Graphics Editor using OPENGL

Sourav Parmar (86511933) Priyanshu Pandey(21081358)
Computer and Information Science Engineering, University of Florida

What is "Graphics Editor" ?

Graphics Editor is a OPENGL based two dimensional drawing tool that can be used to draw various shapes.

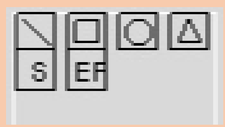
Motivation for "Graphics Editor" ?

- Simulation of MS Paint using OPENGL.
- Provide visual representation of various computer graphics Algorithms.

Feature List

Toolbox
- Pencil
- Eraser
- Brush
- Floodfill
- Spray
- Convex Hull

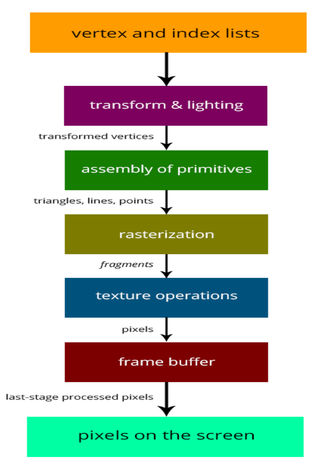
Shapebox
- Line
- Rectangle
- Circle
- Triangle
- Bezier Curve
- Ellipse



