minimization of $\cos\alpha$ in width and $\sin\alpha$ in length. For a rotation angle of $\frac{\pi}{4}$, the rotated image will be $\sqrt{2}/2$ times smaller than the original image [2]. To avoid information loss in multi-pass methods, resampling and magnification of the original image T(x, y) in the horizontal direction must be performed before performing the vertical skewing, and the magnification factor is $1/\cos\alpha$ [10]. The resulting transformation is presented in (5).

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \tan\alpha & 1 \end{pmatrix} \begin{pmatrix} 1 & -\sin\alpha \\ 0 & 1/\cos\alpha \end{pmatrix} \begin{pmatrix} \cos\alpha & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \quad (5)$$

The "augmented two-pass rotation" method [11] is able to maintain the data and preserve the information; however, this method requires extra processing. Equation (6) corresponds to a two-pass rotation using twice-skewing transformations.

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \tan\alpha & 1+\tan\alpha\times\tan(\alpha/2) \end{pmatrix} \times \begin{pmatrix} 1-\sin\alpha\times\tan(\alpha/2) & -\sin\alpha \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \quad (6)$$

Fig. 1 depicts the process for rotating an image using the three shearing operations presented in (7).

$$\begin{aligned} &(x_0, y_0) \rightarrow (x_0 - \tan(\alpha/2) * y_0, y_0) = (x_1, y_1), \\ &(x_1, y_1) \rightarrow (x_1, y_1 + \sin\alpha * x_1) = (x_2, y_2), \\ &(x_2, y_2) \rightarrow (x_2 - \tan(\alpha/2) * y_2, y_2) = (x_3, y_3). \end{aligned} \quad (7)$$

These three steps can be expressed together to form a rotation. Equation (7) describes a coordinate rotation from (x_0, y_0) in T(x, y) to (x_3, y_3) in R(x, y). The three-shear method can be decomposed from the basic rotation matrix in (2) into three sub transformations in (8) [12], [13].

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \begin{pmatrix} 1 & -\tan(\alpha/2) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \sin\alpha & 1 \end{pmatrix} \times \begin{pmatrix} 1 & -\tan(\alpha/2) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \quad (8)$$

Thus, as indicated in Fig. 1, the image is first sheared horizontally by $-\tan(\alpha/2)$. Next, it is sheared vertically by $\sin(\alpha)$. Finally, it is sheared again along the horizontal direction by $-\tan(\alpha/2)$.

The three-pass method consists of three simple skewing matrixes in which the changing coordinates can be incrementally acquired. Fewer floating-point multiplications are required compared to the two-pass methods, but an extra pass is introduced [2], [5].

Several methods for image rotation have been proposed. All such methods can be classified into two categories: one-pass and multi-pass methods. Our proposed one-pass method, which is called double-line rotation (DLR), applies a direct rotation point with linear estimation onto the target image. The proposed method uses columns or rows of the original image as lines for rotation and finds the corresponding line's pixel coordinates in the target image. Applying the proposed DLR technique, other interpolation techniques in the rotation phase, such as midpoint [2], cubic [6], [9], and convolution-based interpolation [3], are no longer necessary; thus, the response time is reduced.

II. Related Works

A high-speed one-pass algorithm for image rotation was proposed by Xuede *et al.* [2]. The algorithm considers that an m × n image T consists of m line images. Thus, the rotation of an image is simply the rotation of all of the line images. The distinguished midpoint technique for line generation and interpolation is utilized to accurately and rapidly perform the rotation of each line image. The algorithm rotates m images instead of one image and merges them to achieve the final rotated result. The method uses a midpoint algorithm as an interpolation technique to determine the best fit for the pixels in the rotated image. In their proposed method, the input images must be pre-rotated by a special angle (i.e., 90°, 180°, or 270°) to limit the absolute angles of rotation to 45°. However, the method produces unassigned pixels in the rotated image after merging all of the rotated lines and therefore necessitates post-processing to fill the holes. Xuede *et al.* [2] suggested a technique for general rotation by proposing only a base pixel rotated by a single calculation. The rotated x and y coordinates for the neighboring pixels are obtained by adding $\cos\alpha$ and $\sin\alpha$ to the x and y coordinates, respectively. Consequently, an image is rotated more rapidly than it is with the basic one-pass methods in (2) because the method does not require additional treatment other than two additions of decimal fractions and rounding operations. However, it is important to note that large coordinate errors may be accumulated through continuous additions.

In a large coordinate system, a resampling function is usually necessary to assign values to unallocated pixels and to determine the best fit for rotated values when they are floats. Pixels in images use the grid coordinates, and the grid coordinates are integers. Therefore, the floating-point numbers from the basic rotation transform (2) should be assigned to grid coordinates using resampling functions and interpolation techniques. Usually, rotated pixels cannot be easily assigned using only two additions of decimal fractions and rounding operations in large coordinate systems because of pixel overwriting or unallocated pixels in the rotated image. Many interpolation methods and resampling functions have been used to determine the proper coordinates of rotated pixels

and to assign the best value to unallocated pixels; examples include bicubic, bilinear, midpoint, convolution-based, and nearest neighbor interpolation.

Various improvements, such as those presented in [2], [3], [14], and [15], on the computing efficiency of rotation methods have been made. Such proposed methods usually improve the interpolation methods to find the best fit for the rotated pixels. Yu *et al.* [14] proposed a new algorithm for interpolation that can rotate an image by directly moving pixels according to the skewed angle and simultaneously avoid floating-point computations. Consistent resampling theory (CRT) is used for interpolation in three-shearing rotation and resizing to improve the quality of the rotated/resized image [15].

One-pass methods are simple methods, and they provide low-speed rotation, primarily because they require four floating-point multiplications and two floating-point additions for each pixel [2]. This requirement is considered a flaw of the one-pass method and consequently has become one of the motivations for developing multi-pass methods. Despite accelerating the rotation, the accuracy of multi-pass methods is low because of the introduction of frequency aliasing during the rotation [16] and because extra processing is necessary to reduce or avoid additional aliasing [17]. Furthermore, multi-pass methods require more temporary memory for the rotation. Several improvements of three-shear methods have been made to reduce the size of the memory and buffer for rotation. Multi-pass methods have been widely investigated in [5], [15], and [18]. Such methods have also been extended to 3D rotations of image volumes [19].

Several methods of reducing the memory usage through region or array rotation rather than point rotation have been proposed. However, most studies have been restricted to special cases of rotation or special tasks [20], [21] or have used parallel processing to obtain high speeds [22]–[24]. Kermisch [20] has successfully patented a method for rotating digital images that minimizes the necessary number of disk accesses. This method is practical in cases for which the image is large but the available computer memory is limited. Subsequently, Paeth [13] proposed a raster rotation using shearing lines. Chien and Baek [25], [26] proposed a fast black run rotation algorithm for binary images. They could accelerate the rotation by more than twice compared with other rotation methods and successfully utilized the line breakdown method to facilitate the elimination of holes that exist in the rotated plane. After the rotation in the rotated image, there are certain pixels that have no values allocated to them. Generally, these pixels are called holes or unallocated pixels in the rotated image. To eliminate holes, post-processing is necessary. Scratching image pixels, resampling, or interpolation using methods such as nearest-neighbor, bilinear, cubic, or bicubic interpolation during the rotation are used to remove the holes. The algorithm proposed by Chien and Baek [25], [26] works by extracting black lines from a binary image. The algorithm rotates the starting and ending points and then draws a line between the two rotated points.

Local interpolation is an imprecise step and obliterates the global information of the image. When performing shears, interpolations to new grid points must be performed as a discrete image is defined on a grid. The interpolation used in resizing avoids losses of information about the image; however, this process affects the image quality [1], [19]. Despite these drawbacks, two- and three-dimensional image rotation using the fast Fourier transform (FFT) and three shears yields accurate results [3], [4], [27]–[30]. The basic concept of the Fourier method for rotation is that three-shear processes are applied to the original image, and each shear process is implemented using the FFT [30]. Park *et al.* [1] demonstrated that Hermite functions provide a better representation and are suitable for rotation compared with FFT techniques, and the authors elaborated on how these functions can be utilized for image representation.

Various works related to the Radon and Trace transforms have used projections and a polar coordinate system for image rotation [31], [32]. The main objective of these approaches is to maintain the original pixel intensities by downscaling the rotated image. Usually, the rotation is conducted without interpolation. The method presented in [31] and [32] produces exact digital image rotations by re-ordering the elements in the discrete sinogram of projections defined by the finite Radon transform (FRT), but it has a high computational cost. All original pixel values are accurately preserved by synchronized scaling and rotation of image pixels.

III. DLR Method

One-pass methods are not sufficiently rapid, but they provide a high-quality rotated image [2], [5]. In this paper, a new one-pass method for realizing the high-speed rotation of images while preserving the image quality is proposed. The new rotation method, which is based on line views of the image and their rotation, employs the line coordinates in the target image. It finds the base-line equation in the target image and extracts all pixels that correspond to the base-line. Extracting the pixels that correspond to the base-line is performed only once. The base-line equation is assumed to be an injective function; therefore, $\forall x \in \mathbb{Z}^{\geq}$, there is only one f (x) value. By rounding f(x) to the nearest number, the corresponding pixel on the base-line is reached. All extracted coordinates from the base-line are applied to calculate other adjacent line coordinates. The corresponding pixels in other lines of the target image are calculated by addition or subtraction from the base-line pixel coordinates. Consequently, the proposed algorithm calculates the line equation only once and applies its coordinates to calculate all of the image lines' corresponding pixels in the target image. The proposed method exactly preserves the original pixel values and intensities.

A. Base-Line Calculation

In the proposed method, a base-line equation in the target image with an arbitrary slope of α is calculated. Then, all columns or rows from the original image, depending on α, are transformed to the corresponding coordinates in the target image. As shown in Fig. 2(a), for a rotation, all pixels in line ω at an angle of 0° transfer to the corresponding line ω_{r} at an angle of α. $\alpha \in [0, 2\pi)$ is segmented into eight areas and two groups, including group H and group V.

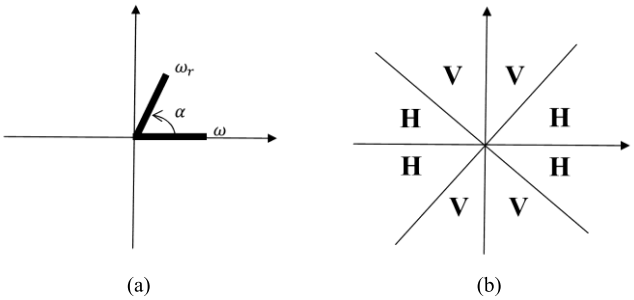


Fig. 2. Basic concept of line rotation. (a) Rotate a line by α. (b) Vertical and horizontal zones.

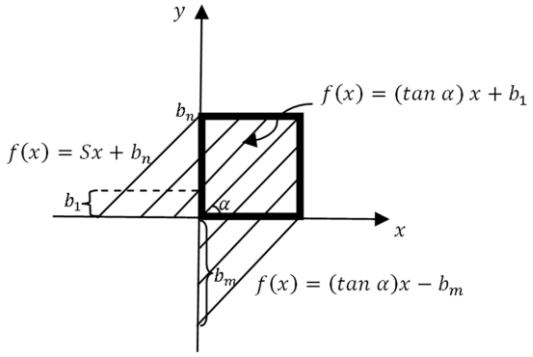


Fig. 3. Using the base-line equation to calculate all available transform lines in the target image.

In the H areas, horizontal lines from the original image are used for rotation; in the V zones, vertical lines from the original image are employed. The m × n image T has m lines in V zones and n lines in H zones.

The H zones are $\alpha = \left[-\frac{\pi}{4}, \frac{\pi}{4}\right] \cup \left[\frac{3\pi}{4}, \frac{5\pi}{4}\right]$ (Fig. 2b), and the V zones include vertical lines in the image, where $\alpha = \left|\frac{\pi}{4}, \frac{3\pi}{4}\right| \cup \left|\frac{5\pi}{4}, \frac{7\pi}{4}\right|$.

