Affine Transformation of Digital Curves Using Chain Codes

Submitted by

Sandeep Dasgupta and **Subrata Mazumder**
(Regn.No. 002720) (Regn.No. 002724)

under the supervision of
Partha Bhowmick

Computer Science and Technology Department
Bengal Engineering and Science University, Shibpur
Howrah 711 103
India

June 8, 2006

BENGAL ENGINEERING AND SCIENCE UNIVERSITY, SHIBPUR
HOWRAH-711 103

Foreword

The foregoing progress report entitled Affine Transformation of Digital Curves Using Chain Codes prepared by Sandeep Dasgupta (Regn.No. 002720) and Subrata Mazumder (Regn.No. 002724) under my supervision is hereby approved as a creditable study of Computer Science and Technology, carried out and presented satisfactorily to warrant its acceptance as a partial fulfilment for the degree of Bachelor of Engineering in Computer Science And Engineering of the University.

It is understood that by this foreword the undersigned do not necessarily endorse or approve any statement made, opinion expressed, or conclusion drawn therein, but approve the term paper only for the purpose for which it is submitted.

Dated:
Place:

(Partha Bhowmick)
Computer Science and Technology Department
Bengal Engineering and Science University, Shibpur
Howrah 711103

Countersigned by:

(Dr. Uma Bhattacharya)
Professor & Head,
Computer Science and Technology Department
Bengal Engineering and Science University, Shibpur
Howrah 711103

Acknowledgement

First and foremost, we render our sincere respect and deep gratitude to Prof. Partha Bhowmick of Dept. of Computer Science and Technology for allowing us to work on this project under his supervision. We deeply indebted to him as he supplied us the necessary information for the development of our project work.

Moreover, we owe to the Librarian of our university for helping us by allowing the valuable books for lending for home and supplying the reference copies for library study.

Finally, we convey our deep sense of gratitude to our friends for their cooperation and unconditional support. This project work could not have been successfully accomplished without their help.

Dated:

Sandeep Dasgupta and Subrata Mazumder
Bengal Engineering and Science University, Shibpur
Howrah 711 103

Contents

Chapter 1

Introduction

1.1 Problem definition

The subject project work deals with study and analysis of elementary (geometric) properties of digital curves. We have attempted in this work to exploit these properties in order to design efficient algorithms for producing the transformed pattern of a given digital curve after arbitrary affine transformations (translation, rotation, and scaling).

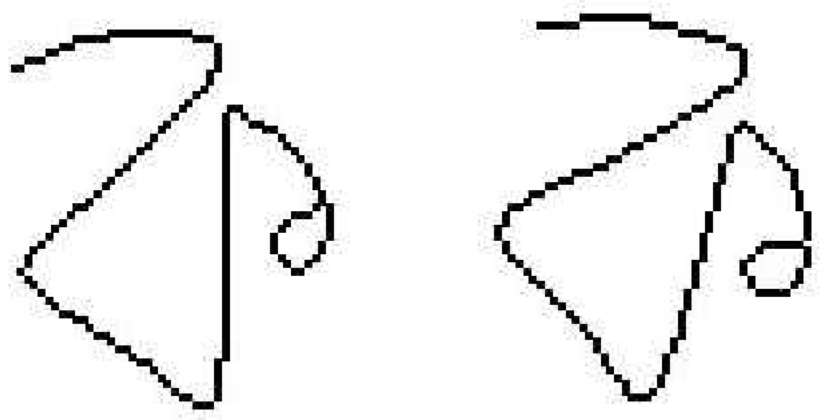


Figure 1.1: A digital curve representing the first consonant in Bengali alphabet that shows the objective of our work. The right curve is the transformed image of the left curve after a clockwise rotation of 15 degrees.

1.2 Preliminaries

Digital Curve: If $P : (x, y)$ is a grid point, then an 8-neighbor of P is any grid point $P' : (i', j')$ with $\max(|i - i'|, |j - j'|) = 1$. A digital curve C is, therefore, an ordered sequence of grid points such that any point in C is an 8-neighbor of its predecessor in the sequence. We consider that a digital curve C is irreducible; that is, removal of any grid point P from C makes C disconnected.

Affine Transformation: A coordinate transformation of the form

$$x' = a_{xx} * x + a_{xy} * y + b_x, \quad y' = a_{yx} * x + a_{yy} * y + b_y$$

is called a-two dimensional affine transformation. Each of the transformed coordinates x' and y' is a linear function of the original coordinates x and y and parameters $a_{(\ )}$ and $b_{(\ )}$ are constants determined by the transformation type. Affine transformation have the general properties that parallel lines are transformed into parallel lines and finite points mapped to finite points. Translation, rotation, scaling, and reflection are examples of two-dimensional affine transformation. Any general two-dimensional affine transformation can always be expressed as composition of these basic transformations. Another affine transformation is the conversion of coordinate descriptions from one reference system to another which can be described as a combination of translation and rotation. An affine transformation involving only rotation, translation, and reflection preserves angles and lengths as well as parallel lines. For these three transformations, the lengths and angles between two lines remain the same after the transformation.

1.3 Scope of work

Given a digital curve and the angle by which it is to be rotated, the present scope of work is to output the rotated curve, while meeting the following challenges.

1. The segment of the curve that is continuous before rotation should remain continuous after the rotation.
2. The number of grid points before rotation should be equal to the number of grid points after the rotation (i.e., there should not be any overlap of grid points).
3. The digital curve should remain irreducible after the rotation (i.e., there should not be any redundant grid points in the curve after the rotation).
4. The shape of the curve (with respect to the difference chain codes) should remain the same after the rotation.
5. Floating point arithmetic should be avoided as much as possible in order to have fast execution of the algorithm.