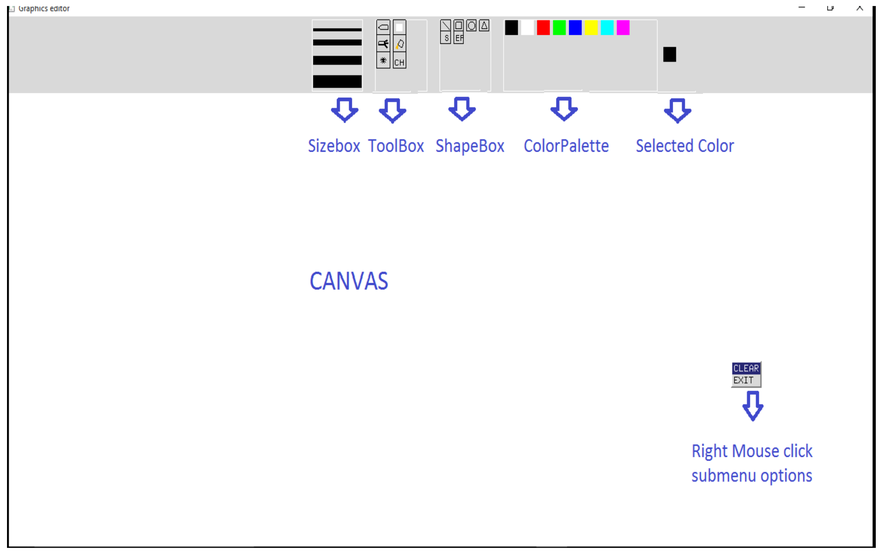
Colorpalette

Sizebox

Fixed Function Pipeline OPENGL

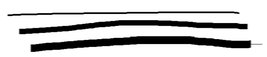


Results

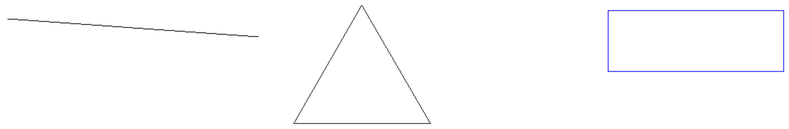


Pencil

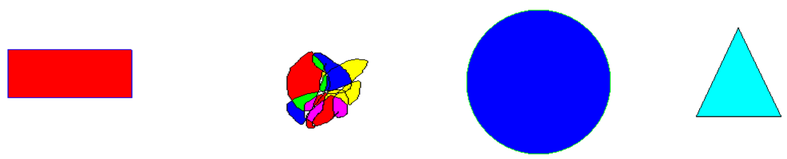
Brush



Triangle / Line /Rectangle

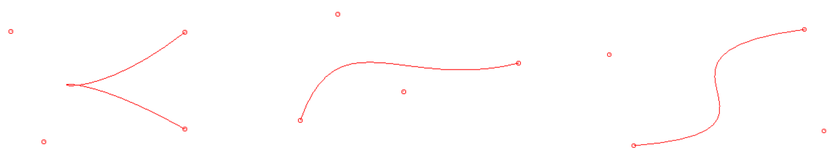


Flood Fill

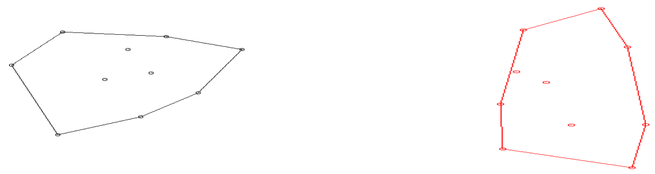


Circle / Ellipse

Cubic Bezier Curve



Convex Hull



Implementation Details

Tool and Shapes

PENCIL /BRUSH/Eraser
- Initialize (X_{new} ,Y_{new}) and (X_{old}, Y_{old}) to -1 when Pencil tool is selected
- For every Mouse_Motion_CB
  - (X_{new} ,Y_{new}) are assigned the new (x,y) coordinates.
  - GL_LINES is drawn between (X_{new} ,Y_{new}) and (X_{old}, Y_{old})
  - (X_{new} ,Y_{new}) are copied to (X_{old}, Y_{old})
- Eraser selected color is same as background color
- Pencil has fix Line Width of one pixel.

Spray
- Fifty randomly generated pixels are colored in the vicinity of the current mouse position.

Rectangle/Triangle/Line
- Old and new coordinates are stored in (X_{new} ,Y_{new}) and (X_{old}, Y_{old}).
- For **Rectangle** GL_LINE_LOOP is completed by connecting the following points : (X_{old}, Y_{old}), (X_{old}, Y_{new}) , (X_{new}, Y_{old}) , (X_{new}, Y_{new})
- For **Triangle**, compute height for equilateral triangle (h). GL_LINE_LOOP is completed by points (X_{old}, Y_{old}) , (X_{new}, Y_{old}), ((X_{old} + X_{new})/2, Y_{old} +h)
- For **Line** GL_LINE_STRIP is drawn between (X_{new} ,Y_{new}) and (X_{old}, Y_{old}).

Circle/Ellipse
- Old and new coordinates are stored in (X_{new} ,Y_{new}) and (X_{old}, Y_{old})
- Draw circle using center (X_{old}, Y_{old}) and radius abs(X_{new} - X_{old})
- Draw ellipse using center (X_{old}, Y_{old}), rh abs(X_{new} - X_{old}) and rv abs(Y_{new} - Y_{old})

Bezier Curve
- Select four control points using mouse click
- Find points of cubic Bezier curve using Bernstein cubic polynomials
- Connect points using GL_LINES

Convex Hull
- Select ten points using mouse click
- Find points of convex hull using Graham Scan Algorithm
- Connect points using GL_LINES

Floodfill
- Start at a point inside a region
- Replace a specified interior color (old color) with fill color
- Fill the 4-connected until all interior points being replaced
- 4 way method is implemented with std::stack

Handler

Paintpencil() /paintbrush()
- setselectedcolor()
- glLineWidth(selectedsize)
- glBegin(GL_LINES)
- glVertex2f(X_{old}, Y_{old})
- glVertex2f (X_{new} ,Y_{new})
- glEnd()

Spray(float x , float y)
- glBegin(GL_POINTS)
- glVertex2f(XRand , YRand)
- glEnd()

Drawrect() / drawtriangle() / drawline()
- glLineWidth(selectedsize)
- setselectedcolor()
- glBegin(GL_LINE_STRIP)
- glVertex2f: (X_{old}, Y_{old})
- glVertex2f(X_{new}, Y_{new})
- glEnd()

Draw_circle()/draw_ellipse()
- Draw circle using circle midpoint algorithm
- Draw ellipse using ellipse midpoint algorithm

findbiezerpoint() :
Loop till t <1
- glBegin(GL_LINES)
- glVertex2f(xold, yold)
- glVertex2f(xbeiz, ybeiz)
- glEnd() xold = xbeiz yold = ybeiz

drawGrahamscan ()
- Using Graham Scan algorithm Find convex hull point
- Connect point in final stack with glBegin(GL_LINES)

Floodfill(x,y,newcolor)
- Push(x,y) on stack
- Until stack is not empty
- Get top x',y' pair of stack
- If pixelcolor (x',y')!= newcolor && (x',y') not marked visited
- setPixelColor(x', y', newcolor)
- stack.pusx(x'+1 ,y')
- stack.push(x' ,y'+1)
- stack.push(x' ,y'-1)
- stack.push(x'-1 ,y')