Fast Parallel Image Rotation Algorithm

Dhrubajyoti Mandal
School of Computer Engineering
Kalinga Institute of Industrial Technology
Bhubaneswar, India
22053859@kiit.ac.in

Dr. Sujoy Dutta
Asst. Prof. & Asst. CoE
School of Computer Engineering
Kalinga Institute of Industrial Technology
Bhubaneswar, India
sdattafcs@kiit.ac.in

Sourav Kumar Parida
School of Computer Engineering
Kalinga Institute of Industrial Technology
Bhubaneswar, India
22051032@kiit.ac.in

Chiranjeev Bhaya
Lead Engineer (Camera System)
Samsung R&D Institute
Bangalore, India
c.bhaya@samsung.com

Subhadeep Bhadra
School of Computer Engineering
Kalinga Institute of Industrial Technology
Bhubaneswar, India
2205336@kiit.ac.in

Chandra Mohan V
Architect (Camera System)
Samsung R&D Institute
Bangalore, India
chandra.mv@samsung.com

***Abstract*—This paper presents a novel fast image rotation algorithm that leverages CPU parallelization while maintaining high image quality. The proposed method is a single-pass algorithm, derived from the existing Double-Line Rotation (DLR) method, which reduces computational complexity and generalizes the algorithm. Initially, a baseline for the given image is calculated to determine the starting line, which defines the initial point for each vertical or horizontal line in the image to be rotated. The corresponding pixels to be mapped are identified using floating point arithmetic, and trigonometric calculations are performed only once per line. This approach ensures precise image transformation with minimal computational overhead.**

***Index terms:* Image rotation, line rotation image transform, double-line rotation (DLR), parallel image rotation.**

I. Introduction

2-D Image Rotation has a number of applications in digital photography and image processing like image matching, alignment, [1] etc. Highly accurate image rotation is essential for certain image processing tasks such as feature extraction and matching. Whilst there are many state-of-the-art image rotation algorithms, they have limitations in terms of image quality and speed [2] [3]. Parallel algorithms for image processing aim to execute faster than sequential algorithms, however designing and scheduling a fast parallel image rotation is a challenge [4].

The most straightforward, and perhaps the most well known approach to implement image rotation is by using the rotation matrix [5], as described here

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x - cx \\ y - cy \end{bmatrix} * \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

where, θ is the angle of rotation, x' and y' are the new coordinates of the rotated image, and (cx, cy) is the center of the original image, which is treated as the axis of rotation. Assuming the centre of rotation of the new image is (cx', cy'), the final new coordinates are:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x' + cx' \\ y' + cy' \end{bmatrix}$$

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

II. Literature Survey

A. Affine Transformation

A basic idea in computer graphics and geometry, affine transformation is a linear mapping method that maintains straight lines, planes, and points. The capacity to carry out a variety of geometric operations, including translation, scaling, rotation, shearing, and reflection, is what defines an Affine transformation. These transformations are powerful and adaptable tools with a wide range of applications because they are defined mathematically by combining translations and linear transformations.

1) Properties:

- Linearity and Translation: The transformation is composed of a linear transformation Ax and a translation β. This combination allows for a wide range of geometric transformations while preserving the structure of the space.

- Preservation of Collinearity and Ratios: Affine transformations preserve the collinearity of points and the ratios of distances along parallel lines.
- Combining Transformations: Multiple affine transformations can be combined into a single affine transformation by matrix multiplication and vector addition.

Mathematical Formulation

Affine transformations can be described by a combination of linear transformations and translations. The general form of an affine transformation in two dimensions is:

$$y = Ax + b$$

where

- $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$ is the input coordinate vector.
- $\mathbf{A}$ is a 2×2 matrix that represents the linear part of the transformation.
- $\mathbf{b} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$ is the translation vector.

For a rotation by an angle θ around the origin, the matrix $\mathbf{A}$ is:

$$\mathbf{A} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

To center the rotated image within its original dimensions, the translation values t_x and t_y are calculated as follows:

Calculations

New Dimensions

The new width and height of the rotated image are determined using trigonometric functions based on the rotation angle:

$$\text{newWidth} = \lceil \text{width} \cdot |\cos(\theta)| + \text{height} \cdot |\sin(\theta)| \rceil$$

$$\text{newHeight} = \lceil \text{width} \cdot |\sin(\theta)| + \text{height} \cdot |\cos(\theta)| \rceil$$

Center Points

fine transformation matrix that includes rotation and translation is:

$$\mathbf{T} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{bmatrix}$$

The new coordinates (x', y') after applying the affine transformation are:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = \cos(\theta) \cdot x - \sin(\theta) \cdot y + t_x$$

$$y' = \sin(\theta) \cdot x + \cos(\theta) \cdot y + t_y$$

$$c_x = \frac{\text{width}}{2}, \quad c_y = \frac{\text{height}}{2}$$

$$\text{newCx} = \frac{\text{newWidth}}{2}, \quad \text{newCy} = \frac{\text{newHeight}}{2}$$

Translation Values

The translation values to center the rotated image within its new dimensions are:

$$t_x = c_x - (\text{newCx} \cdot \cos(\theta) - \text{newCy} \cdot \sin(\theta))$$

$$t_y = c_y - (\text{newCx} \cdot \sin(\theta) + \text{newCy} \cdot \cos(\theta))$$

Thus, the affine transformation matrix that includes rotation and translation is:

$$\mathbf{T} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{bmatrix}$$

The new coordinates (x', y') after applying the affine transformation are:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = \cos(\theta) \cdot x - \sin(\theta) \cdot y + t_x$$

$$y' = \sin(\theta) \cdot x + \cos(\theta) \cdot y + t_y$$

[6]

III. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] B. Gaster, L. Howes, D. R. Kaeli, P. Mistry, and D. Schaa, *Heterogeneous computing with openCL: revised openCL 1.* Newnes, 2012.

[2] A. H. Ashtari, M. J. Nordin, and S. M. M. Kahaki, "Double line image rotation," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 3370–3385, 2015.

[3] W. Park, G. Leibon, D. N. Rockmore, and G. S. Chirikjian, "Accurate image rotation using hermite expansions," *IEEE Transactions on Image Processing*, vol. 18, no. 9, pp. 1988–2003, 2009.

[4] H.-C. Kwon, H.-J. Cho, and H.-Y. Kwon, "A study on high speed image rotation algorithm using cuda," *The Journal of the Institute of Internet, Broadcasting and Communication*, vol. 16, no. 5, pp. 1–6, 2016.

[5] P. R. Evans, "Rotations and rotation matrices," *Acta Crystallographica Section D: Biological Crystallography*, vol. 57, no. 10, pp. 1355–1359, 2001.

[6] Weisstein and E. W., "Affine transformation."