Chapter 2

Existing Methods

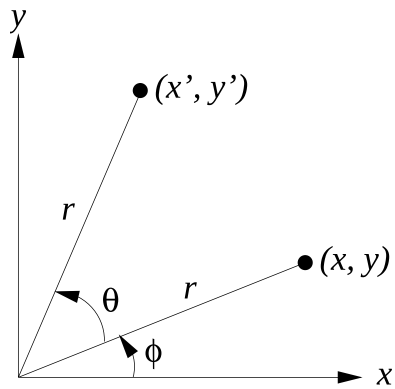


Figure 2.1: Rotation of a point (x, y) about the origin by an angle θ produces the point (x', y'), where $x' = x\cos(\theta) - y\sin(\theta)$ and $y' = x\sin(\theta) + y\cos(\theta)$.

2.1 Using rotation matrix

Existing methods in most of the graphics and image processing softwares use rotation matrix. Rotation of a point $P(x, y)$ in the 2D plane is done about a point (x_c, y_c), which is called the *center of rotation*, to get the rotated point $P'(x', y')$, using the 2D rotation matrix $R(\theta)$ (or, simply R), as follows.

Rotation about the origin:

$$P' = RP = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.1}$$

Rotation about an arbitrary center $P_c : (x_c, y_c)$: The major steps are
1. Translate the object by $T(P_c)$ such that the center P_c coincides to origin $(0, 0)$.
2. Apply rotation with origin as center of rotation.
3. Translate the object by $T(-P_c)$.

The complete equation is, therefore, as follows (see Fig. 2.2).

$$M = T(-P_c)RT(P_c) \tag{2.2}$$

From the above equation, the coordinates of the point $P'(x', y')$ after rotation by angle θ about the point $P_c(x_c, y_c)$ are given by

$$\begin{aligned} x' &= x_c + (x - x_c)\cos\theta - (y - y_c)\sin\theta, \\ y' &= y_c + (x - x_c)\sin\theta + (y - y_c)\cos\theta. \end{aligned} \tag{2.3}$$

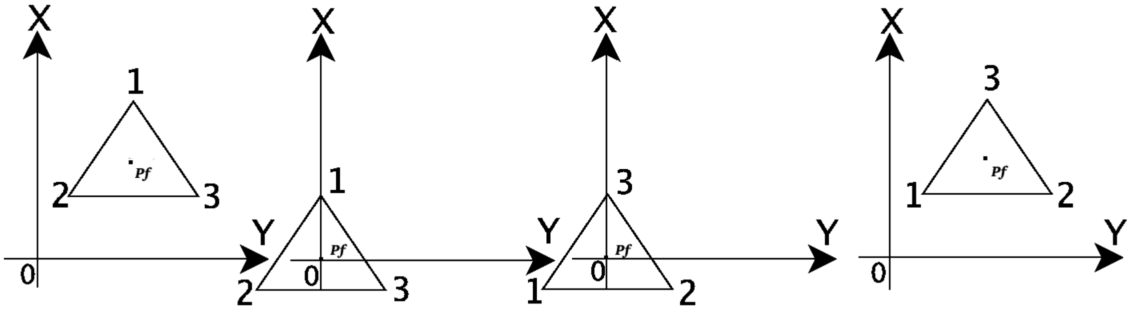


Figure 2.2: Rotation of the digital curve 123 by an angle of 120^o about the point P_c(other than origin).

2.1.1 Algorithm A1

Let C be the given connected digital curve, and P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of grid points of C. Then each point $P_i \in C$ is rotated using the rotation matrix R about the point P_c, as shown in Eqn. 2.3.

2.1.2 Observations and Results

The rotated curve is found to be discontinuous as evident from the results shown in Fig. 2.4.

Causes of discontinuity:

a. **Round-off error.** Due to the usage of floating point arithmetic in the calculation of the coordinates of the rotated point (x', y'), followed by rounding off (for realization in digital plane) as shown below.

$$x' = \quad (\text{int})\Big(0.5 + x_c + (x - x_c)\cos\theta - (y - y_c)\sin\theta\Big)$$
$$y' = \quad (\text{int})\Big(0.5 + y_c + (x - x_c)\sin\theta + (y - y_c)\cos\theta\Big)$$

These round-offs often lead to discontinuity between two points after rotation, which are otherwise continuous before rotation.

b. **Overlapping of grid points.** Due to this, the number of points in the curve after rotation is not equal to that before rotation. And this often leads to discontinuity in the rotated curve.

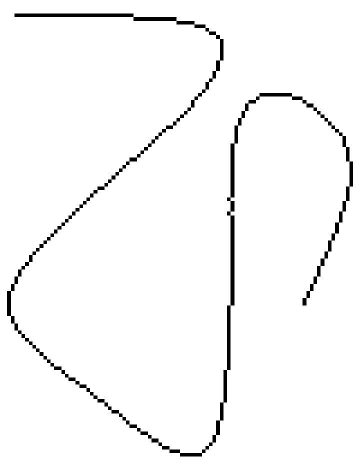


Figure 2.3: Input image.

(a) 10^o (b) 30^o

Figure 2.4: Two rotated images of a test curve (grid points overlapped, not continuous, not irreducible).

Chapter 3

Developed Methods

3.1 Based on Euclidean distance

3.1.1 Algorithm A2

Let C be the given connected digital curve. Let P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of grid points of C.

1. Rotate P_m to P'_m, with respect to P_c, using the rotation matrix R.
2. Set $Q = P'_m$ and $i = m - 1$.
3. Compute the square of the Euclidean distance $d_2^2(P_c, P_i)$ of P_i from P_c.
4. Compute $d_2^2(P_c, Q_j)$, where Q_j corresponds to the jth 8-N point of Q, $\forall\ j \in \{0, 1, \ldots, 7\}$.
5. Choose Q_j such that $|d_2^2(P_c, P_i) - d_2^2(P_c, Q_j)|$ is minimum. Note that this Q_j represents the rotated point corresponding to P_i.
6. Set Q to Q_j.
7. Decrement i.
8. If $i = 0$, then stop; else goto step 4.

