Computer Graphics

Assignment- 2

Aryan Mohan

500092142

Batch- 2

Q1. Write a short note on Filling Polygons Clipping Lines.

Filling polygons and clipping lines are two common tasks in computer graphics.

Filling polygons refers to the process of coloring the interior of a closed shape, defined by a set of vertices. There are several algorithms for polygon filling, including scanline, boundary, and flood fill algorithms. These algorithms typically involve iterating over the pixels within the boundaries of the polygon, and determining whether each pixel should be colored or left blank based on its position relative to the polygon.

Clipping lines, on the other hand, refers to the process of removing portions of a line that lie outside of a specified region, such as a window or viewport. This is often used in 2D graphics to ensure that only the parts of a line that are visible to the viewer are drawn. The most commonly used algorithm for line clipping is the Cohen-Sutherland algorithm, which involves classifying each endpoint of the line as being inside or outside of the clipping region, and then removing any portions of the line that lie outside.

In summary, filling polygons and clipping lines are important tasks in computer graphics that involve manipulating shapes and lines to create visually appealing and accurate images.

Q2. Explain Cyrus Beck's algorithm with an example.

Cyrus-Beck algorithm is a line-clipping algorithm used to remove the parts of a line segment that lie outside of a given rectangular region or any other convex polygonal window. It is a parametric algorithm that works by finding the intersection points of the line segment with the edges of the clipping region.

Here's how the Cyrus-Beck algorithm works:

- Define the line segment to be clipped as a vector P, where P = Q2 - Q1. Q1 and Q2 are the endpoints of the line segment.
- Define the edges of the clipping region as a set of vectors Ei, where i = 1 to n, and n is the number of edges of the clipping region.

- For each edge Ei, calculate the dot product of P with Ei, and the dot product of Q1 with Ei.
- Calculate the parameter values tE and tL for each edge, where tE = -(Q1 dot Ei) / (P dot Ei), and tL = -(Q2 dot Ei) / (P dot Ei).
- For each edge, if P dot Ei is less than or equal to zero, the line segment is either entirely inside or parallel to the edge, and both tE and tL should be set to zero. Otherwise, if P dot Ei is greater than zero, the line segment intersects the edge, and tE and tL should be calculated as described above.
- Calculate the maximum value of tE and the minimum value of tL.
- If the maximum value of tE is less than or equal to the minimum value of tL, the line segment is inside the clipping region and should be drawn. Otherwise, the line segment lies outside the clipping region and should be discarded.

Let's consider an example to illustrate the Cyrus-Beck algorithm. Suppose we have a line segment defined by endpoints Q1 = (2, 3) and Q2 = (6, 7), and a rectangular clipping region with vertices at (1, 1), (1, 5), (5, 5), and (5, 1).

- The vector P is calculated as P = Q2 - Q1 = (6 - 2, 7 - 3) = (4, 4).
- The edges of the clipping region are defined by the vectors E1 = (0, -4), E2 = (4, 0), E3 = (0, 4), and E4 = (-4, 0).
- For each edge, we calculate the dot product of P with Ei and Q1 with Ei:
  P dot E1 = (4, 4) dot (0, -4) = -16 Q1 dot E1 = (2, 3) dot (0, -4) = -12
  P dot E2 = (4, 4) dot (4, 0) = 16 Q1 dot E2 = (2, 3) dot (4, 0) = 8
  P dot E3 = (4, 4) dot (0, 4) = 16 Q1 dot E3 = (2, 3) dot (0, 4) = 12
  P dot E4 = (4, 4) dot (-4, 0) = -16 Q1 dot E4 = (2, 3) dot (-4, 0) = -10
- We calculate tE and tL for each edge:
  tE1 = -(-12) /

Q3. Explain the Liang-Barsky algorithm with an example.

The Liang-Barsky algorithm is a line-clipping algorithm used to remove the parts of a line segment that lie outside of a given rectangular region or any other convex polygonal window. It is a parametric algorithm that works by finding the intersection points of the line segment with the edges of the clipping region.

Here's how the Liang-Barsky algorithm works:

1. Define the line segment to be clipped as a vector P, where P = Q2 - Q1. Q1 and Q2 are the endpoints of the line segment.