The original image T is assumed to have size m × n. m is the number of rows on the y-axis, and n is the number of columns on the x-axis. In this paper, image R is assumed to be the target, or rotated, image. The base-line equation is $f(x) = Sx$, where S is the line gradient. S is calculated via the angle of rotation, which is $\tan\alpha$. All available lines with the same slope can be extracted using a defined $b \in \mathbb{Z}$, $b =]-\infty, +\infty]$, which is added to the base-line equation to extract all available lines with the same slope in image R by $f(x) = Sx + b$. Fig. 3 shows that, using b, all available lines in the m × n image T can be allocated to lines in image R with angles of α. The movement of b depends on the image size, and α can vary. The maximum movement of b equals $\sqrt{m^2 + n^2}$, where $\alpha \in [\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4}, \frac{7\pi}{4}]$, and the minimum movement is min(m, n), where $\alpha \in [0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}, 2\pi]$.

Using the base-line and b to transform all lines from the original image to the target image causes the pixel Euclidian distance problem shown in Fig. 4(b). The distance between diagonal adjacent pixels in the target image is $\sqrt{2}$ for the case of $\alpha = \frac{\pi}{4}$ rad (instead of 1, which is the distance in the original image), but the distance between two perpendicular adjacent pixels in the target image is equal to 1 (and 1 is the corresponding difference in the original image). In the same manner, the distance between the first line and the third line in the target image is equal to $\sqrt{2}$ (instead of 2 in the original image). Heterogeneous distances between rotated pixels in the target image cause an affine image, which is shown in Fig. 4(c). To avoid the affine transform, all pixel distances in the target image must be the same multiple of the original image's corresponding distance.

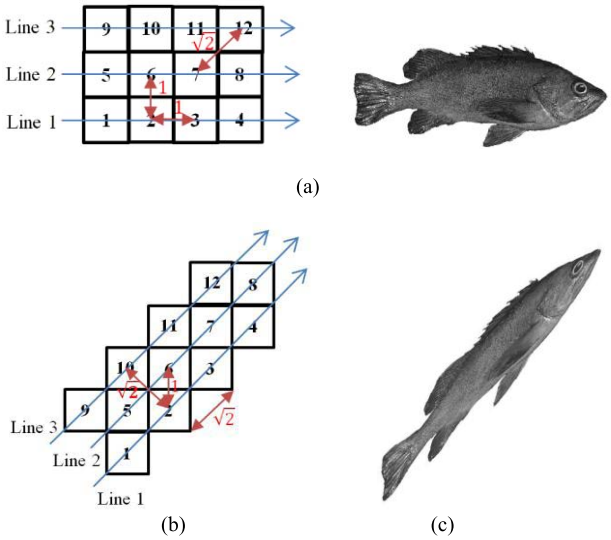


Fig. 4. Affine effect in the rotated image. (a) Original images. (b) Transformed adjacent lines causing incorrect distances between pixels in the target image. (c) Image after the transform without calculating the starting point for each line for the rotation of $\frac{\pi}{4}$ rad.

To achieve a homogenous distance between pixels, the starting points for each line in the target frame are employed. If the starting points for each line are not used, the transformation is performed with the affine transform. The starting points are calculated as the corresponding pixels on a line for which the equation is specified by the $\left|\alpha \pm \frac{\pi}{2}\right|$ gradient. The start-line is perpendicular to the base-line, and the first pixel of each line in the target frame must be on the start-line. An example of a start-line is presented in Fig. 5(a) as $f_s(x)$. Fig. 4(c) shows a rotation by $\frac{\pi}{4}$ rad in which the starting points have not been used. Without starting points, the resultant image contains affine effects along the y- or x-axis, depending on α and its zone.

Forming homogenous distances between adjacent pixels in the target image indicates a difference between the original pixel distances and the rotated pixel distances. The rotated image is enlarged by a maximum factor of $\frac{\sqrt{2}}{2}$ (in the case of $\alpha = \frac{\pi}{4}$ rad) because the adjacent pixel distance changes from 1 to $\sqrt{2}$ (when $\alpha = \frac{\pi}{4}$ rad). The DLR method preserves all of the original pixel values by synchronized scaling of the original image aspect. Retaining the fixed-order contour of the original shape during rotation magnifies the rotated image, but resizing (minimizing) the rotated image is not necessary unless the enlargement, aliasing of the edges of the rotated image and the expense of memory or storage are not optimal for certain special tasks in image processing and computer vision. Fig. 5(b) shows the homogenous distances in the target frame obtained using the start-line. In the original image,

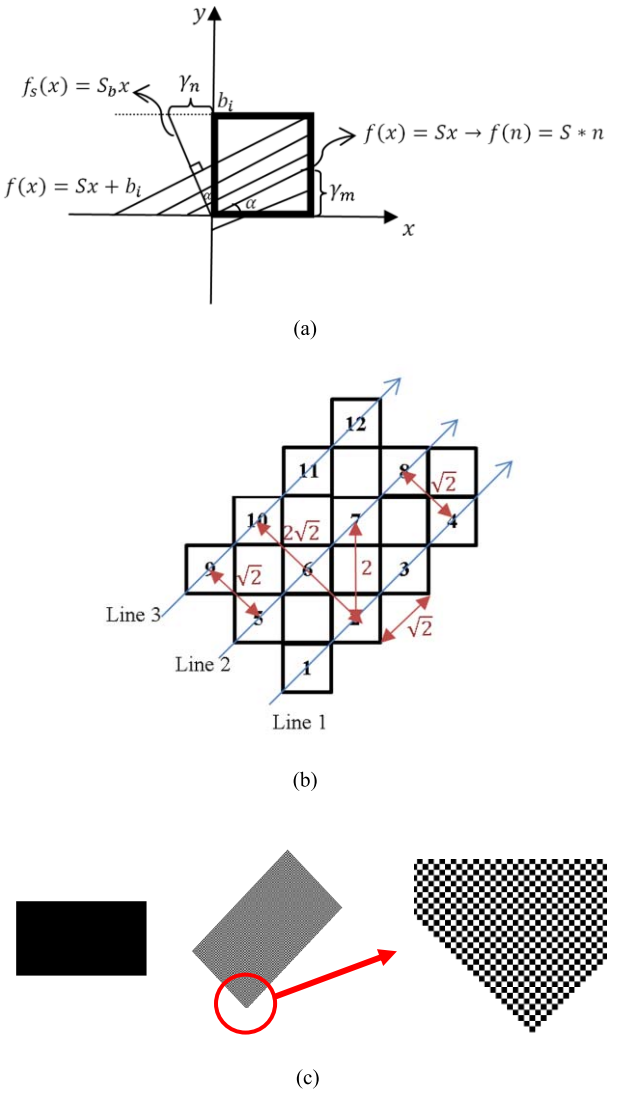


Fig. 5. Applying the start-line in the rotation transform. (a) The start-line calculation and the size of the target image where $\alpha \in [0, \frac{\pi}{4}]$. (b) The target image adjacent pixel distance after applying the start-line. (c) A sample of rotating a black rectangle to show unallocated pixels with line rotation for $\alpha = \frac{\pi}{4}$.

the Euclidian distance between pixels 2 and 7 is $\sqrt{2}$; after the rotation by $\alpha = \frac{\pi}{4}$ rad, as shown in Fig. 5(b), the distance is multiplied by $\sqrt{2}$ such that the distance is enlarged by a factor of $\frac{\sqrt{2}}{2}$ in the target image. The magnification of the proposed method is equal to $|\sin(\alpha)|$ for vertical zones and $|\cos(\alpha)|$ for horizontal zones (where α is the angle of rotation) and ranges between 0 and $\frac{\sqrt{2}}{2}$.

Using the base-line equation for rotation, some unallocated pixels occur, as Figs. 5(b) and 5(c) show for $\alpha = \frac{\pi}{4}$; these pixels must be addressed. The unallocated pixels can be eliminated by interpolation algorithms (such as linear, nearest neighbor, bicubic, or midpoint interpolation), which reduce the speed of rotation.

B. Unallocated Pixel-Filling Strategy

Unallocated pixels can be filled using various image processing filters such as mean and median filters. Using a filter requires post-processing to fill the holes and allocate

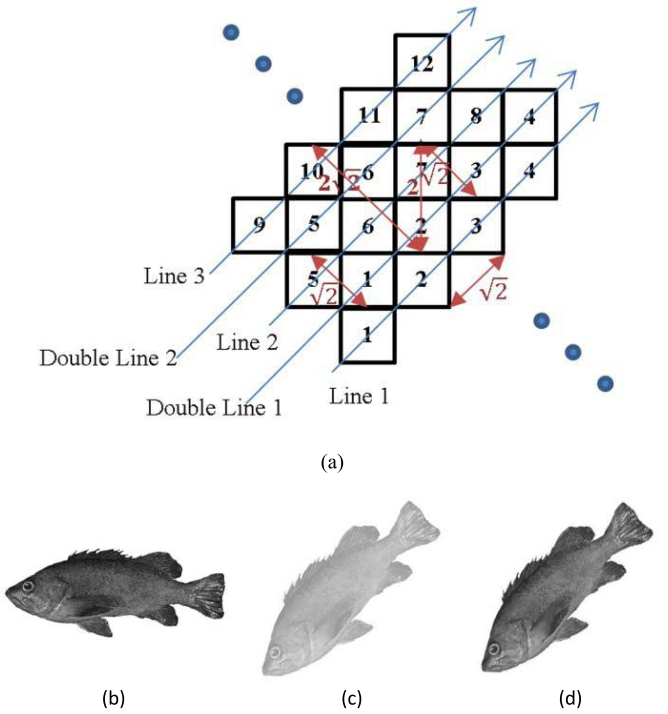


Fig. 6. DLR. (a) Distance between pixels in the target image after DLR is performed. (b) Original image. (c) Line rotation of $\alpha = \frac{\pi}{4}$. (d) DLR of $\alpha = \frac{\pi}{4}$.

the proper value for unknown pixels according to the adjacent pixel values, which decreases the speed of the method. Moreover, maintaining the original intensity or color in the target image is important for certain pattern recognition and computer vision tasks. To guarantee high-speed operation, a new approach for solving the unallocated pixel problem without interpolation techniques or filters is proposed. The proposed technique is applied during the rotation; thus, post-processing is no longer required unless resizing to the actual image size upon request. The results of the method indicate that the quality of the proposed strategy is as good as the median filter at maintaining the original intensity of the pixels.

The proposed method uses duplication in which each line in the original image is transferred to two adjacent lines in the target image. The second line corresponds to the same x value but a different $f(x)$. For example, for $x_n \in T$, there are two lines, $f(x)$ and $f(x)+1$. $f(x)+1$ would be overwritten when $f(x_{n+1})$, $x_{n+1} \in T$ has the same coordinates. The second line (the so-called double-line) has less priority than the main line; therefore, the main-line coordinates overcome the double-line coordinates in terms of overlapping. The results of DLR are shown in Fig. 6. In the vertical zones, vertical lines are duplicated, while in the horizontal zones, horizontal lines are used. In the horizontal zones, the image quality assessment results do not show any differences between duplicating up-lines to below-lines and duplicating below-lines to up-lines. Similarly, in the vertical zones, duplicating left-lines to right-lines does not change the quality of the resulting image compared to duplicating right-lines to left-lines.

After rotation, the distance ratio between pixels in the original and rotated image must be the same. For example,

if the distance between two points $A \rightarrow B$ and $B \rightarrow C$ in the original image equals Z, then in the rotated image, the corresponding distance $A' \rightarrow B'$ equals Z', and $B' \rightarrow C'$ equals Z''. For a correct rotation, the distance ratio between the same points in the rotated and original images must be the same such that $Z' = Z'' = \delta Z$, where δ indicates the enlargement or reduction in the distance between two points in the rotated image. If $\delta > 1$, the rotated image is larger than the original image; if $0 < \delta < 1$, the rotated image is smaller than the original image; and if $\delta = 1$, the rotated and original images are the same size. Using DLR, each pixel in the original image is mapped to two pixels in the target image. The distance between the centers of two mapped pixels as a group is calculated rather than using the center a single pixel. Thus, two identical mapped pixels are treated as one, and the center of their rectangle is used to measure the distance. Using the proposed method, the distance ratio between pixels remains constant, as shown in Fig. 6(a). In the worst cases (e.g., $\alpha = \frac{\pi}{4}$), the maximum distance is $\sqrt{2}$ instead of 1; thus, the image is enlarged by a factor of $\frac{\sqrt{2}}{2}$, as mentioned above. The double-line generates the second pixels using the first-line equation in the target image, as shown in Fig. 6. The double-line pixels have less priority than the first-line pixels; therefore, the first-line pixels overcome the double-line pixels in terms of overlapping.