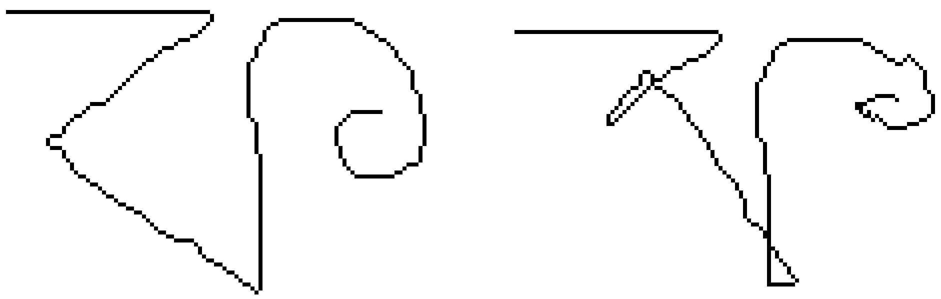
3.1.2 Observations and Results

1. **Ensuring continuity.** The algorithm is based on the fact that if there exist two points P_1 and P_2, one in the 8-neighborhood of the other before rotation, and if P_1 is rotated to P'_1, then the rotated point P'_2 corresponding to P_2 will be one of the 8-N grid points of P'_1. As the

current rotated point will be placed on any one of the 8-N grid points of the previous rotated point, so continuity is maintained.

2. **No floating point arithmetic.** Since for each discrete curve C, only the end point (P_m) is rotated using floating point arithmetic (using rotation matrix), this algorithm limits the usage of floating point arithmetic to a great extent.

3. **Not preserving shape.** This algorithm is based on the assumption that the end point P_m of the connected component $C = \{P_1, P_2, \ldots, P_m\}$ will be placed at the proper position after the rotation. But the calculation of the rotated point using rotation matrix could not ensure such an accuracy. The initial inaccuracy leads to accumulation of inaccuracies in the subsequent calculations of grid points leading to shape deformation.



(a) Input image. (b) 5^o ACLW Rotatation.

Figure 3.1: Results of algorithm A2 on the curve shown in Fig. 3.1(a) (grid points overlapped, continuous curve, not irreducible, shape not maintained).

3.2 Based on Euclidean distance and sign.

Unlike the previous algorithm A2, where Euclidean distance of all the 8-N grid points (of the previous rotated point) are computed in order to determine the location of the current grid point, here Euclidean distance computation is done for at most four grid points at a time, which are selected on the basis of a vector multiplication concept as follows.

If P_1P_2 be a directed line segment from P_1 to P_2, then the signed area A (see Fig. 3.2) of the triangle $\triangle P_1P_2P_3$ determines the position of P_3 w.r.t. P_1P_2 as shown in the following equation.

$$\begin{array}{ll} P_3 \text{ lies to the left of } P_1P_2, & \text{if } A > 0; \\ P_3 \text{ lies to the right of } P_1P_2, & \text{otherwise.} \end{array} \quad (3.1)$$

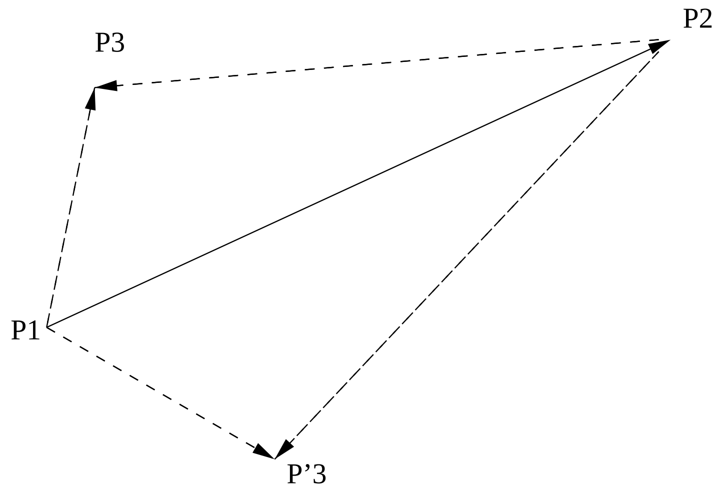


Figure 3.2: Vector multiplication

3.2.1 Algorithm A3

Let C be the given connected digital curve. Let P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of continuous grid points of C.

