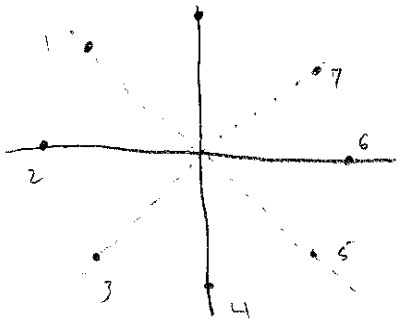
#1

First Pixel:

Plot $(x_0, y_0) = (0, r)$

relative to $(0, r)$, plot the other 7 octant pixels



vaguely — can't tell what you coded

* consider start and End Theta. If it falls in between, then plot.

End Pixel: (relative to the P_{k+1} calculated from the second to last pixel)

so, from 2nd to last pixel:

increment x;

midpt alg doesn't say how last pixel gotten

```
then if(p<0)
    P_k+1 += 2 * circPt.getX() + 1;
else
{
    dec. y;
    P_k+1 += (circPt.getX() - circPt.getY()) + 1;
```

CODE

Q. Write a program to create a window using GLUT and a display callback function and start the GLUT event loop

```
#include <GL/glut.h>
void init (void)
{
  glClearColor (1.0, 1.0, 1.0, 0.0); // Set display window color to white
  glMatrixMode (GL_PROJECTION); // 2D projection matrix
  glLoadIdentity (); // Assign identity matrix to current matrix
  gluOrtho2D (xmin, xmax, ymin, ymax); // Coordinates of display window
}
void draw (void) // display callback function
{
  //
  //
  //      drawing
  //
  //
  //
  glFlush (); // Forces all OpenGL routines to be executed completely
}
void main (int argc, char** argv)
{
  glutInit (&argc, argv); // Initialize GLUT
  glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
  glutInitWindowPosition (x0, y0);
  glutInitWindowSize (w, h);
  glutCreateWindow ("title");
  init ();
  glutDisplayFunc (draw);
  glutMainLoop ();
}
```

(3) line $y = -3m + 20.$

Since $|m| > 1$
then we will step along y and calculate x

we have 2 choices: (5,4) or (6,4)

(5,5)
(5,4) (6,4)

Start point (0,20)
end point (5,5)

the decision function is

$p = 2(\Delta x) - \Delta y$

$= 2(5 - (-15))$
$= 2(20) = 35$

Since $p > 0$

Then we choose to draw (6,4).

need to swap roles of x & y do decision fn

Brute Force Way

Also could look at

$y = -3x + 20$

$4 = -3x + 20$

$\frac{16}{3} = x$

So $x = 5\frac{1}{3}$ and $(5\frac{1}{3}, 4)$ is closer to

pt would plot → (5,4)

MIDTERM PRACTICE #4

new x_{max} given by Prof.

$addr(x,y) = addr(0,0) + y(100+1) + x$

$addr(10,50) = addr(0,0) + 50(101) + 10$

$addr(10,50) = addr(0,0) + 5060$

↖ THIS ANSWER IS AS GOOD AS YOU CAN GIVE [redacted]

Comp Graphics

Ikai Lan
Asami Enomoto
Marvin Gouw

5. Ellipse midpoint algorithm decision function

$$f(x,y) = r_y^2 x^2 + r_x^2 y^2 - r_x^2 r_y^2$$

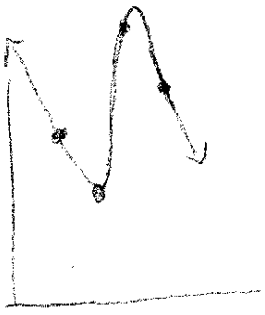
where r_y is the radius along the y-axis and
r_x is the radius along the x-axis

$f(x,y) > 0$, (x,y) resides outside the ellipse
< 0, (x,y) is inside the ellipse
$= 0$, (x,y) is on the ellipse