The image enlargement by the proposed method makes heuristic sense for logical rotations, for which all boundary pixels remain in the same position and number. Fig. 7 shows the rotation of a 3×5 rectangle that includes 15 pixels by different methods based on a basic rotation function, where $\alpha = \pi/4$. All methods vary in terms of the number and order of contour pixels due to their resampling/resizing functions. However, unallocated pixels do not appear in three-shear methods, but the contour pixels of the shape do not remain with their order, as shown in Fig. 7(b). The bilinear and bicubic interpolation methods generate more logical results, but they change the order of contours and the values of the pixels. The proposed algorithm changes the number of pixels, but the shape contour and pixel values remain fixed. The presented algorithm generates a result that is similar to that of the bilinear interpolation method but preserves the intensity of the original pixels. Retained fixed-size contours and pixel intensities are useful measures for some tasks in computer vision and image processing applications such as invariant feature extraction. Fig. 7(g) illustrates the concept of retaining fixed-size contours in a grid view. Hence, ten pixels are the boundary pixels; their order remains identical to the original order when the shape is rotated. DLR yields the result without any changes in the pixel values, and the contour grid size of the shapes remains the same in the rotated image.

C. Calculation of the Base-Line, Start-Line and Target Image Size

The first step for rotation is the calculation of the target image size. Each area uses a different equation to determine the size of the target image. For ease of understanding and calculation, eight zones, $\text{Zone}\,1 = \left[0, \frac{\pi}{4}\right]$, $\text{Zone}\,2 = \left]\frac{\pi}{4}, \frac{\pi}{2}\right]$,

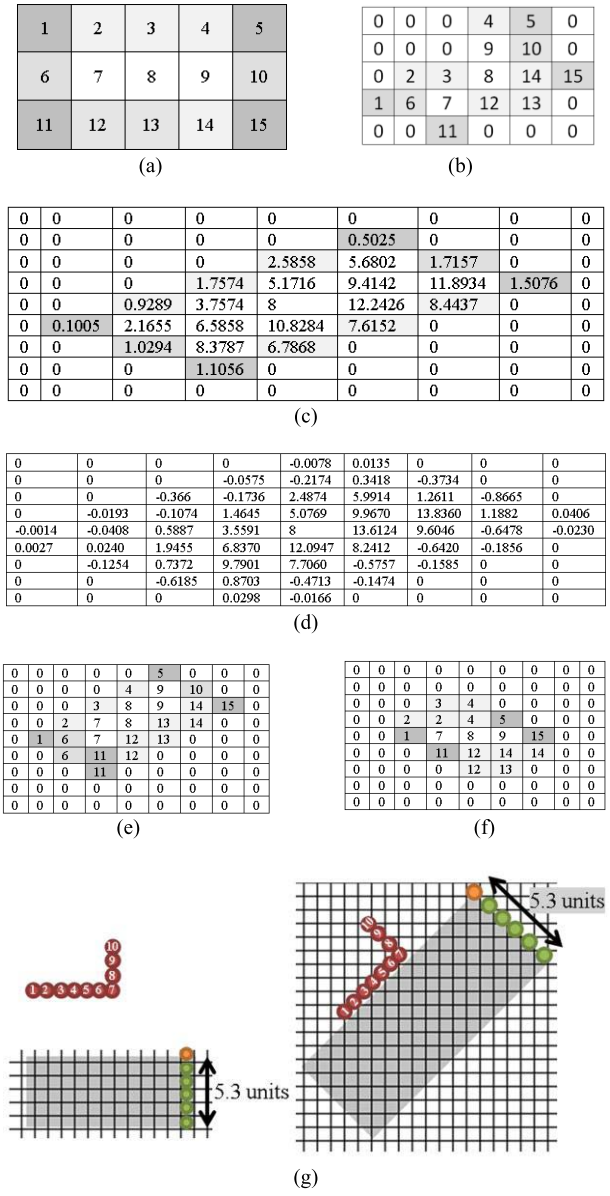


Fig. 7. Rotation of a rectangle by $\alpha = \frac{\pi}{4}$ rad. (a) Original rectangle. (b) Rectangle rotated by the three-shear method and nearest neighbor interpolation without resampling. (c) Rectangle rotated using the basic function and bilinear interpolation (the number of pixels increases to 23, and the values of all pixels change). (d) Rectangle rotated using the basic function and bicubic interpolation (the number of pixels increases to 49, and the values of all pixels change). (e) Rectangle rotated using the double-line algorithm (the number of pixels increases to 23, but the values of the pixels remain fixed). (f) Rectangle rotated by the basic function and nearest-neighbor interpolation (the number of pixels changes to 17, but two pixels, 6 and 10, are missed). (g) The rotated and original image have the same contour size in a grid computation by DLR for $\alpha = \frac{\pi}{4}$ rad.

$\text{Zone}\,3 = \left]\frac{\pi}{2}, \frac{3\pi}{4}\right]$, $\text{Zone}\,4 = \left[\frac{3\pi}{4}, \pi\right]$, $\text{Zone}\,5 = \left]\pi, \frac{5\pi}{4}\right]$, $\text{Zone}\,6 = \left]\frac{5\pi}{4}, \frac{3\pi}{2}\right]$, $\text{Zone}\,7 = \left]\frac{3\pi}{2}, \frac{7\pi}{4}\right]$, and $\text{Zone}\,8 = \left[\frac{7\pi}{4}, 2\pi\right]$, are defined, as shown in Fig. 8. The second step is the calculation of the starting points. All of the lines in the rotated image begin from line ω_s in $f_s(x) = S_b x$ and $S_b = \tan(\alpha + \frac{\pi}{2})$. S_b is the slope of the start-line that is used to determine the coordinates of the starting points for each line in the target image. Each rotated line must begin from a starting point, and its slope depends on α.

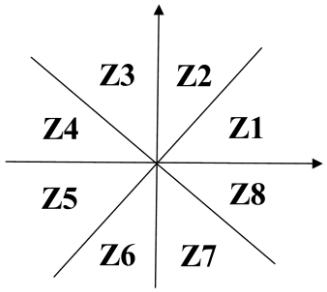


Fig. 8. Segmenting eight zones for DLR.

1) Zone 1: The size of the rotated image is calculated before rotation for an m × n image T and angle of α. The size of the rotated image R is $m_{rt} \times n_{rt}$ and is calculated by (9), where the rotation angle is in zone 1 ($\alpha \in [0, \frac{\pi}{4}]$).

$$m_{rt} = m + \gamma_m, \quad n_{rt} = n + \gamma_n. \tag{9}$$

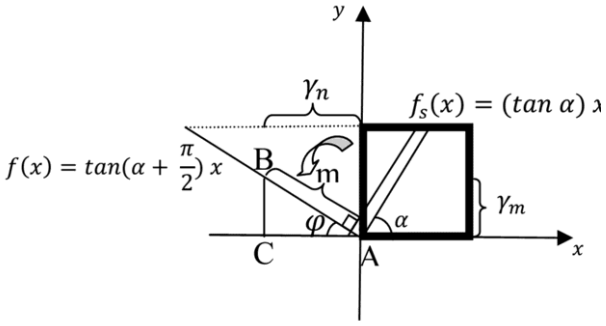
The calculation method and value of γ_m and γ_n are different for each zone. In zone 1, $\gamma_m = f(n)$ and $\gamma_n = \left|\frac{m}{S_b}\right|$. The base-line is calculated by $f(x) = Sx$, where $S = \tan\alpha$. γ_m and γ_n are shown in Fig. 5(a). When α is in zone 1, it is in the H zone, and all horizontal lines are rotated; thus, there are m lines. Surely, the m lines have m starting points. In zone 1, for the starting point with coordinates (x_s, y_s), all y_s are given, and the x_s are unknown; thus, the x coordinate of the starting point is needed. We define $f_s(y) = \frac{y}{S_b}$ $\forall y \epsilon [1, m]$. Thus, line i begins from $f_s(i)$; therefore, the starting point of line i has coordinates $(f_s(i), i)$. Fig. 5(a) denotes the start-line as $f_s(x)$.

The vector values of all members of $f(x)$ and $f_s(x)$ are rounded to the nearest integer number. In the exact half, the rounding algorithm determines the highest number. Therefore, $f(x)$ and $f_s(x) \in \mathbb{Z}$ (where in zone 1, for $f(x)$, $\forall x \epsilon [0, n-1]$, and for $f_s(x)$, $\forall x \epsilon [0, m-1]$). The indices must be positive numbers (absolute values) as they are image-pixel indices. Thus, we define $f(x) \in \mathbb{Z}^{\geq}$ and $f_s(x) \in \mathbb{Z}^{\geq}$. $f(x)$ and $f_s(x)$ are calculated and rounded only once as the base-line and start-line. Other transferring lines follow from the base-line equation by adding or subtracting b from $f(x)$.

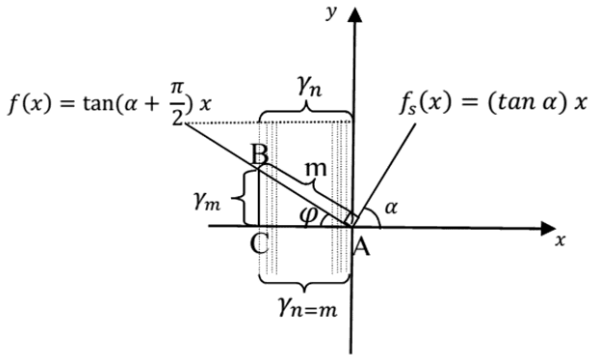
2) Zone 2: In zone 2, all available vertical lines (n lines) in the image transform to the corresponding lines in the target image. The starting points in the rotated lines are calculated using line f_s. In zone 2, $f(y)$ is calculated instead of $f(x)$ because vertical lines are being mapped. There are m points necessary for each line to determine $f(y)$. Thus, the value on the y-axis is given, and the corresponding value on the x-axis is unknown. Therefore, the rotated coordinate for each pixel in line i is $(f(i), i)$, for which $f(i)$ is calculated as $\frac{i}{\tan(\frac{\pi}{2}+\alpha)}$.

The size of the rotated image is calculated using γ_m and γ_n, as shown in Fig. 9(a). In △ABC, m is the hypotenuse, and $\varphi = \pi - \alpha - \frac{\pi}{2}$; thus, $\gamma_n = (\cos\varphi)m$ and $\gamma_m = \sqrt{m^2 - \gamma_n^2}$. In the image T, when the calculation extends to pixels instead of lines, $\gamma_n = m$ and $\gamma_m = f(n)$. Fig. 9(b) shows the grid when pixel m lies on line $f(x) = \tan(\alpha + \frac{\pi}{2})x$. The size of the target image in zone 2 is calculated using (10).

$$m_{rt} = \gamma_m + n, \quad n_{rt} = f(n) + \gamma_n, \tag{10}$$



(a)



(b)

Fig. 9. DLR in zone 2. (a) Calculating the start-line and the base-line for rotation. (b) $\gamma_n = m$, where line $x = 0$ lies on $f(x) = \tan(\alpha + \frac{\pi}{2})x$.

where $f(n)$ in n_{rt} comes from $f(y)$ such that $f(y) = \frac{y}{\tan(\frac{\pi}{2}+\alpha)}$ in Z2. All pixels in the vertical lines map to rotated lines and start from line f_s with the linear equation $f_s(x) = (\tan\alpha)x$. The rotation equation is $f(x) = \tan(\alpha + \frac{\pi}{2})x + b_i$, where b_i corresponds to the vertical line such that $\forall i \in [0, n-1]$. Thus, each line in the m × n image T with equation $x = b_i$, where $b_i \in [0, n-1]$, maps to $f(x) = \tan(\alpha + \frac{\pi}{2})x + b_i$.