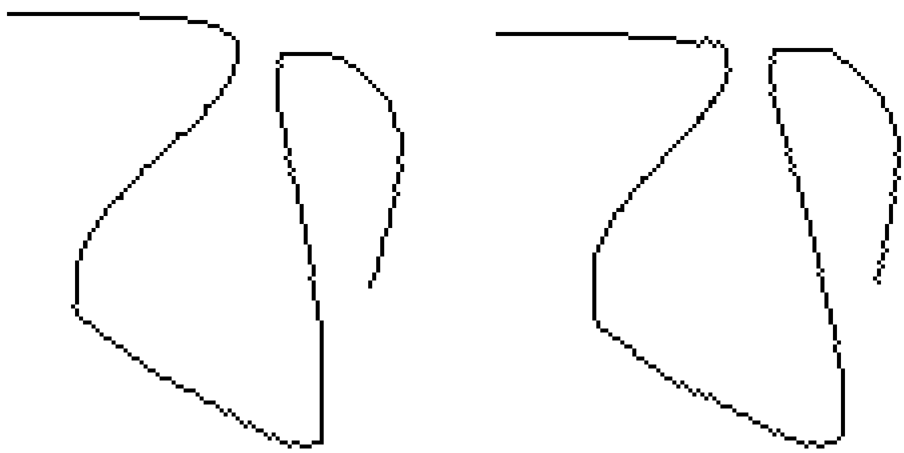
1. Rotate P_m to P'_m, with respect to P_c, using the rotation matrix R. Set $Q = P'_m$ and $i = m-1$.
2. Compute the sign of $\triangle P_c P_{(i+1)} P_i$. Let it be $sign(A)$.
3. Compute $sign(A_j) = sign(\triangle P_c Q Q_j)$, where Q_j corresponds to the jth 8-N point of Q, $\forall\ j \in \{0, 1, \ldots, 7\}$.
4. Select those Q_j which satisfy $sign(A_j) = sign(A)$. Let the selected set of points be $\{Q_k\}$.
5. Compute $d_2^2(P_c, P_i)$.
6. Compute $d_2^2(P_c, Q_k)$, for each Q_k obtained in step 5.
7. Choose a point Q' from $\{Q_k\}$ for which $|d_2^2(P_c, P_i) - d_2^2(P_c, Q')|$ is minimum. Q' represents the rotated point of P_i.
8. Set Q to Q'
9. Decrement i.
10. If $(i = 0)$, then stop; else goto step 2.

3.2.2 Observations and Results

1. **Ensuring continuity.** Explained in Sec. 3.1.2.
2. **No floating point arithmetic.** Explained in Sec. 3.1.2.

3. **Preserving shape.** Though the calculation of the rotated point using rotation matrix may lead to inaccurate determination of initial rotated position, but this initial inaccuracy is corrected by elimination of at least four grid points that are redundant in the sense that the current rotated point can't be any of these points.

4. **An anomaly when $sign(A) = 0$.** In case of horizontal or vertical lines, $sign(A) = 0$. In that case it may happen that $sign(A_j) \neq 0 \; \forall j \in \{0, 1, \ldots, 7\}$. Then it will be impossible (using signed area) to select any point from the 8-N points of Q, while finding the rotated coordinate of P_i. This degenerate case is handled by allowing Euclidean distance computation of all the 8-N grid points of Q (an approach similar to algorithm A2), but this will incorporate the disadvantages of algorithm A2.



(a) 8^o (b) 12^o

Figure 3.3: Results of algorithm A3 (continuous curve, not irreducible, shape maintained).

(a) Input (b) 15^o (c) 20^o

Figure 3.4: Results of algorithm A3 showing its limitations in case of straight line segments.

3.3 Based on arc length

3.3.1 Algorithm A4

Let C be the given connected digital curve. Let P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of continuous grid points of C.

1. For each P_i in C,

   1.1. Make *Circle_points* that stores the circle points (in anticlockwise order) of all radii using Bresenham's algorithm*/

   1.2. Compute $r = d_\top(P_c, P_i)$, where $d_\top(A, B) = \max(|x_a - x_b|, |y_a - y_b|)$ represents isothetic distance between two points $A : (x_a, y_a)$ and $B : (x_b, y_b)$.

   1.3. Compute $arc\_length = r * \theta$.

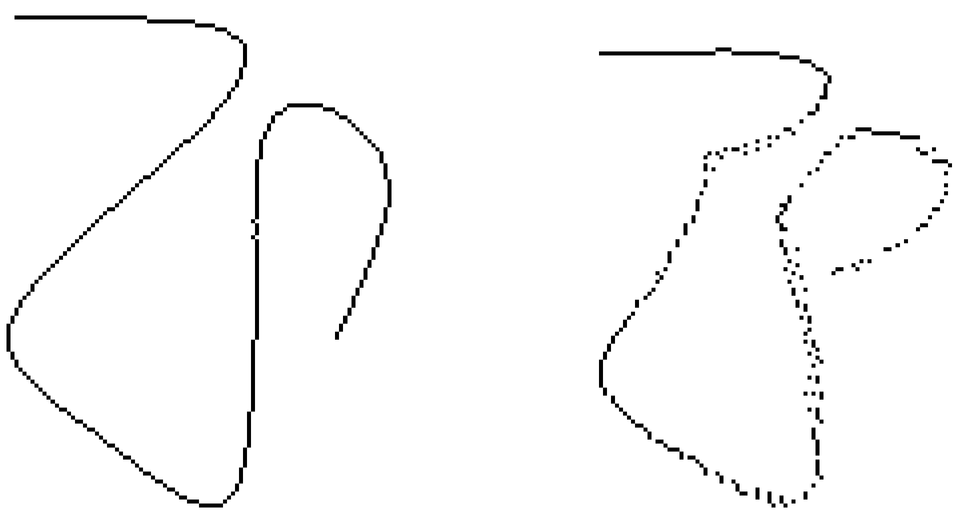
   1.4. Identify the point $P_{j(i)}$ corresponding to P_i in *Circle_points*, as follows.

      a. For each point P_j on $Circle\_points[r]$, compute $d_2^2(P_i, P_j)$.

      b. Return $P_{j(i)}$, such that its corresponding distance d_2^2 is minimum.

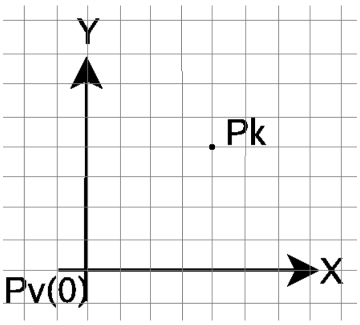
   1.5. If $Circle\_points[r][k] = P_{j(i)}$, then the rotated version of P_i is given by $P_i' = Circle\_points[r][k + arc\_length]$.

