Lab Exercise Date 13-02-2019

1. Write a C program for polygon clipping using Sutherland Hodgeman Polygon Clipping.
   Input:
   Polygon: (100,150), (200,250), (300,200)
   Clipping Area: (150,150), (150,200), (200,200), (200,150)

2. Write a C program for polygon clipping using Weiler Atherton fill area clipping
   Polygon: (100,150), (200,250), (300,200)
   Clipping Area: (100,300), (300,300), (200,100)

3. Write a C program for drawing a curve using Hermite interpolation for control points (10,10) and (100,100) and parametric derivatives (slope of the curve) at control points 3 and 4 respectively.

Instructions

Suther land Hodgeman Polygon Clipping:

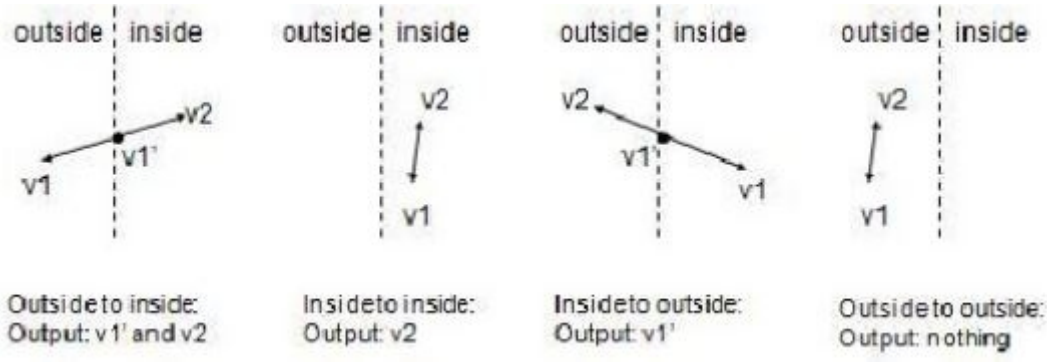
The Sutherland Hodgeman Polygon Clipping Algorithm works by extending each lineof the convex clip polygon in turn and selecting only vertices from the subject polygon that is on the visible side. It accepts an ordered sequence of vertices v1, v2, v3... vn and puts out a set of vertices defining the clipped polygon.

- Clip the polygon by processing the polygon boundary as a whole against each window.
- Process all polygon vertices against each clip rectangle boundary in turn.
- First clip the polygon against the left boundary to produce a new sequence of vertices.
- The new sequence of vertices is then successive passed to right, bottom and top boundary clipper.
- At each step a new sequence of output vertices is generated and passed to the next boundary clipper

Rule for each edge clipper:

Input each edge (vertex pair) successively.
Output is a new list of vertices
Each edge goes through 4 clippers.

- If first input vertex is outside, and second is inside, output the intersection and the second vertex
- If both input vertices are inside, then just output second vertex
- If first input vertex is inside, and second is outside, output is the intersection
- If both vertices are outside, output is nothing.
- The last vertex in the list is always added to the first vertex



Weiler Atherton Polygon Clipping:

It Clips a "Subject Polygon" to a "Clip Polygon" giving one or more output polygons that lie entirely inside the clip polygon. Both polygons can be of any shape.

- Set up the vertex list for the subject and clip polygons. (The ordering should be such that as you move down each list the inside of the polygon is always on the right side.)
- Compute all intersection points between subject polygon and clip polygon edges; insert them into each polygon's list; mark as intersection points.

  Mark those intersection points in which the subject polygon edge is moving from outside of the clip polygon to inside. (outside/inside tests on subject polygon edge endpoints.)

- Do until all intersection points have been visited:

  Traverse the subject polygon list until you find a non-visited intersection point; output the point to a new output list.

  Make the subject polygon list be the active list.

  Do until a vertex is revisited:

  Get the next vertex from the active list and output.

  If vertex is an intersection point, make the other list active.

  End the current output polygon list.

Hemite interpolating:

A Hemite spline is an interpolating piecewise cubic polynomial with a specified tangent at each control point. Unlike the natural cubic splines, Hermite splints can be adjusted locally because each curve section is only dependent on its endpoint constraints.

1. Enter two hermite points (x1,y1) = (10,10) and (x2,y2) = (100,100)
2. Enter and slope of the curve at those points as slope1 = 3 and slope2 = 4
3. Obtain each coordinate position as follows along the curve for 0<=u<=1 at the interval of .001

Calculate x = (2*u*u*u-3*u*u+1)*x1+(-2*u*u*u+3*u*u)*x2+(u*u*u-2*u*u+u)* slope1*x1+(u*u*u-u*u)* slope2*x2;

Calculate y = (2*u*u*u-3*u*u+1)*y1+(-2*u*u*u+3*u*u)*y2+(u*u*u-2*u*u+u)* slope1*x1+(u*u*u-u*u)* slope2*x2;