3) Zone 3: In zone 3, the slope is negative, but the equations are the same as for zone 2. Vertical lines instead of horizontal lines and $f(y)$ instead of $f(x)$ are used to calculate the base-line's corresponding pixels. The size of the rotated image R ($m_{rt} \times n_{rt}$) is $m_{rt} = \gamma_m + n$, and $n_{rt} = \gamma_n + m$. $\gamma_n = |f_s(n)| \rightarrow f_s(y) = \frac{y}{-\tan\alpha}$, and $\gamma_m = \sqrt{m^2 - \overline{AC}^2}$ in △ABC, where $\overline{AC} = (\cos\varphi)m$. As shown in Fig. 10, in △ABC, m is the hypotenuse, and $\varphi = (\pi - \alpha) - \frac{\pi}{2}$. In the image, the calculation extends to the pixels and grid; thus, $\gamma_m = |f(m)|$. Each pixel in image T has two corresponding values in image R_t, i.e., $\forall x, y \in T, \exists X, Y$, $X_D, Y_D \in R_t$, $X = f(x)$, $Y = \gamma_m + f_s(y)$, $X_D = f(x+1)$, and $Y_D = \gamma_m + f_s(y)$, where X_D and Y_D are the double-line coordinates.

4) Zone 4: In Zone 4, horizontal lines are used for the rotation. The slope of $f_s(x)$ is positive, but $f(x)$ has a negative gradient. Each horizontal line in image T lies in a line with $S = |\tan(\alpha)|$ and shears using the line $f_s(x)$, with $f_s(x) = \left(\tan\alpha + \frac{\pi}{2}\right)x$. The size of the rotated image is $m_{rt} = \gamma_m + m$, and $n_{rt} = \gamma_n + n$, where $\gamma_n = f_s(m) \rightarrow \left\| \frac{m}{\tan(\alpha+\frac{\pi}{2})} \right\|$ and

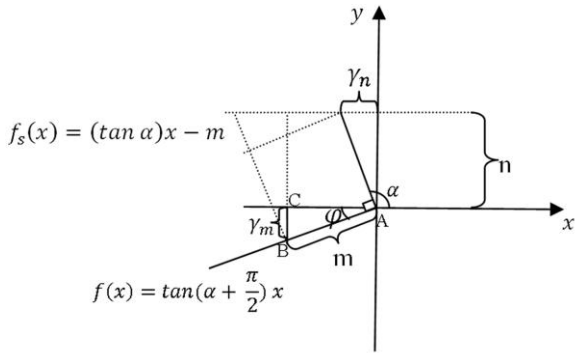


Fig. 10. Calculating the base-line, start-line, and size of the rotated image for the DLR in zone 3.

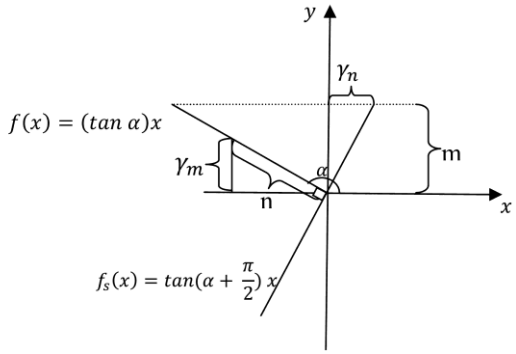


Fig. 11. Calculation of the base-line, start-line, and size of the rotated image for the DLR in zone 4.

$\gamma_{\mathrm{m}} = f(\mathrm{n}) \rightarrow \|\mathrm{n}\tan\alpha\|$. Each horizontal line in the m × n image T with equation $\mathrm{y} = \mathrm{b_i}, \mathrm{b_i} = [0, \mathrm{m}-1]$ is mapped to $f(\mathrm{x}) = \mathrm{Sx} - \mathrm{b_i}$ and shears using $f_{\mathrm{s}}(\mathrm{x}) = \left(\tan\alpha + \frac{\pi}{2}\right)\mathrm{x}$, as presented in Fig. 11.

5) Zone 5: In zone 5, the rotated image can be generated by logical justification performed on the indices. All zones are a transform from Z1 or Z2, depending on whether vertical or horizontal lines are used. Z5 is a transform from Z1 or Z4. The sizes of the rotated image in Z1 and Z5 are the same; thus, $\mathrm{m_{rt}} = \mathrm{m_{Z1}}$ and $\mathrm{n_{rt}} = \mathrm{n_{Z1}}$. The transform $\tau_1|\mathrm{X} = -\mathrm{x}, \mathrm{Y} = -\mathrm{y}$ makes a rotation in Z5 based on the Z1 rotated image. When $\alpha = \left]\pi, \frac{5\pi}{4}\right]$, a rotation of $\alpha - \pi$ in Z1 is performed, and the result transforms by τ_1. Thus, for each line in the m × n image R in Z1, there is a corresponding line in the $\mathrm{m_{rt}} \times \mathrm{n_{rt}}$ image R in Z5 that is transformed by τ_1. The internal transform in each line from Z1 to Z5 is applied by

$$\forall \mathrm{x},\ \mathrm{y} \in \mathrm{L_i},\ \mathrm{i} = [0, \mathrm{m}-1],$$
$$\mathrm{L_i new}(\mathrm{X}, \mathrm{Y}) = \mathrm{L_{(m-i)}}(\mathrm{n_{rt}} - \mathrm{x}, \mathrm{m_{rt}} - \mathrm{y}), \tag{11}$$

where Lnew is the transformed line in Z5 (see Fig. 12).

Generally, all zones that use vertical lines (Z2, Z3, Z6, and Z7) are a transform of each other, and in the same manner, all horizontal zones (Z1, Z4, Z5, and Z8) can be generated by transforms from each other. A direct rotation is faster than an indirect rotation in (11). For a direct rotation in zone 5 without using a transform from zone 1 or zone 4, horizontal lines are used for the rotation. The slope of

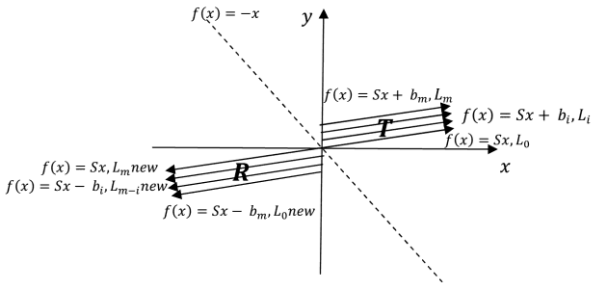


Fig. 12. Transform of lines from zone 1 to zone 5.

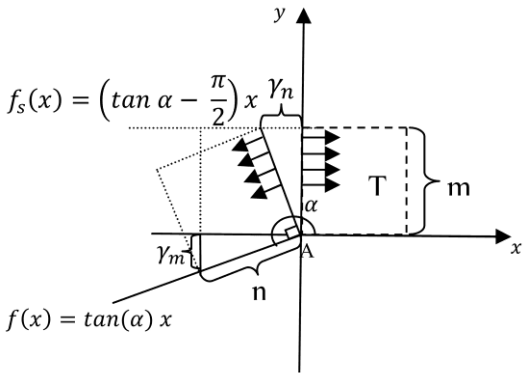


Fig. 13. Direct line rotation in zone 5 using the base-line equation and the start-line.

$f(\mathrm{x})$ is positive, but $f_{\mathrm{s}}(\mathrm{x})$ has a negative gradient. Each horizontal line in image T lies in a line with $\tan(\alpha)$ and shears using $f_{\mathrm{s}}(\mathrm{x}) = \left(\tan\alpha + \frac{\pi}{2}\right)\mathrm{x}$. The dimensions of the rotated image are $\mathrm{m_{rt}} = \gamma_{\mathrm{m}} + \mathrm{m}$ and $\mathrm{n_{rt}} = \gamma_{\mathrm{n}} + \mathrm{n}$, where $\gamma_{\mathrm{n}} = f_{\mathrm{s}}(\mathrm{m}) \rightarrow \left\|\frac{\mathrm{m}}{\tan(\alpha - \frac{\pi}{2})}\right\|$ and $\gamma_{\mathrm{m}} = f(\mathrm{n}) \rightarrow \|\mathrm{n}\tan\alpha\|$. Each horizontal line in the m × n image T with equation $\mathrm{y} = \mathrm{b_i}$ is mapped to $f(\mathrm{x}) = \mathrm{Sx} - \mathrm{b_i},\ \mathrm{b_i} = [0, \mathrm{m}-1]$ and shears using $f_{\mathrm{s}}(\mathrm{x}) = \left(\tan\alpha - \frac{\pi}{2}\right)\mathrm{x}$, as presented in Fig. 13.

6) Zone 6: In zone 6, a rotated image is generated by a transform from either zone 2 or 3. A transform from zone 1 to zone 5 has been previously described, and it is similar to the transform from zone 2 to zone 6. Here, the transform from zone 3 to zone 6 is discussed. The description of the method can be used for a transform from zone 4 to zone 5. The size of a rotated image in Z6 and Z3 is the same; therefore, $\mathrm{m_{rt}} = \mathrm{mZ3_{rt}}$ and $\mathrm{n_{rt}} = \mathrm{nZ3_{rt}}$. The transform $\tau_2|\mathrm{X} = \mathrm{x}, \mathrm{Y} = -\mathrm{y}$ produces a rotation in Z6 of a rotated image in Z3. When $\alpha = \left]\frac{5\pi}{4}, \frac{3\pi}{2}\right]$, a rotation in zone 3 of $\alpha - \frac{3\pi}{4}$ is performed, and the result is transformed by τ_2. Thus, for each line in the m × n image R in zone 3, there is a corresponding line in the $\mathrm{m_{rt}} \times \mathrm{n_{rt}}$ image R in zone 5 that is transformed by τ_2. The internal transform in each line is applied using (12) for zone 3 to zone 6 as follows:

$$\forall \mathrm{x}, \mathrm{y} \in \mathrm{L_i},\quad \mathrm{i} = [0, \mathrm{n}-1],$$
$$\mathrm{L_i new_{(x,y)}} = \mathrm{L_{(n-i)_{(x,n-y)}}}, \tag{12}$$

where Lnew is the transformed line. The transform from zone 3 to zone 6 is shown in Fig. 14.

For direct rotation in zone 6, vertical lines instead of horizontal lines and $f(\mathrm{y})$ instead of $f(\mathrm{x})$ are used.

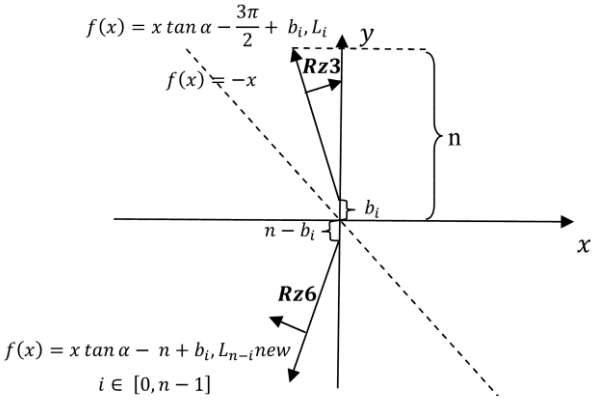


Fig. 14. Transform of lines from zone 3 to zone 6 using τ_2.

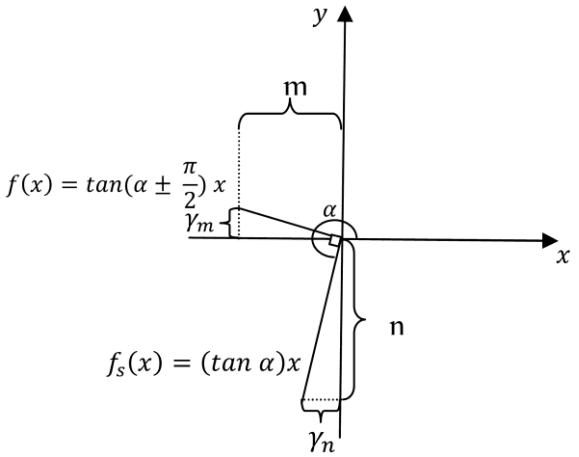


Fig. 15. Direct line rotation in zone 6 using the base-line and starting points.