3.3.2 Results and Observations

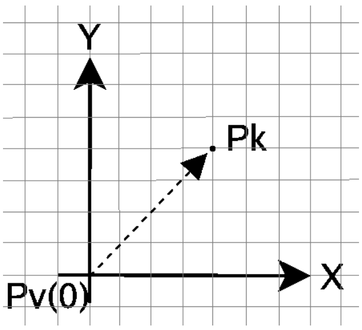


(a) Input image. (c) Output image.

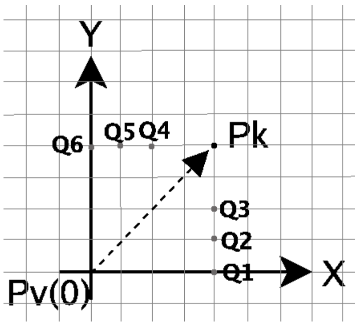
Figure 3.5: Result of algorithm A4. A 5^o rotated image of a bengali alphabet.



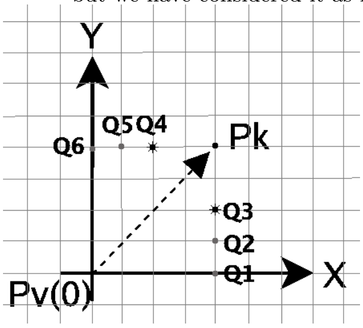
(a) P_k is the point to be rotated.



(b) Compute $S = r * \theta$, where r is the distance of P_k from the center of rotation P_V and θ is the angle of rotation. r should be Euclidean, but we have considered it as isothetic.



(c) The circle with center $P_{v(0)}$ and radius r.



(d) Identify the point P_k on the circle. Here $d_2^2(P_k, Q_j) = e \neq 0$ for any Q_j. Also $d_2^2(P_k, Q_3) = d_2^2(P_k, Q_4)$ is minimum. Hence Q_3 or Q_4 is identified as P_k. Moreover the error e increases with r, and then it become difficult to identify P_k on the circle.

Figure 3.6: An anomaly of the algorithm A4.

3.4 Based on polling using sign check, isothetic distance, and area

Previously by "the distance between two points" we meant the Euclidean distance, i.e., $(x_1 - x_2)^2 + (y_1 - y_2)^2$ is the distance between P_1 and P_2, but from now we introduce the concept of isothetic distance as the distance between two points, i.e., $\max(|x_1 - x_2|, |y_1 - y_2|)$. In all the previous algorithms, a point is rotated based on its position w.r.t the previous point which is rotated most currently, i.e., the rotation decision depends upon a single pixel. But in the subsequent algorithms, polling will be used, where a point will be rotated based on its position w.r.t three previously rotated point.

3.4.1 Algorithm A5

Let C be the given connected digital curve. Let P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of continuous grid points of C.

1. Set constants $Points\_rotated\_by\_matrix = 5$, $Num\_poll = 3$, $Offset = Points\_rotated\_by\_matrix + 1 - Num\_poll = 3$.

2. Rotate the points from P_1 to P_5 to get P'_1 to P'_5, with respect to P_c, using the rotation matrix R. Set $i = 6$.

3. Initialize $M = 0, sign\_selected\_points = [\,], Distance\_selected\_points = [\,]$. /*$sign\_selected\_points$ is used to store points selected by sign check, $Distance\_selected\_points$ is used to store selected points of $sign\_selected\_points$ after isothetic distance check.*/

4. Until $M \leq 2$, do

   4.1 Compute the sign of $A = \triangle P_{(i-(M+offset)}P_{(i-1)}P_i$. Let it be $sign(A)$.

   4.2 Compute the isothetic distance $D_\top(P_{(i-(M+offset)}, P_i)$.

   4.3 Also we compute $sign(A_j) = sign(\triangle P'_{(i-(x+offset))}P'_{(i-1)}Q_j)$, where Q_j corresponds to the jth 8-N point of $P'_{(i-1)}$, $\forall\ j \in \{0, 1, \ldots, 7\}$ and $P'_{(i-(x+offset))}, P'_{(i-1)}$ are the points obtained after rotating $P_{(i-(x+offset))}$ and $P_{(i-1)}$ respectively.

   4.4 Also we compute the isothetic distance $d_\top(P'_{(i-(x+offset))}, Q_j)$, where Q_j corresponds to the jth 8-N point of $P'_{(i-1)}$, $\forall\ j \in \{0, 1, \ldots, 7\}$.

   4.5 Compute $sign\_selected\_points = sign\_selected\_points \cup Q_j$, $\forall\ Q_j$ which satisfy $sign(A_j) = sign(A)$.

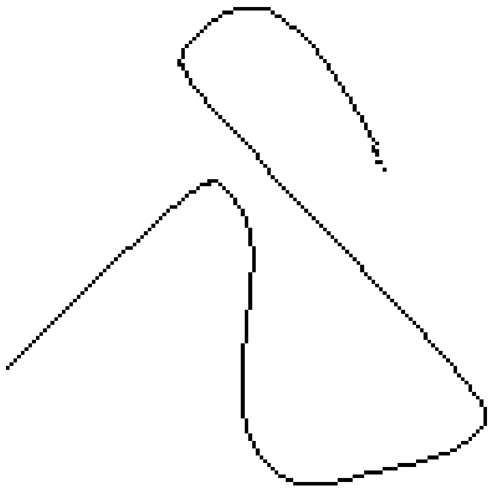
   4.6 Among the points of $sign\_selected\_points$ transfer those points Q_j to $Distance\_selected\_points$ so that the correponding error $|D_\top(P_{(i-(M+offset)}, P_i) - d_\top(P'_{(i-(x+offset))}, Q_j)|$ is minimum.

   4.7 Select a point Q_j from $Distance\_selected\_points$ so that the corresponding error in area $|A - A_j|$ is minimum. Let the selected point be Q_{jM}, increment M, goto step 4.