PROBLEM 6

DEGREE 3 POLYNOMIAL

POINTS { (2,9) (4,6) (6.22) (7,13) }



USING LANGRANGE

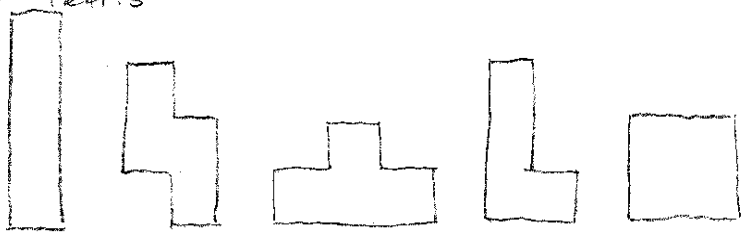
$$y = \frac{(x-x_2)(x-x_3)(x-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)}\,y_1 + \frac{(x-x_1)(x-x_3)(x-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)}\,y_2 + \frac{(x-x_1)(x-x_2)(x-x_4)}{(x_3-x_1)(x_3-x_2)(x_3-x_4)}\,y_3 + \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_4-x_1)(x_4-x_2)(x_4-x_3)}\,y_4$$

PLUGIN POINTS { }

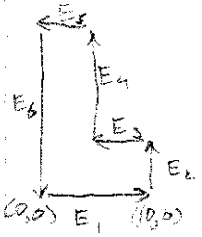
$$y = \frac{(x-4)(x-6)(x-7)}{(2-4)(2-6)(2-7)}(9) + \frac{(x-2)(x-6)(x-7)}{(4-2)(4-6)(4-7)}(6) + \frac{(x-2)(x-4)(x-7)}{(6-2)(6-4)(6-7)}22 + \frac{(x-2)(x-4)(x-6)}{(7-2)(7-4)(7-6)}(13)$$

* YOU CAN SIMPLIFY THE FRACTIONS EX. $\frac{13}{5 \cdot 3 \cdot 1} = \frac{13}{15}$

⑦ Tetris



convex concave concave concave convex



1. Make the polygon into edge vectors. Edge vectors can be determined by vertexes of the polygon. For example the edge vector for E_1 is $(10,0)$.

$E_1 = (10,0) - (0,0)$ $\quad E_k = V_{k+1} - V_k$

$E_1 = (10,0)$

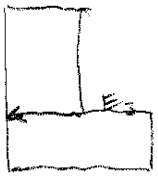
2. Then calculate the cross product of successive edge vectors in order around the polygon.

$E_1 \times E_2 > 0$

$E_2 \times E_3 > 0$

$E_3 \times E_4 < 0$

$E_5 \times E_6 > 0$



$E_3 \times E_4$ is the only cross product with a different sign. Extend E_3 until it intersects E_6. This will create 2 convex polygons.

Deepthi Korrani
James Wong
Peter Nguyen

(8) GLubyte bitshape[20] = {
0x1c, 0x00, 0x1c ..... }

glPixelStorei (GL-UNPACK-ALIGNMENT, 1);
glRasterPos2i (30, 40);
glBitMap (9, 10, 0.0, 0.0, 20.0, 15.0, bitshape)

(1) bitshape is an Array that stores the bit patterns at each row.

(2) glpixelstorei → set the storage mode for bitMap.

Ref: page 144

(9) GL DISPLAY LISTS

```
name = glGenList(1);
glNewList(name, GL_COMPILE OR GL_COMPILE_AND_EXECUTE)
    .
    .
    .
    //SOME SET OF GL COMMANDS
    .
    .
    .
glEndList();
glCallList(name);
```

// THE PURPOSE OF USING DISPLAY LISTS IS TO CREATE AN ID SO THAT WE CAN EXECUTE CODE MANY TIMES LATER JUST BY CALLING THE ID.

10.

```
glBlendFunc(src, dest);
glColor4f(r, g, b, alpha)
glEnable(GL_BLEND);
    // DRAWING FUNCTIONS
glDisable(GL_BLEND);
result = Vsrc * Bg + Vdest * Fg;
```

symbolic constants that say what 4-vector to use

dot product

$V_{src} = (w_x, w_y, w_z, w_a)$

$V_{dest} = (w'_x, w'_y, w'_z, w'_a)$

colors to be blended

soft-fill: soften fill colors at edges
tint-fill: repaint area with mixture of tint

In the book soft-fill = tint-fill = technique to fill region whose boundaries have shaded (for instance, during antialiasing)