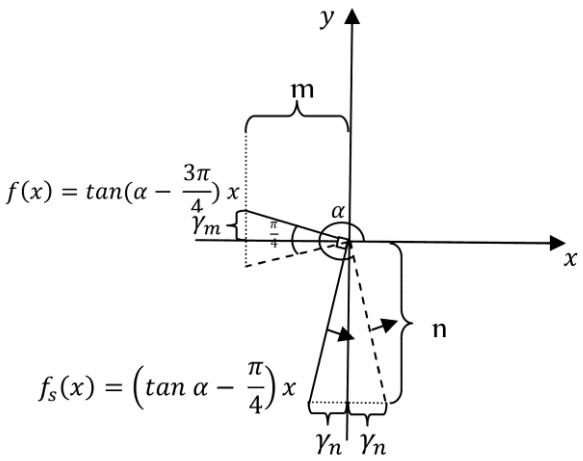


Fig. 16. Transform of lines from zone 6 to zone 7 using τ_3.

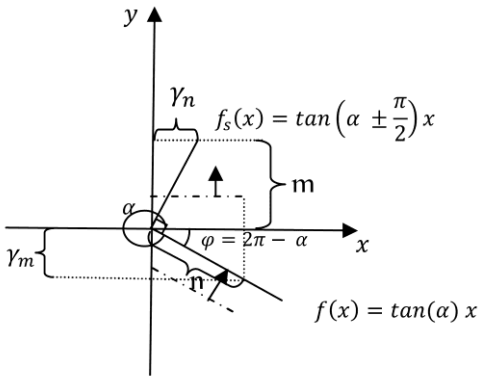


Fig. 17. Direct line rotation in zone 8 using the base-line and starting points.

Values on the y-axis are given and available, but values on the x-axis are unknown. The dimensions of the rotated image R ($m_{rt} \times n_{rt}$) are $m_{rt} = \gamma_m + n$ and $n_{rt} = \gamma_n + m$, where $\gamma_m = \| f(m) \|$ and $\gamma_n = f_s(n) \rightarrow f_s(y) = \frac{y}{\tan\alpha}$, as indicated by Fig. 15. The rotation equation is presented in (13).

$$\forall x, y \in T,\ \exists X,\ Y \in R_t, X = n_{rt} - x - f_s(y),\quad Y = f(m - y), \tag{13}$$

where x, y are the image indices that map to X, Y as new matrix indices of the rotated image.

7) Zone 7: In zone 7, a rotation can be performed directly or by a transform from other vertical-line zones (Z2, Z3, or Z6). Z7 can be generated using τ_1 from zone 3 or τ_2 from zone 2. It can also be generated from Z6 using $\tau_3 | X = -x, Y = y$, as shown in Fig. 16. Each vertical line in the rotated image in zone 6 uses (14) to transform to zone 7.

$$\forall x,\ y \in L_i,\ i = [0, n],\ L_i new_{(x,y)} = L_{(i)_{(m-x,y)}}. \tag{14}$$

For direct rotation in zone 7, as for zone 6, vertical lines instead of horizontal lines and $f(y)$ instead of $f(x)$ are used. The dimensions of the rotated image R($m_{rt} \times n_{rt}$) are $m_{rt} = \gamma_m + n$ and $n_{rt} = \gamma_n + m$, where $\gamma_m = f(m)$ for $f(x) = x\tan(\alpha + \frac{\pi}{2})$ and $\gamma_n = \| f_s(n) \| \rightarrow f_s(y) = \frac{y}{\tan\alpha}$. The rotation equation is presented in (15).

$$\forall x,\ y \in T,\ \exists X,\ Y \in R_t,\ X = m - y + f_s(x),\quad Y = f(y), \tag{15}$$

where x, y are the image-matrix indices that map to X, Y as the new matrix indices of the rotated image.

8) Zone 8: In zone 8, the rotated image can be generated from other horizontal-line zones (Z1, Z4, and Z5), using τ_1 for a transform from Z4, using τ_2 for a transform from Z1, and using τ_3 for a transform from Z5. As Fig. 17 shows, the slope of $f_s(x)$ is positive, but $f(x)$ has a negative gradient. Each horizontal line in image T lies in a line with slope $\tan(\alpha)$ and shears using line $f_s(x)$ with equation $f_s(x) = \left(\tan\alpha + \frac{\pi}{2}\right) x$. The dimensions of the rotated image are $m_{rt} = \gamma_m + m$ and $n_{rt} = \gamma_n + n$, where $\gamma_n = f_s(m) \rightarrow \left\| \frac{m}{\tan(\alpha + \frac{\pi}{2})} \right\|$ and $\gamma_m = f(n) \rightarrow \| n\tan\alpha \|$. Each horizontal line in the $m \times n$ image T with equation $y = b_i, b_i = [0, m-1]$, is mapped to $f(x) = Sx + b_i$ and shears using $f_s(x) = \left(\tan\alpha + \frac{\pi}{2}\right) x$. The rotation is specified in (16).

$$\forall x,\ y \in T,\ \exists X,\ Y \in R_t,\ X = \gamma_n - f_s(y) + x,\quad Y = f(x). \tag{16}$$

Direct line rotation and separate calculations for rotation in each zone have been provided. Therefore, pre-rotation to

the basic zones and then using a transform of τ_1, τ_2 or τ_3 to complete the rotation is no longer necessary; thus, the speed is higher because of the direct rotation. The proposed method primarily uses integer addition and logical justification after calculating the base-line and start-line. This significantly reduces the number of floating-point computations and thus can be rapidly performed.

IV. Evaluation

The image-rotation methods are evaluated based on two measures: accuracy and speed. A high accuracy of image rotation is fundamental for follow-up processing such as feature extraction and feature matching [1], [3]. A high speed of rotation is also significant in many applications, especially real-time systems [2], [14], [25]. The proposed method (DLR) can perform rapid rotation because it primarily uses integer addition and index justification. The speed and response time of DLR are assessed based on the number of multiplications and additions of integers or floating-point numbers. The accuracy of DLR is assessed using evaluation metrics such as the peak signal-to-noise ratio (PSNR) [33], [34], structural similarity (SSIM) [35], [36], relative entropy (Re) [1], [37], Laplacian mean-square error (LMSE), normalized absolute error (NAE), and cross-correlation (CC) [37] and compared with well-known methods in six types of tests.

A. Accuracy Analysis

We define six types of tests to assess the accuracy of the rotation methods. First, the reliability and accuracy of the DLR should be demonstrated; therefore, test 1 for all available zones is defined. Six assessment metrics are used for image comparison, and three standard images, the Baboon, Lenna, and the Boat, are used in the assessment.

The well-known PSNR, which is classified under difference-distortion metrics, is applied to the rotated and original images as a performance measurement for image distortion. The PSNR is defined in (17) [33], [34] as follows:

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{c_{\max}^2}{\mathrm{MSE}}\right), \tag{17}$$

where MSE is given in (18).

$$\mathrm{MSE} = \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n}\left(R_{xy} - T_{xy}\right)^2, \tag{18}$$

and $C_{\max}$ is the maximum value in the original image; for example,

$$C_{\max} \leq \begin{cases} 1, & \text{in double-precision intensity images} \\ 255, & \text{in 8-bit unsigned-integer intensity images.} \end{cases}$$

In (18), x and y are the image pixel coordinates, m and n are the dimensions of the image, R_{xy} is the rotated image, and T_{xy} is the original image. The PSNR is often expressed on a logarithmic scale in decibels (dB). Combining (17) and (18) provides (19) as the definition of the PSNR.

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{c_{\max}^2}{\frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(R(i,j)-T(i,j))^2}{mn}}\right). \tag{19}$$

The SSIM between the original image and the target image is applied as another evaluation metric. SSIM uses comparisons of three components: the luminance, contrast, and structure. Finally, the three components are combined to yield an overall similarity measure, as indicated by (20) [35], [36].

$$\mathrm{SSIM}(x, y) = [l(x, y)]^{\alpha}.[c(x, y)]^{\beta}.[s(x, y)]^{\gamma}, \tag{20}$$

where $\alpha > 0$, $\beta > 0$, and $\gamma > 0$ are parameters that are used to adjust the relative importance of the three components, namely, the luminance (21), contrast (22), and structure (23).

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2+\mu_y^2+C_1}, \tag{21}$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2+\sigma_y^2+C_2}, \tag{22}$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y+C_3}, \tag{23}$$

where $\sigma_x = \sqrt[2]{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2}$, $\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y)$, and $\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i$, $\mu_y = \frac{1}{N}\sum_{i=1}^{N} y_i$. C_1,C_2, and C_3 are constants. C_1 is sufficiently small to be ignored (relative to μ_x^2). $C_1 = (K_1L)^2$, $C_2 = (K_2L)^2$, and $K_1, K_2 \ll 1$. It is easy to verify that this definition satisfies the three conditions given above. To simplify the expression SSIM, we set $\alpha = \beta = \gamma = 1$ and define $C_3 = C_2/2$ such that $C_1 \cong C_2 \cong C_3 \cong 0$. The result of a specific form of the SSIM is given in (24) [35].

$$\mathrm{SSIM}(x, y) = \frac{4\mu_x\mu_y\sigma_{xy}}{(\mu_x^2+\mu_y^2)(\sigma_x^2+\sigma_y^2)}. \tag{24}$$

The Re is used as a method for evaluation accuracy. The Re, or Kullback–Leibler divergence, is an asymmetric measure of the discrepancy between two probability distributions [38]. It can also be used to measure the difference between images and is defined in (25) [1], [37].

$$\mathrm{Re} = \sum_x \log\left[R(x)\left(\frac{R(x)}{T(x)}\right)\right], \tag{25}$$

where R(x) and T(x) are the image functions of the rotated image and the original image, respectively. Usually, the Re is used for histogram evaluations. Let T(x) be the histogram of the original image, let R(x) be the histogram of the rotated image for the accuracy test, and $x \in [0, 255]$.

The result is also evaluated using the LMSE, which is defined in (26).

$$\mathrm{LMSE} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}[\omega(x_{i,j}) - \omega(x'_{i,j})]}{\sum_{i=1}^{m}\sum_{j=1}^{n}\omega(x_{i,j})^2},$$
$$\omega(x_{i,j}) = x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1} - 4x_{i,j}. \tag{26}$$

The result of the original image and the rotated image can be assessed using the NAE, which is defined in (27).

$$\mathrm{NAE} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|x_{i,j} - x'_{i,j}\right|}{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|x_{i,j}\right|}. \tag{27}$$

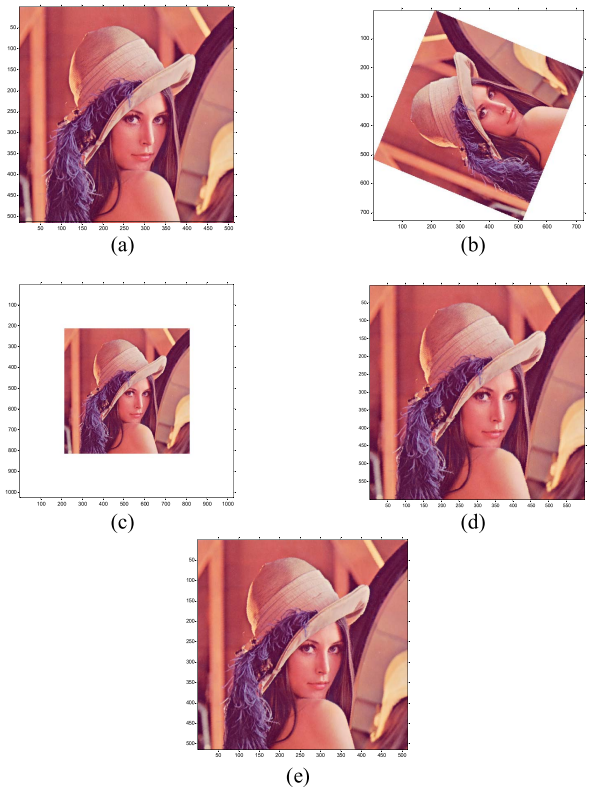


Fig. 18. Sample of the assessment of test 1 for $\frac{3\pi}{8}$ rad. (a) The original image. (b) Rotating the original image using DLR by $\frac{3\pi}{8}$ rad. (c) Rotating the image using DLR by $-\frac{3\pi}{8}$ rad. (d) Trimming the margin from the rotated image. (e) Resizing the trimmed image to the original image size for assessment methods that require the original and rotated image to be of the same size.