5. Select that point among Q_{jM} for $0 \leqslant M \leqslant 2$ which wins the vote. Q_{jM} represents the rotated point of P_i. (In case of a tie, where all the three points Q_{jM} for $0 \leqslant M \leqslant 2$, are distinct, the points that encountered all the three checks (sign check, distance check, area check, in order) will be given the higher priority in selection than those which encountered lesser number of checks.)

6. Increment n.

7. If $(i = 0)$, then stop; else goto step 3.



Output image.

Figure 3.7: Result of algorithm A5. A ACLW 60^o rotated image of fig.2.3.

(a) Input (b) 15^o (c) 20^o

Figure 3.8: Results of algorithm A5.

3.4.2 Results and Observations

1. $sign(\triangle) = 0$ **anomaly tackled:** In case of horizontal or vertical lines, $sign(\triangle) = 0$. In that case it may happen that $sign(\triangle_{(D)})! = 0 \; \forall \; D \in \{E, NE, N, NW, W, SW, S, SE\}$. Then it will be impossible (according to algorithm A3) to select any point from the 8-nbd points of Q, while finding the rotated coordinate of P_n. This case is handled by allowing isothetic distance check and area check of all the 8-nbd grid points, which helps to extract the best (and accurate, in most of the cases) results.

2. **Shape not maintained:** It is clear from the figure shown below that if n points are placed horizontally or vertically, then if after rotation they are placed diagonally, then there is an increment in real distance by $(n\sqrt{2} - n)$.

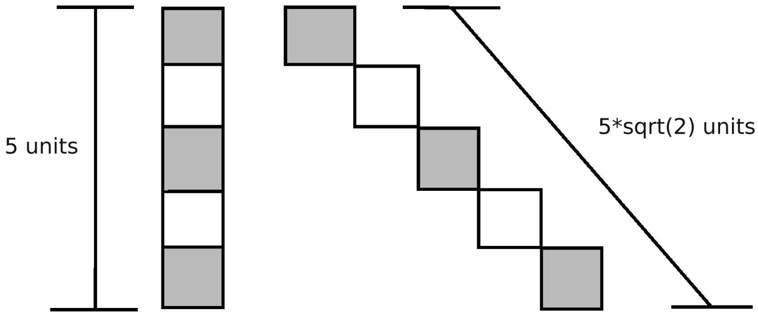
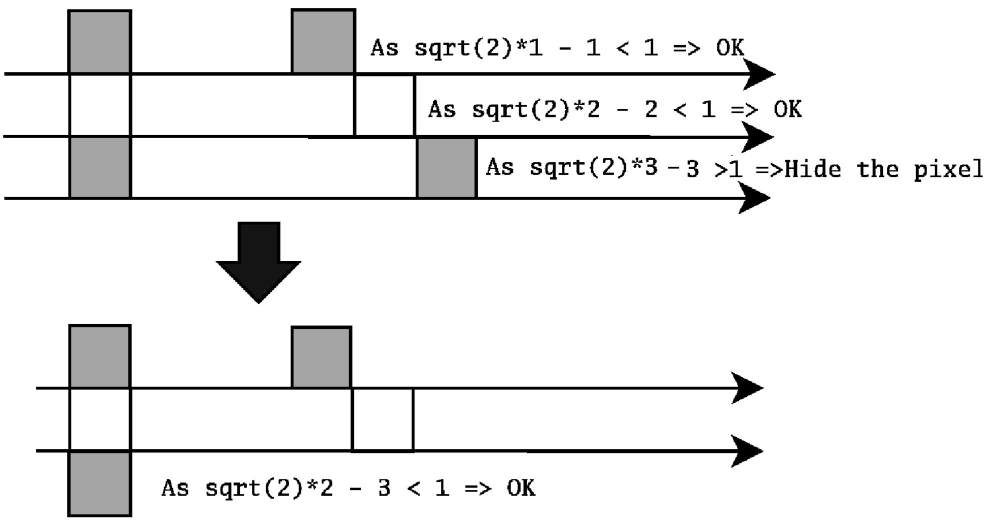


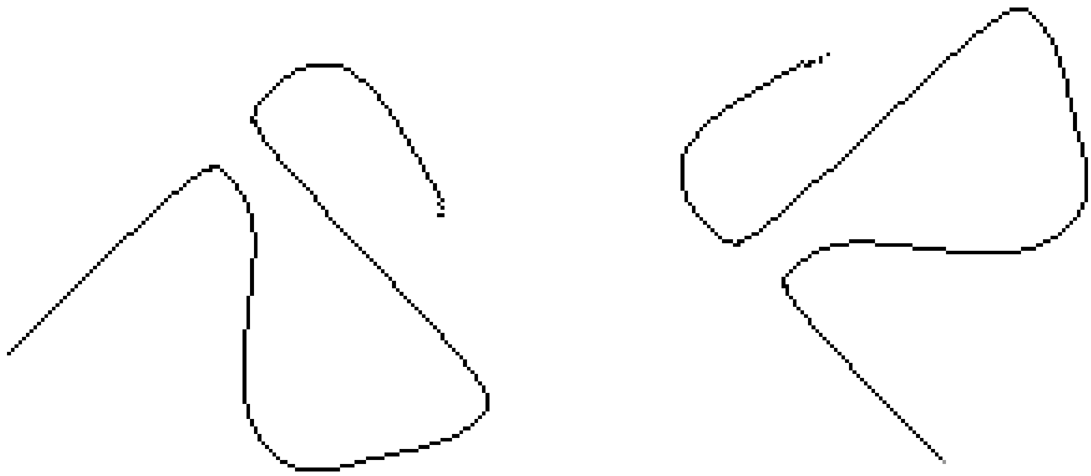
Figure 3.9: Cause of non-maintenace of shape.

