CS560-460 Spring2016

**Assignment 2**

(Due on March 7, 2016 by 11:00pm)

**Topics: 2D transformation, clipping, and filling**

Part A: Theory part; Part B: Programming part.

**(Part A) Theory Part (48%)**

(1) (7%) Given two perpendicular vectors P0P1 and P0P2, derive a transformation matrix which can transform object description from xy coordinate system to x’y’ coordinate system.

y

**P1=(x1, y1)**

**P2=(x2, y2)**

**P0 =(x0, y0)**

x

0

x’

y’

The final transformation matrix should be

(2) (7%) Let line L have a y intercept (0, b) and an angle of inclination θ (with respect to the x axis). Describe the transformation M which reflects an arbitrary point P in the triangle around the line QQ’ (e.g., reflect P to P’). Note that the point Q is known, and Q’ is the reflection of Q around line L.

**Q’**

x

0

**B**

**(0, b)**

**y**

**θ**

**P**

**L**

**P’**

**Q**

The transformation be represented by the matrix

Such that

Leaving the final transformation to be

(3) (8%):

Given a triangle (ABC), where A=(2,0), B=(4,0), C=(2,4). Apply the following two transformation sequences (a) and (b) to the original triangle ABC individually, and generate two new triangles A1B1C1 and A2B2C2. Plot the two new triangles on the figure.

(a) T(-1, 0) S (1/2, 1/4) R (90°) Rf(y,x)

(b) R (90°) T (-1, 0) S (1/2, 1/4) Rf(y, -x)

Note: The transformations in the sequence are performed in the right-to-left order. Rf(y,x) refers to the reflection around a line y=x; Rf(y, -x) refers to the reflection around a line y= -x.

C1

A2

C

A

B

Y

X

B2

B1

A1

C2

(4) (6%) Given a triangle ABC in a window, after the window-to-viewport mapping, ABC is mapped to A’B’C’ in the viewport 1, and to A”B”C” in the viewport 2. Derive the coordinates of A’, B’, C’,A”, B”, C”, and draw the triangle A’B’C’ and A”B”C”

A = (3, 5); B=(4, 5.5); C=(5, 4)

A’ = (1.5, 5.25); B’ = (2, 5.625); C’ = (2.5, 4.5)

A” = (4.75, 6.5); B” = (5.5, 6.75); C” = (6.25, 6)

**Viewport 1 Viewport 2**

B”

**Window**

**7**

A”

C”

B’

A’

**B**

**6**

**6**

**5**

**A**

C’

**C**

**3**

**2**

**7**

**4**

**3**

**1**

**2**

**6**

**2**

(5) (8%) Clip the polygon P1,…, P8 in the following figure against the rectangular clipping window.

(a)Using the Sutherland-Hodgman algorithm, give the list of output vertices and plot your result.

P1…P8 => P1…P8 => C4,C1,P4,C2,C3,P6,P7,P8,P1 => B1,C1,P4,C2,B1,C9,P7,P8,C7 => C1,P4,C2,B1,C9,P7,C10

(b) If using the Weiler-Atherton algorithm instead, give the list of output vertices and plot your results.

Polygon Verts: P1,P2,P3,C1,P4,C2,P5,P6,C9,P7,C10,P8

Clip Verts: B1,C2,C1,B2,B3,C10,B4,C9

Final Verts: C1,P4,C2,C1; C9,P7,C10,B4,C9;

B4

B3

B2

B1

C10

C9

C8

C7

C6

C3

C4

C2

C1

P1

P2

P3

P4

P6

P7

P8

P5

Clipping Window

Polygon

P1

P2

P3

P4

P6

P7

P8

P5

Clipping Window

Polygon

(6) (7%): We can use the general scan-line polygon-fill algorithm to fill the following polygon. Based on the odd-parity rule, indicate the odd-even segments on the dashed-lines.

Key: Dashed = Even, Blue Filled = Odd

(7) (5%) Use the Weiler-Atherton algorithm to clip the polygon against another polygon as follow, plot the result. Indicate the turning points that switch the subject polygon edges to the clip polygon edges. Number the order for edges that you visited.

E9

E8

E7

E6

T4

E5

T3

E4

E3

T2

E2

E1

T1

Clip polygon

Subject polygon

Starting vertex