Furthermore, the CC, which is defined in (28), is used in the evaluation [37].

$$CC = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}.x'_{i,j}}{\sum_{i=1}^{m}\sum_{j=1}^{n} x_{i,j}^{2}}. \quad (28)$$

In (26), (27), and (28), x is a pixel from the original image, and x' is a pixel from the rotated image with coordinates i, j.

Six types of tests for assessing the accuracy of the proposed rotation method are defined. These six tests are based on the fact that if an image-rotation method is accurate, it will preserve the information of the original image over single or consecutive rotations, and the resultant image should be similar to the original [1].

Test 1 demonstrates the reliability and accuracy of DLR in all zones. In test 1, as shown in Fig. 18, an image rotates by $i\frac{\pi}{8}$, where$1 \leq i \leq 16$ indicates $\frac{\pi}{8}$, $\frac{\pi}{4}$, $\frac{3\pi}{8}$, $\frac{\pi}{2}$, $\frac{5\pi}{8}$, $\frac{3\pi}{4}$, $\frac{7\pi}{8}$, π, $\frac{9\pi}{8}$, $\frac{5\pi}{4}$, $\frac{11\pi}{8}$, $\frac{3\pi}{2}$, $\frac{13\pi}{8}$, $\frac{7\pi}{4}$, $\frac{15\pi}{8}$ and 2π rad. Then, each rotated image is rotated in the opposite direction by the same angle to return to the original situation and is compared to the original image using an assessment method. The rotated image is resized to the original image size using the nearest-neighbor interpolation technique for assessment methods that require the original and rotated images to be of the same size.

DLR is compared with various currently standard rotation methods. Standard well-known methods for rotation, such as nearest neighbor, bilinear, and bicubic interpolation, apply three-pass algorithms with different interpolation techniques [6]. Furthermore, DLR is compared with the FFT [4], [27], [29], [30] and Hermite [1] methods for rotation. DLR provides accurate results for most rotation radians. As Fig. 19 demonstrates, on average, DLR is the most accurate method when compared with other methods using the PSNR metric. Table I presents the SSIM results from test 1 for Lenna, where on average, bilinear interpolation provides a rotated image that is most similar to the original. Fig. 19 shows the PSNR comparison for test 1 for the Boat image using different rotation angles.

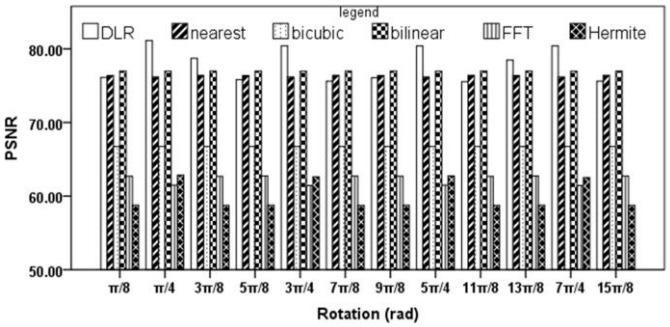


Fig. 19. PSNR comparison for test 1 for the Boat image.

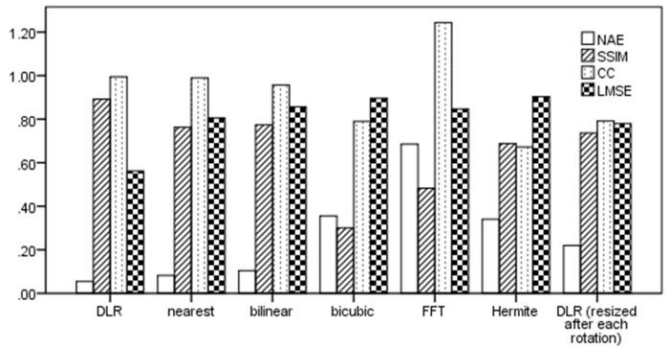
TABLE I
SSIM IN TEST 1 FOR LENNA

Rad	*nearest*	*bilinear*	*bicubic*	*FFT*	*Hermite*	*DLR*
π/8	0.974206	0.985213	0.776901	0.857801	0.694935	0.969463
π/4	0.971477	0.985151	0.776978	0.838937	0.470047	0.986981
3π/8	0.974273	0.9852	0.77701	0.86177	0.694912	0.984616
π/2	1	1	1	0.978798	0.695166	1
5π/8	0.974206	0.985213	0.776901	0.86156	0.694936	0.972757
3π/4	0.971477	0.985151	0.776978	0.840723	0.442725	0.988241
7π/8	0.974273	0.9852	0.77701	0.861494	0.694913	0.965185
π	1	1	1	0.978637	0.695166	1
9π/8	0.974206	0.985213	0.776901	0.857801	0.694937	0.967107
5π/4	0.97148	0.985151	0.776978	0.838937	0.443166	0.988573
11π/8	0.974273	0.9852	0.77701	0.86177	0.694913	0.971746
3π/2	1	1	1	0.978798	0.695166	1
13π/8	0.974206	0.985213	0.776901	0.86156	0.694936	0.984244
7π/4	0.971493	0.985151	0.776978	0.840723	0.440494	0.988241
15π/8	0.974273	0.9852	0.77701	0.861494	0.694913	0.965185
2π	1	1	1	0.978637	0.695166	1
Avg.	0.97999	0.988891	0.832722	0.884965	0.633531	0.983271

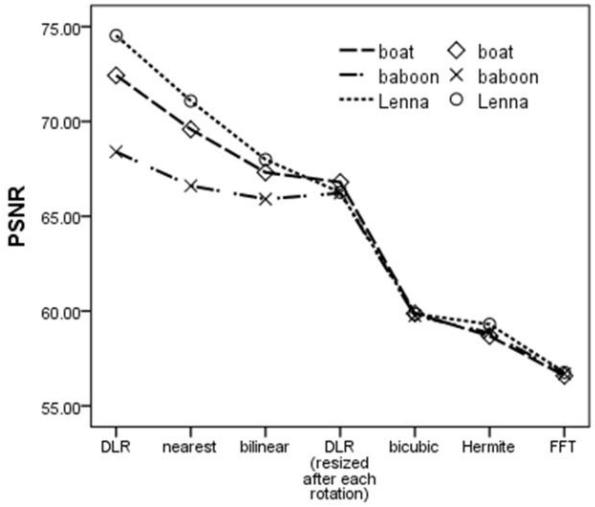
TABLE II
RESULTS OF TEST 2 FOR THE BABOON

method	*PSNR*	*SSIM*	*CC*	*Re*	*LMSE*	*NAE*
DLR (resized after last rotation)	68.39901	0.689736	0.974663	42.40087	0.636096	0.128475
DLR (resized after each rotation)	66.22091	0.633647	0.819426	283.6668	0.642728	0.200899
nearest	66.60464	0.480337	0.968919	21.40505	0.755788	0.162249
bilinear	65.90593	0.507828	0.936782	81.17545	0.831101	0.166768
bicubic	59.75659	0.128361	0.78587	206.3038	0.876915	0.365387
FFT	56.69499	0.380077	1.237759	4761.565	0.800127	0.676419
Hermite	58.858	0.704401	0.743902	364.5623	0.871667	0.356874

Test 2: The image is rotated by angles α in the range $[0, 2\pi[$ at $\frac{\pi}{8}$ intervals, which means that the image rotates by $\frac{\pi}{8}$ rad sixteen times consecutively, and the resulting image is compared with the original image. The DLR image obtained after sixteen rotations is larger than the original image; therefore, the resulting image was resized to the original image size using the nearest neighbor interpolation technique for those assessment methods that require the rotated image and the original image to be of the same size. There is an alternative for test 2: each rotated image by DLR can be resized to its actual size at each step after rotation by $\frac{\pi}{8}$ rad. Table II shows the results



(a)



(b)

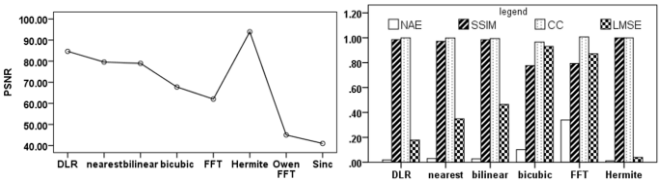
Fig. 20. (a) Comparison result of test 2 with Lenna using SSIM, CC, LMSE, and NAE. (b) PSNR results for test 2 with Lenna, the Boat, and the Baboon.

of test 2 for the Baboon image, where for DLR, a) the rotated image was resized in the last step, and b) the rotated image was resized at each step after rotation. Fig. 20 shows the result after sixteen rotations, where the DLR resulting image is cropped and resized to the original image size in the last step. Fig. 20(a) shows the results of test 2 for Lenna. DLR usually provides the lowest error rate in terms of NAE and LMSE. The DLR method yields acceptable results for SSIM and CC. The CC value of DLR is smaller than that of FFT, but this difference is offset by DLR's speed advantage. Fig. 20(b) shows the PSNR results for the Boat, Baboon and Lenna images. The result in Fig. 20 demonstrates the robustness and accuracy of DLR. Hermite and FFT methods distort the edge of the image, as demonstrated in Fig. 23(a). Using extra margins for image rotation can solve this problem, although this increases the computation time and the memory usage. Hermite and FFT rotation are slow rotation methods, although they are efficient in terms of preserving important information in the image such as edges (i.e., high frequency areas). The Hermite method is used for comparison based on its application as implemented and published by Park *et al.* [1].

Test 3: The image is rotated by $\frac{\pi}{4}$ rad two times consecutively, and the resulting image and the image rotated once by $\frac{\pi}{2}$ rad are compared [1]. Table III presents the results of test 3 for the Baboon image. Fig. 21(a) shows the PSNR result for Lenna. DLR utilizes a second step after the Hermite rotation.

TABLE III
RESULTS OF TEST 3 FOR THE BABOON

method	*PSNR*	*CC*	*LMSE*	*NAE*	*Re*	*SSIM*
DLR	71.50826	0.987661	0.501018	0.0772	9.477901	0.926652
nearest	70.34986	0.987167	0.668752	0.092756	3.29434	0.889552
bilinear	72.66044	0.982108	0.508272	0.077538	22.35235	0.943504
bicubic	64.85791	0.943975	0.849141	0.184472	31.50289	0.499904
FFT	61.39371	0.906034	0.903734	0.231378	335.5602	0.830282
Hermite	87.68847	1.000664	0.078921	0.023408	1266.289	0.927157

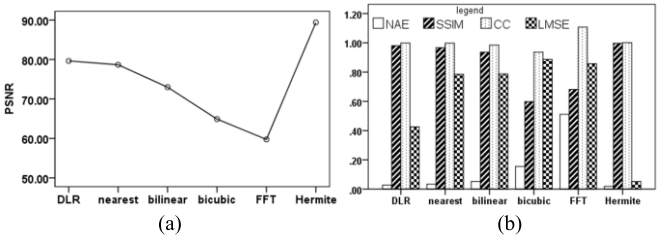


(a) (b)

Fig. 21. Results of test 3 using Lenna. (a) PSNR in test 3 using Lenna. (b) Comparison of results using SSIM, CC, LMSE, and NAE.

TABLE IV
RESULTS OF TEST 4 FOR THE BABOON

method	*PSNR*	*CC*	*LMSE*	*NAE*	*Re*	*SSIM*
DLR	71.70862	0.987916	0.612487	0.081314	8.496067	0.920345
nearest	69.83619	0.986216	0.666108	0.099953	3.058059	0.878662
bilinear	68.82763	0.966673	0.779267	0.117052	38.14799	0.802115
bicubic	63.34114	0.91629	0.858878	0.224468	57.23203	0.319276
FFT	60.33622	0.985123	0.881095	0.350305	741.7661	0.738017
Hermite	71.70862	0.987916	0.612487	0.081314	8.496067	0.920345



(a) (b)

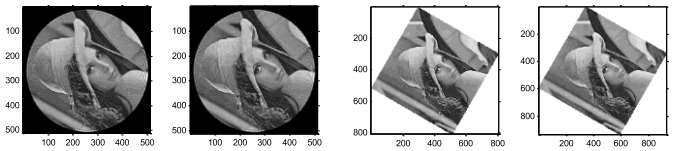
Fig. 22. Test 4 results for Lenna. (a) PSNR results for Lenna. (b) Results of the CC, SSIM, LMSE, and NAE.