Shape Maintenance Algorithm

1. Set $in\_count\_dis1 = 0, in\_count\_dis\sqrt{2} = 0, out\_count\_dis1 = 0, out\_count\_dis\sqrt{2} = 0.$
2. In the input image, if the next point scanned is placed horizontally or vertically, increment in_count_dis1, otherwise increment $in\_count\_dis\sqrt{2}$ (when the next point scanned lie diagonally).
3. Similarly in the output image, if the next point is placed (after rotation) horizontally or vertically, increment out_count_dis1, otherwise increment $out\_count\_dis\sqrt{2}$ (when the next point lie diagonally).
4. Every placement of point in the output image is followed by a consistency check , $|(in\_count\_dis1 + \sqrt{2} * in\_count\_dis\sqrt{2}) - (out\_count\_dis1 + \sqrt{2} * out\_count\_dis\sqrt{2})| < TOLERANCE(= 1)$. Violation of this check prevents points to be outputted to the output image until the normal condition is restored.

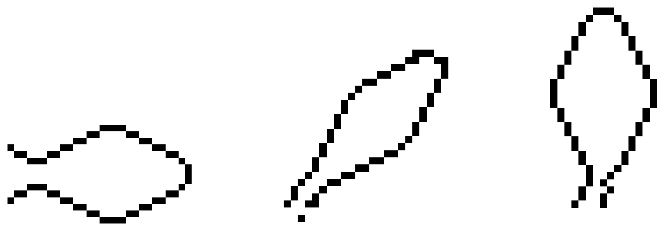


Results of Shape Maintenance Algorithm



(a) 30^o (c) 120^o

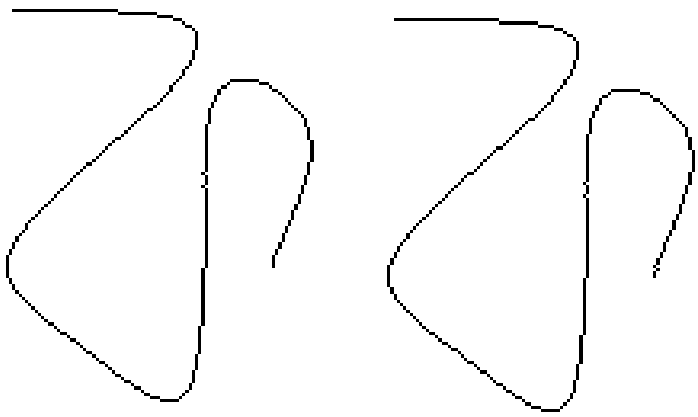
Figure 3.10: Results of algorithm A5 with shape maintenance algorithm.Two ACLW rotated images of fig.2.3



(a) Input (b) 15^o (c) 20^o

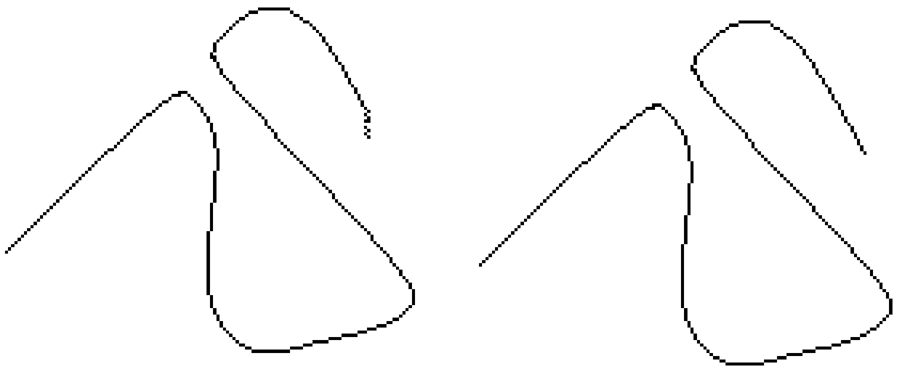
Figure 3.11: Results of algorithm A5 with shape maintenance algorithm.

3. **Variation of rotation of the curve with the angle of rotation is small:**



(a) 10^o (c) 20^o

Figure 3.12: Two ACLW rotated images of fig.2.3. No variation in shape in the range 10^o-20^o



(a) 25^o (c) 55^o

Figure 3.13: Two ACLW rotated images of fig.2.3. No variation in shape in the range 25^o-55^o

3.5 Based on algorithm A5 with some initialisation points.

3.5.1 Description

The inability of algorithm A5 to rotate digital curves at all possible angles (especially at small angles) could be accounted for inaccuracy in the placement of initial five points by floating point arithmetic using rotation matrix. This problem could be thought to be alleviated by appending five initial points at the beginning of the digital curve and rotating those five points using rotation matrix (instead of rotating initial five points on the digital curve using rotation matrix).

3.5.2 Algorithm A6

Let C be the given connected digital curve. Let P_c be the given point of rotation. Let $\{P_1, P_2, \ldots, P_m\}$ be the ordered set of continuous grid points of C.

1. Set constants $Points\_rotated\_by\_matrix = 5$, $Num\_poll = 3$, $Offset = Points\_rotated\_by\_matrix + 1 - Num\_poll = 3$.

2. Append points $\{P_{(-4)}, P_{(-3)}, P_{(-2)}, P_{(-1)}, P_{(0)}\}$ at the beginning of the curve $\{P_1, P_2, P_3, \ldots, P_n\}$. Then $P_{(-4)}$ to $P_{(0)}$ is rotated to $P'_{(-4)}$ to $P'_{(0)}$, with respect to P_c, using the rotation matrix $R_{(\theta)}$. Set i = 1.