2. Define the edges of the clipping region as four parameters: Left, Right, Bottom, and Top, which represent the coordinates of the left, right, bottom, and top edges of the clipping region.
3. Calculate the values of the four parameters u1, u2, v1, and v2, where u1 = 0, u2 = 1, v1 = 0, and v2 = 1.
4. For each edge of the clipping region, calculate the values of p and q, where:
    - For the left edge: p = -Px and q = Qx – Left
    - For the right edge: p = Px and q = Right – Qx
    - For the bottom edge: p = -Py and q = Qy – Bottom
    - For the top edge: p = Py and q = Top – Qy

    Px and Py are the x and y coordinates of the vector P, and Qx and Qy are the x and y coordinates of the endpoint Q.

5. For each edge, check whether p is equal to zero. If p is not equal to zero, calculate the values of r1 and r2, where r1 = q / p and r2 = (q + p) / p. If p is equal to zero and q is less than zero, the line segment is outside the clipping region and should be discarded. Otherwise, the line segment is either entirely inside or parallel to the edge and should be retained.
6. Calculate the new values of u1, u2, v1, and v2 by taking the maximum and minimum values of r1 and r2, where applicable.
7. If u1 is greater than v1 or u2 is less than v2, the line segment is outside the clipping region and should be discarded. Otherwise, calculate the new endpoints of the line segment as Q1' = Q1 + u1 * P and Q2' = Q1 + u2 * P, and draw the clipped line segment.

Let's consider an example to illustrate the Liang-Barsky algorithm. Suppose we have a line segment defined by endpoints Q1 = (2, 3) and Q2 = (6, 7), and a rectangular clipping region with vertices at (1, 1), (1, 5), (5, 5), and (5, 1).

1. The vector P is calculated as P = Q2 - Q1 = (6 - 2, 7 - 3) = (4, 4).
2. The edges of the clipping region are defined by the parameters Left = 1, Right = 5, Bottom = 1, and Top = 5.
3. The initial values of u1, u2, v1, and v2 are u1 = 0, u2 = 1, v1 = 0, and v2 = 1.
4. We calculate the values of p and q for each edge:
    - For the left edge: p = -4 and q = 2 - 1 = 1
    - For the right edge:

Q4. Elaborate on “Clipping Polygons”.

Clipping polygons is the process of removing or retaining parts of a polygon that lie inside or outside of a given clipping region or window. It involves determining which vertices of the polygon are inside the clipping region and which ones are outside, and then creating a new polygon that only includes the vertices that are inside the region.

The clipping region is typically defined by a set of edges that form a closed polygonal boundary. The edges of the clipping region can be either straight or curved, and they can intersect each other at any angle.

There are several algorithms that can be used for clipping polygons, including the Sutherland-Hodgman algorithm, the Weiler-Atherton algorithm, and the Greiner-Hormann algorithm. These algorithms work by breaking down the original polygon into smaller parts, testing each part against the edges of the clipping region, and then combining the parts that lie inside the region to form the final clipped polygon.

One of the most widely used algorithms for polygon clipping is the Sutherland-Hodgman algorithm. This algorithm works by iteratively clipping the polygon against each of the edges of the clipping region, one edge at a time.

Here's how the Sutherland-Hodgman algorithm works:

1. Define the polygon to be clipped as a set of vertices in clockwise or counterclockwise order.
2. Define the clipping region as a set of edges that form a closed polygonal boundary.
3. For each edge of the clipping region, clip the polygon against that edge using the following steps:
    a. Define a list of output vertices, initially empty.
    b. Set the previous vertex to be the last vertex in the polygon, and loop over all vertices in the polygon.
    c. If the current vertex is inside the clipping region, add it to the list of output vertices.
    d. If the current vertex is outside the clipping region, calculate the intersection point between the edge and the line segment formed by the previous vertex and current vertex. Add the intersection point to the list of output vertices.
    e. Set the previous vertex to be the current vertex.
    f. Repeat steps c through e for all vertices in the polygon.
    g. Set the polygon to be the list of output vertices.

4. The final polygon is the result of clipping the original polygon against all edges of the clipping region.

Clipping polygons is used in computer graphics to cull parts of an image that are outside of the viewing area, to create complex shapes by combining multiple polygons, and to render 3D objects using 2D graphics primitives.