Fig. 21(b) shows the results in terms of the CC, LMSE, NAE and SSIM for test 3 using Lenna.

Test 4: An image is rotated by $\frac{\pi}{8}$ rad four times consecutively, and the resulting image and the image rotated once by $\frac{\pi}{2}$ rad are compared. Table IV presents the results of test 4 for the Baboon image. Fig. 22 shows the result of test 4 for Lenna. DLR was the second-most accurate method after the Hermite method.

Test 5: An image is rotated by $\frac{\pi}{6}$ rad twice consecutively, and the resultant image and the image directly rotated by $\frac{\pi}{3}$ rad are compared. Fig. 23 shows some samples of test 5. Tables V and VI present the results of test 5 for the Boat and Baboon images, respectively. The DLR method in test 5 is not successful and remains the fourth-best method after the Hermite, bilinear, and bicubic techniques. Fig. 24(a) shows the comparison results in terms of PSNR for test 5 and Fig. 24(b) shows the test 5 results for Lenna.

Test 6: An image is rotated by $\frac{\pi}{6}$ rad twelve times consecutively, and the resultant image and the original image



(a) (b)

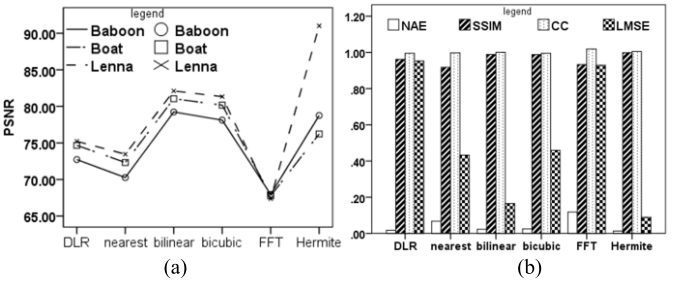
Fig. 23. Test 5 samples with Lenna, wherein the left images are directly rotated by $\frac{\pi}{3}$ rad and the right images are rotated 2 times by $\frac{\pi}{6}$ rad. (a) Rotation using the Hermite method. (b) Rotation using the DLR.

TABLE V
RESULTS OF TEST 5 FOR THE BOAT

method	*PSNR*	*SSIM*	*CC*	*Re*	*LMSE*	*NAE*
DLR	74.668	0.938183	0.994805	0.810833	1.065793	0.022119
nearest	72.32122	0.880603	0.995618	1.037359	0.688978	0.08286
bilinear	81.04754	0.983886	1.000059	0.827005	0.267386	0.02825
bicubic	80.16706	0.982976	0.994929	0.725786	0.401223	0.030869
FFT	67.77193	0.934342	1.021355	738.4885	0.720452	0.125073
Hermite	76.22687	0.822591	0.99484	1671.901	0.235456	0.071054

TABLE VI
RESULTS OF TEST 5 FOR THE BABOON

method	*PSNR*	*CC*	*LMSE*	*NAE*	*Re*	*SSIM*
DLR	72.72055	0.993757	0.936884	0.034267	0.916851	0.921755
nearest	70.26833	0.987673	0.641127	0.131608	0.964096	0.836101
bilinear	79.23585	0.998174	0.203932	0.044799	1.172176	0.97829
bicubic	78.11803	0.993144	0.416951	0.050511	0.868322	0.976939
FFT	67.97017	1.022549	0.866301	0.113103	614.0791	0.936086
Hermite	78.75304	0.99269	0.310416	0.057518	2960.568	0.978311



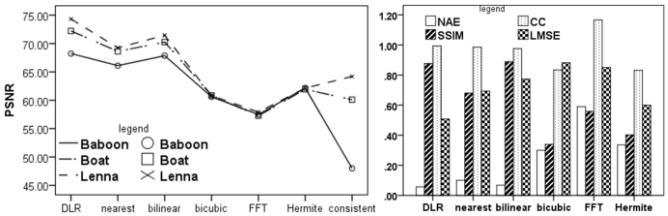
(a) (b)

Fig. 24. Results of test 5. (a) Comparison of PSNR for test 5 for the Boat, the Baboon and Lenna. (b) Results in terms of the CC, SSIM, LMSE, and NAE for Lenna.

TABLE VII
RESULTS OF TEST 6 FOR THE BABOON

Method	*PSNR*	*CC*	*LMSE*	*NAE*	*Re*	*SSIM*
DLR	68.23262	0.973453	0.586085	0.13176	39.94552	0.665394
nearest	66.10555	0.965671	0.675338	0.172607	20.63307	0.403003
bilinear	67.86149	0.955983	0.769368	0.135526	63.2731	0.676255
bicubic	60.60908	0.826996	0.862717	0.323098	158.3176	0.161767
FFT	57.49836	1.18753	0.842037	0.612202	3919.447	0.492892
Hermite	62.17504	0.78155	0.717647	0.321194	2166.982	0.205457

are compared [1], [15]. Table VII presents the results of test 6 for the Baboon. The proposed method provides the best results in test 6. Fig. 25(a) shows the PSNR results for the Boat, the Baboon, and Lenna, wherein the DLR method yields the highest value. Fig. 25(b) shows the results of test 6 for Lenna.



(a) (b)

Fig. 25. (a) PSNR results of test 6 for Lenna, the Boat, and the Baboon. (b) Results of test 6 for Lenna.

TABLE VIII
NUMBER OF MULTIPLICATION AND ADDITION OPERATIONS IN DIFFERENT ROTATION METHODS

Method	*Multiplication*	*Addition*
one-pass nearest neighbor interpolation	$4n^2$	$4n^2 + 2n$
one-pass bilinear interpolation	$4n^2 + 4n$	$7n^2$
one-pass bicubic interpolation	$68n^2 + 4n$	$37n^2$
two-pass linear interpolation	$4n^2 + 4n$	$6n^2$
two-pass cubic interpolation	$44n^2 + 4n$	$22\,n^2$
three-pass linear interpolation	$6n^2 + 3n$	$6n^2 + 3n$
three-pass cubic interpolation	$12n^2 + 57n$	$12n^2 + 30n$
three-pass linear spline interpolation	$54n^2 - 20n - 4$	$27n^2 - 7n - 2$
three-shear without interpolation	$3n$	$3n^2 + 3n$
FFT	$64n^2 + 16n$	$34n^2 + 27n$
midpoint line	$8n$	$6n^2 + 12n$
DLR	$2n$	$4n^2 + 3n + 1$
DLR + resizing (nearest)	$2n^2 + 2n$	$5n^2 + 3n + 1$

B. Computational and Time Analysis

Single- and multi-pass algorithms that use various interpolation techniques, such as nearest neighbor, linear, cubic, sinc, midpoint, and cubic spline interpolation, have been developed for application to the problem of image rotation. The target image can be resampled based on linear, bilinear, bicubic or consistence resampling theory. Sinc and cubic spline interpolation produce high-quality images, but they are computationally intensive and not suitable for real-time rotation applications. The overall performance of three-pass algorithms is better than that of two-pass and one-pass algorithms because three-pass algorithms have no signal contraction and because the interpolation is performed by convolution [6].

In image rotation algorithms where there are unallocated pixels or incorrect aspect ratios, resizing and resampling functions must be applied to provide the correct result and to avoid information loss, whereas DLR provides the magnified result without using any resampling or resizing techniques. This study focuses on a high-speed image rotation algorithm to retain fixed-order shape contours, as shown in Fig. 7(g). The magnification may not be optimal for certain image processing tasks. In those cases, resizing to the actual size of the original

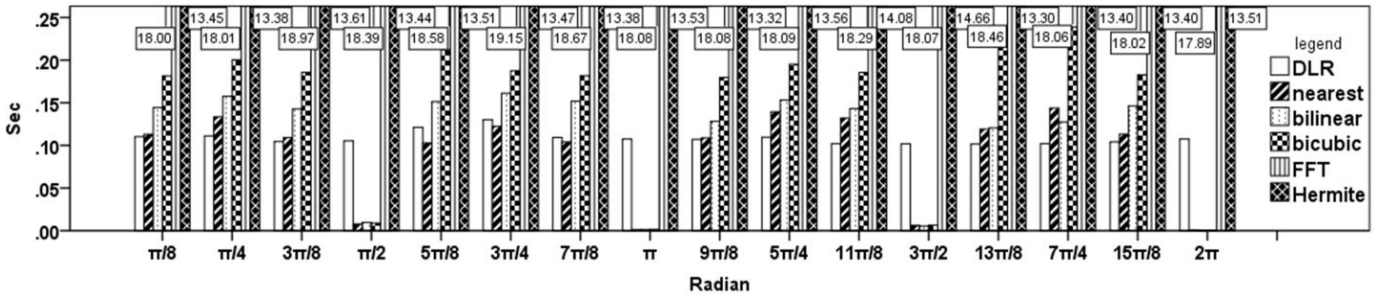


Fig. 26. Response time (s) for the Baboon rotation (the FFT and Hermite methods have a response time greater than 1 s).

TABLE IX
NUMBER OF FLOATING AND INTEGER MULTIPLICATIONS, ADDITIONS, ROUNDING, INCREMENTING/DECREMENTING, AND INDEX LOGICAL JUSTIFICATION IN DIFFERENT ROTATION METHODS

Methods	*Floating Computation*			*Integer Computation*		
	mul.	*add*	*round*	*add.*	*inc/dec 1*	*JUST*
one-pass	$4n^2$	$2n^2$	$2n^2$	-	n^2+n	n^2+n
two-pass	$6n$	$3n$	$3n$	$2n^2$	$2n^2+2n$	$2n^2+2n$
three-pass (three-shear)	$3n$	-	$3n$	$3n^2$	$3n^2+3n$	$3n^2+3n$
midpoint line[2]	$8n$	$5n$	$5n$	n^2+7n	$3n^2$	$2n^2$
DLR	$2n$	-	$2n+2$	$2n+1$	$2n^2+$ n	$2n^2$
DLR + resizing (nearest)	$2n^2+2n$	-	$2n^2+2n+2$	$2n+1$	$4n^2+$ n	$4n^2$

TABLE X
RESPONSE TIME (S) FOR TEST 1 FOR LENNA

Rad	nearest	bilinear	bicubic	FFT	Hermite	DLR	DLR + resizing (nearest)
π/8	0.11168	0.132247	0.179404	18.18878	13.54004	0.111069	0.127230
π/4	0.166283	0.133934	0.196448	17.89464	13.53307	0.104381	0.138225
3π/8	0.107664	0.112442	0.18084	18.23758	13.87688	0.103818	0.116778
π/2	0.066789	0.008666	0.005414	17.8762	14.11454	0.10293	0.10293
5π/8	0.107051	0.118927	0.189431	18.01728	13.58028	0.102101	0.114976
3π/4	0.108487	0.133709	0.188261	17.99642	13.98368	0.10934	0.139557
7π/8	0.102756	0.122321	0.181368	18.29607	13.52166	0.108324	0.122480
π	0.001694	0.001324	0.001384	18.49404	13.98547	0.107289	0.107289
9π/8	0.102553	0.114668	0.194242	18.29927	13.52597	0.105463	0.121739
5π/4	0.114896	0.123072	0.214647	18.81671	13.55645	0.110029	0.129875
11π/8	0.108532	0.113275	0.179906	18.16149	13.72339	0.102617	0.118653
3π/2	0.010639	0.005358	0.005348	18.01363	13.811	0.104034	0.104034
13π/8	0.107503	0.125481	0.182368	17.92197	13.80063	0.101892	0.115081
7π/4	0.140026	0.128894	0.198516	18.10618	13.47072	0.10161	0.133813
15π/8	0.106178	0.115291	0.189491	17.92537	14.25543	0.100884	0.114901
2π	0.000494	0.000402	0.000315	17.93036	13.87295	0.109373	0.109373

image is necessary. The DLR preserves sharp edges, but it increases aliasing (the same as rotation using the nearest neighbor interpolation technique). Aliasing of the edges of the rotated image (see Fig. 18b) can be reduced by resampling. Several approaches that attempt to perform optimization by interpolating areas of continuous tone, preserve the sharpness of horizontal and vertical lines and smooth all other curves have been developed. Table X shows that the proposed algorithm remains fast, where the magnified rotated image is resized to the actual size using the nearest neighbor interpolation technique. Furthermore, the quality assessment results demonstrate the high quality of the resized DLR image compared with those produced by single- or multi-pass algorithms.