3. Initialize $M = 0, sign\_selected\_points = [\,], Distance\_selected\_points = [\,]$. /*$sign\_selected\_points$ is used to store points selected by sign check, $Distance\_selected\_points$ is used to store selected points of $sign\_selected\_points$ after isothetic distance check.*/

4. Until $M \leq 2$, do

   4.1 Compute the sign of $A = \triangle P_{(i-(M+offset)} P_{(i-1)} P_i$. Let it be $sign(A)$.

   4.2 Compute the isothetic distance $D_\top(P_{(i-(M+offset)}, P_i)$.

   4.3 Also we compute $sign(A_j) = sign(\triangle P'_{(i-(x+offset))} P'_{(i-1)} Q_j)$, where Q_j corresponds to the jth 8-N point of $P'_{(i-1)}$, $\forall\, j \in \{0, 1, \ldots, 7\}$ and $P'_{(i-(x+offset))}, P'_{(i-1)}$ are the points obtained after rotating $P_{(i-(x+offset))}$ and $P_{(i-1)}$ respectively.

   4.4 Also we compute the isothetic distance $d_\top(P'_{(i-(x+offset))}, Q_j)$, where Q_j corresponds to the jth 8-N point of $P'_{(i-1)}$, $\forall\, j \in \{0, 1, \ldots, 7\}$.

   4.5 Compute $sign\_selected\_points = sign\_selected\_points \cup Q_j$, $\forall\, Q_j$ which satisfy $sign(A_j) = sign(A)$.

   4.6 Among the points of $sign\_selected\_points$ transfer those points Q_j to $Distance\_selected\_points$ so that the correponding error $|D_\top(P_{(i-(M+offset)}, P_i) - d_\top(P'_{(i-(x+offset))}, Q_j)|$ is minimum.

   4.7 Select a point Q_j from $Distance\_selected\_points$ so that the corresponding error in area $|A - A_j|$ is minimum. Let the selected point be Q_{jM}, increment M, goto step 4.

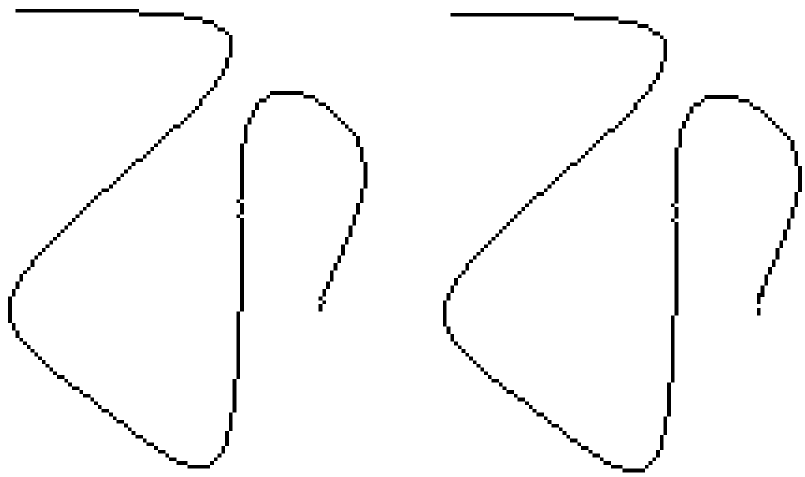
5. Select that point among Q_{jM} for $0 \leqslant M \leqslant 2$ which wins the vote. Q_{jM} represents the rotated point of P_i. (In case of a tie, where all the three points Q_{jM} for $0 \leqslant M \leqslant 2$, are distinct, the points that encountered all the three checks (sign check, distance check, area check, in order) will be given the higher priority in selection than those which encountered lesser number of checks.)

6. Increment n.

7. If $(i = 0)$, then stop; else goto step 3.

3.5.3 Observations and Results

The inability to rotate at all possible angles(especially at small angles) could be accounted for inability to rotate initial five imaginary points by floating point arithmetic using rotation matrix as the set of imaginary points form a straight line and straight lines are unable to rotate at small angles ($< 5^o - 10^o$).



(a) 20^o (c) 30^o

Figure 3.14: Results of algorithm A6. Two ACLW rotated images of fig.2.3

Chapter 4

Conclusion and Future Scope

Table 4.1: Comparision of algorithms A1 to A6 on basis of image criteria.

	A1	A2	A3	A4	A5	A6
Maintenance of shape	Yes	No	Yes	No	Yes	No
Continuity maintenance	No	Yes	Yes	No	Yes	Yes
Limited usage of floating arithmatic	No	Yes	Yes	Yes	Yes	Yes
Overlapping of grid point	Yes	Yes	Yes	Yes	No	No
Irreducible	No	No	No	No	No	No

From the above comparison it is clear that algorithm A5 yeild the best results among the others. But it suffered from the sole disadvantage that it yeilds results for multiples of 45^o, but there is no change in the shape in the range $45^o * n$ - $45^o * (n+1), n = 0, 1, \ldots$.

Future scope of work

The future scope of work involves further modification of algorithm A6 so as to ensure the following :

1. Better handling of curve segments with $sign(A) = 0$.
2. Eliminating redundant grid points while maintaining the continuity of the curve.
3. The overlapping of grid points should be totally eliminated.
4. Modification of algorithm A5 to ensure rotation of curve for all angles of rotation.

Bibliography

[1] Donald Hearn, M Pauline Baker, *Computer Graphics*-2^{nd} *Edition.* , Prentice-Hall of India Private ltd.

[2] James D. Folley, Andries Van Dam, Steven K. Feiner, Jhone F. Hughes, *Computer Graphics-Principles and practics*, Addison-Wesley Publishing Company.

[3] Adobe system, Inc., *PS language tutorial and cookbook*, Addsion-Wesley Reading, MA, 1985.

[4] Beatty, J.C., K.S.Booth, eds., *Tutorial : Computer Graphics -Second Edition.* IEEE comp. soc. press, Silver spring, MD, 1982.