Table VIII presents the number of multiplications and additions required. Table IX presents the same information, but in greater detail. Based on Tables VIII and IX, DLR requires fewer floating and integer computations compared to other methods and thus provides high-speed rotation. The floating-point computation and rounding operations in DLR are only performed n + m times. The complexity is O(n + m), and when n ≈ m is assumed, the complexity is O(2n). 2n is still linear; thus, we can consider the loop to be of order O(n).

Table X presents the response times from test 1 for Lenna. Fig. 26 shows the response time for certain rotation angles. The FFT and Hermite response times are greater than 1 s and thus exceed the chart bound in Fig. 26. The Hermite and FFT rotation methods provide the highest quality rotated image, but these methods are more than 13-times slower than a normal rotation. The DLR provides a trade-off between high quality and high-speed image rotation. Fig. 26 and Table X show the average elapsed time for 100 runs in a system with a Core i5 2.3 GHz CPU and 4 GB of RAM.

The DLR does not provide increased memory efficiency. It requires both the original image and the target frame for image rotation, whereas multi-pass algorithms (especially three-shear algorithms) can rotate images in the same frame. Moreover, the DLR resulting image is larger than its actual size; therefore, it raises the expense of computer memory and storage space.

The DLR rotation by $\pi/2$, π, $3\pi/2$, and 2π rad is slower than the nearest neighbor, bilinear, and bicubic methods because they use direct index justification without interpolation; however, the DLR method still determines the base-line and start-line equations to rotate by $\pi/2$, π, $3\pi/2$, and 2π rad. The same speed for each of the aforementioned angles is achieved if direct index justification is applied. For example, in the image T(x, y) for rotation by π, only a simple transform given by $X = -x$, $Y = -y$ and translation the origin to have positive coordinates ($\forall X, Y \rightarrow Z^{\geq}$) are sufficient; therefore, determining the base-line equation and the other corresponding lines is no longer necessary.

V. CONCLUSION

A new one-pass method, called DLR, for realizing high-speed, high-quality image rotation has been proposed. The new method, which is based on line equations and line rotation, employs the line coordinates in the target image for rotation. Vertical or horizontal lines from the original image, depending on the rotation angle, are used. The proposed

method calculates the size, base-line and start-line equations in the target image and extracts all corresponding pixels on both the base-line and start-line. The extracted line pixels are determined only once for the base-line and start-line, as they can be calculated for other lines by addition or subtraction from the base-line coordinates. Each line in the original image is transformed into two corresponding lines in the target image as an unallocated pixel-filling method. DLR primarily uses integer addition and logical index justification, significantly reducing the number of floating-point computations and allowing for efficient rotation. DLR provides rotated images without any changes in the pixel values and it retains the size of the shape contour in the grid view. The implementation results reveal good performance of the DLR method, which is characterized by rapid execution, along with acceptable accuracy compared to other methods currently used as standard. The image information is perfectly retained, and the rotated image can be applied as a basis for other pattern-recognition tasks and issues.

REFERENCES

[1] W. Park, G. Leibon, D. N. Rockmore, and G. S. Chirikjian, "Accurate image rotation using Hermite expansions," *IEEE Trans. Image Process.*, vol. 18, no. 9, pp. 1988–2003, Sep. 2009.

[2] C. Xuede, L. Siwei, Y. Xiaobu, C. Ling, and Z. Beikai, "Midpoint line algorithm for high-speed high-accuracy-rotation of images," in *Proc. IEEE Int. Conf. Syst., Man, Cybernet.*, Oct. 1996, pp. 2739–2744.

[3] M. Unser, P. Thevenaz, and L. Yaroslavsky, "Convolution-based interpolation for fast, high-quality rotation of images," *IEEE Trans. Image Process.*, vol. 4, no. 10, pp. 1371–1381, Oct. 1995.

[4] A. W. Lohmann, "Image rotation, Wigner rotation, and the fractional Fourier transform," *J. Opt. Soc. Amer. A, Opt. Image Sci.*, vol. 10, no. 10, pp. 2181–2186, 1993.

[5] S. Banerjee and A. Kuchibhotla, "Real-time optimal-memory image rotation for embedded systems," in *Proc. 16th Int. Conf. Image Process. (ICIP)*, Nov. 2009, pp. 3277–3280.

[6] R. S. Watve, A. S. Shende, and S. Kshirasagar, "Real time image rotation using SoC architecture," in *Proc. 8th Int. Symp. Signal Process. Appl.*, Aug. 2005, pp. 795–798.

[7] B. Zitová and J. Flusser, "Image registration methods: A survey," *Image Vis. Comput.*, vol. 21, no. 11, pp. 977–1000, Oct. 2003.

[8] T. M. Lehmann, C. Gonner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Trans. Med. Imag.*, vol. 18, no. 11, pp. 1049–1075, Nov. 1999.

[9] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 29, no. 6, pp. 1153–1160, Dec. 1981.

[10] D. E. Friedmann, "Two-dimensional resampling of line scan imagery by one-dimensional processing," *Photogramm. Eng. Remote Sens.*, vol. 47, no. 1981, pp. 1459–1467, Oct. 1981.

[11] P.-E. Danielsson and M. Hammerin, "High-accuracy rotation of images," *CVGIP, Graph. Models Image Process.*, vol. 54, no. 4, pp. 340–344, Jul. 1992.

[12] A. Tanaka, M. Kameyama, S. Kazama, and O. Watanabe, "A rotation method for raster image using skew transformation," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 1986, pp. 272–277.

[13] A. W. Paeth, "A fast algorithm for general raster rotation," in *Proc. Graph. Vis. Inter.*, 1986, pp. 77–81.

[14] Z. Yu, J. Dong, Z. Wei, and J. Shen, "A fast image rotation algorithm for optical character recognition of Chinese documents," in *Proc. Int. Conf. Commun., Circuits Syst.*, Jun. 2006, pp. 485–489.

[15] B. Huang, E. M.-K. Lai, and A. P. Vinod, "Image resizing and rotation based on the consistent resampling theory," in *Proc. Int. Symp. Intell. Signal Process. Commun. Syst. (ISPACS)*, Feb. 2009, pp. 1–4.

[16] T. Thong, "Frequency domain analysis of two-pass rotation algorithm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 1985, pp. 1333–1336.

[17] D. Fraser and R. A. Schowengerdt, "Avoidance of additional aliasing in multipass image rotations," *IEEE Trans. Image Process.*, vol. 3, no. 6, pp. 721–735, Nov. 1994.

[18] C. Berthaud, E. Bourennane, M. Paindavoine, and C. Milan, "Implementation of a real time image rotation using B-spline interpolation on FPGA's board," in *Proc. Int. Conf. Image Process. (ICIP)*, Oct. 1998, pp. 995–999.

[19] T. Toffoli and J. Quick, "Three-dimensional rotations by three shears," *Graph. Models Image Process.*, vol. 59, no. 2, pp. 89–95, Mar. 1997.

[20] D. Kermisch, "Rotation of digital images," U.S. Patent 4 545 069 A, Oct. 1, 1985.

[21] G. Costantini, D. Casali, and R. Perfetti, "Cellular neural network template for rotation of grey-scale images," *Electron. Lett.*, vol. 39, no. 25, pp. 1803–1805, Dec. 2003.

[22] T. Kriz and D. Bachman, "A number theoretic transform approach to image rotation in parallel array processors," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 1980, pp. 430–433.

[23] S. Suchitra, S. K. Lam, C. T. Clarke, and T. Srikanthan, "Accelerating rotation of high-resolution images," *IEE Proc.-Vis., Image Signal Process.*, vol. 153, no. 6, pp. 815–824, Dec. 2006.

[24] S. Suchitra, S. K. Lam, and T. Srikanthan, "Novel schemes for high-throughput image rotation," in *Proc. 38th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2004, pp. 1884–1888.

[25] S.-I. Chien and Y.-M. Baek, "A fast black run rotation algorithm for binary images," *Pattern Recognit. Lett.*, vol. 19, nos. 5–6, pp. 455–459, Apr. 1998.

[26] S.-I. Chien and Y.-M. Baek, "Hierarchical block matching method for fast rotation of binary images," *IEEE Trans. Image Process.*, vol. 10, no. 3, pp. 483–489, Mar. 2001.

[27] K. G. Larkin, M. A. Oldfield, and H. Klemm, "Fast Fourier method for the accurate rotation of sampled images," *Opt. Commun.*, vol. 139, nos. 1–3, pp. 99–106, Jun. 1997.

[28] C. B. Owen and F. Makedon, "High quality alias free image rotation," in *Proc. Conf. Rec. Asilomar Signals, Syst. Comput.*, Nov. 1996, pp. 115–119.

[29] R. W. Cox and R. Tong, "Two- and three-dimensional image rotation using the FFT," *IEEE Trans. Image Process.*, vol. 8, no. 9, pp. 1297–1299, Sep. 1999.

[30] B. S. Reddy and B. N. Chatterji, "An FFT-based technique for translation, rotation, and scale-invariant image registration," *IEEE Trans. Image Process.*, vol. 5, no. 8, pp. 1266–1271, Aug. 1996.

[31] I. Svalbe, "Exact, scaled image rotations in finite Radon transform space," *Pattern Recognit. Lett.*, vol. 32, no. 9, pp. 1415–1420, Jul. 2011.

[32] I. Svalbe, "Exact, scaled image rotation using the finite radon transform," in *Proc. 5th Int. Conf. Discrete Geometry Comput. Imag. (IAPR)*, 2009, pp. 446–456.

[33] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, Jun. 2008.

[34] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. 20th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2010, pp. 2366–2369.

[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[36] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3440–3451, Nov. 2006.

[37] G. P. Penney, J. Weese, J. A. Little, P. Desmedt, D. L. G. Hill, and D. J. Hawkes, "A comparison of similarity measures for use in 2-D-3-D medical image registration," *IEEE Trans. Med. Imag.*, vol. 17, no. 4, pp. 586–595, Aug. 1998.

[38] C. T. Ireland and S. Kullback, "Minimum discrimination information estimation," *Biometrics*, vol. 24, no. 3, pp. 707–713, Sep. 1968.

Amir Hossein Ashtari (S'12–M'15) received the B.S. and M.S. degrees in computer engineering, hardware and architecture of computer systems from Islamic Azad University, Iran, in 2006 and 2009, respectively, and the Ph.D. degree from the Pattern Recognition Research Group, Center for Artificial Intelligence Technology (CAIT), National University of Malaysia (UKM), in 2015. He is currently a Post-Doctoral Fellow with CAIT, UKM. His main research field is computer vision and image processing. His research interests include intelligent transportation systems, image reconstruction, fragment object analysis, shape descriptors, and shape transforms.

Md Jan Nordin (M'15) received the B.S. and M.S. degrees in computer science from Ohio University, Athens, OH, USA, in 1982 and 1985, respectively, and the Ph.D. degree in engineering information technology from Sheffield Hallam University, South Yorkshire, U.K., in 1995. He is currently an Associate Professor with the Center for Artificial Intelligence Technology, National University of Malaysia, Selangor, Malaysia. His current research interests include pattern recognition, computer vision, intelligent systems, and image reconstruction.

Seyed Mostafa Mousavi Kahaki (S'14–M'15) received the B.S. and M.S. degrees in computer engineering, hardware and architecture of computer systems from Islamic Azad University, Iran, in 2006 and 2009, respectively, and the Ph.D. degree. He is currently a Post-Doctoral Fellow with the Pattern Recognition Research Group, Department of Computer Science, Center for Artificial Intelligence Technology, National University of Malaysia. His research interests include computer vision, multiple-view geometry, intelligent systems, feature extraction and matching, and image registration.