Homework Explanation

1 Converting Bezier Curves into Line Segments

To simplify the representation of Bezier curves, a flatness error metric was used to approximate the curve as a series of straight lines. The flatness error ensures that the segments remain within a tolerable deviation from the true curve.

Flatness Formula with Parameter t

For a quadratic Bezier curve defined by control points P_0, P_1, and P_2, a point on the curve for a parameter $t \in [0, 1]$ is given by:

$$B(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2$$

Where:

- $B(t)$ is the point on the curve at parameter t,
- P_0, P_1, P_2 are the control points.

To calculate the flatness, we check the maximum distance of sampled points on the curve from the line segment connecting P_0 and P_2. The maximum distance is given by:

$$F = \max_{t \in [0,1]} \text{dist}(B(t), \text{line } P_0P_2)$$

Where $\text{dist}(B(t), \text{line } P_0P_2)$ is the perpendicular distance from $B(t)$ to the line P_0P_2.

Procedure to Calculate Flatness

- **Generate Sample Points**: Evaluate the curve at regular intervals $t = 0.01, 0.02, \ldots, 0.99$.

$$B(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2$$

- **Compute Distance**: For each point $B(t)$, calculate the perpendicular distance to the line P_0P_2:

$$\text{dist}(B(t), \text{line } P_0P_2) = \frac{|(x_2 - x_1)(y_1 - y_t) - (x_1 - x_t)(y_2 - y_1)|}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

- **Find Maximum Distance**: The flatness F is the maximum of these distances.

If F exceeds a threshold, the curve is recursively subdivided.

Subdivision Procedure

Quadratic Bezier Subdivision

Given control points P_0, P_1, P_2:

1. Compute the midpoints:
$$M_0 = \frac{P_0 + P_1}{2}, \quad M_1 = \frac{P_1 + P_2}{2}$$
2. Compute the midpoint of M_0 and M_1:
$$Q = \frac{M_0 + M_1}{2}$$
3. The original curve is divided into two curves:
   - From P_0 to Q with control points P_0, M_0, Q.
   - From Q to P_2 with control points Q, M_1, P_2.

Cubic Bezier Subdivision

Given control points P_0, P_1, P_2, P_3:

1. Compute the midpoints:
$$M_0 = \frac{P_0 + P_1}{2}, \quad M_1 = \frac{P_1 + P_2}{2}, \quad M_2 = \frac{P_2 + P_3}{2}$$
2. Compute the midpoints of the midpoints:
$$N_0 = \frac{M_0 + M_1}{2}, \quad N_1 = \frac{M_1 + M_2}{2}$$
3. Compute the midpoint of N_0 and N_1:
$$Q = \frac{N_0 + N_1}{2}$$
4. The original curve is divided into two curves:
   - From P_0 to Q with control points P_0, M_0, N_0, Q.
   - From Q to P_3 with control points Q, N_1, M_2, P_3.

2 Calculating Holes and Outer Borders Using Signed Area

The signed area method was used to distinguish between outer borders and holes in a 2D shape. The signed area of a polygon is calculated as follows:

Signed Area Formula

$$A = \frac{1}{2} \sum_{i=1}^{n} (x_i y_{i+1} - y_i x_{i+1})$$

Where:

- (x_i, y_i) are the vertices of the polygon, and $(x_{n+1}, y_{n+1}) = (x_1, y_1)$.
- A positive area indicates a counterclockwise (outer) border.
- A negative area indicates a clockwise (hole) border.

3 Edge Normal Computation for Consecutive Segments

3.1 Computation of Edge Normals

We calculate the edge normals using the following approach:

1. For each point in the shape, identify the previous, current, and next points. Let prev $= (x_{\text{prev}}, y_{\text{prev}})$, curr $= (x_{\text{curr}}, y_{\text{curr}})$, and next $= (x_{\text{next}}, y_{\text{next}})$.

2. Compute the edge vectors from the current point to the previous point, and from the current point to the next point:

$$\text{edgeX1} = x_{\text{curr}} - x_{\text{prev}}, \quad \text{edgeY1} = y_{\text{curr}} - y_{\text{prev}},$$

$$\text{edgeX2} = x_{\text{next}} - x_{\text{curr}}, \quad \text{edgeY2} = y_{\text{next}} - y_{\text{curr}}.$$

3. Normalize the edge vectors to calculate the edge normals. The normal vector for the first edge is:

$$\hat{n}_1 = \left(\frac{\text{edgeY1}}{|\text{edge1}|}, -\frac{\text{edgeX1}}{|\text{edge1}|} \right),$$

and the normal vector for the second edge is:

$$\hat{n}_2 = \left(\frac{\text{edgeY2}}{|\text{edge2}|}, -\frac{\text{edgeX2}}{|\text{edge2}|} \right),$$

where |edge1| and |edge2| are the lengths of the edge vectors:

$$|\text{edge1}| = \sqrt{\text{edgeX1}^2 + \text{edgeY1}^2}, \quad |\text{edge2}| = \sqrt{\text{edgeX2}^2 + \text{edgeY2}^2}.$$

4. Average the two normalized normal vectors to get a smoother transition between the two consecutive segments:

$$\hat{n}_{\text{avg}} = \left(\frac{\hat{n}_1 + \hat{n}_2}{2} \right).$$

5. Normalize the resulting average normal to ensure it is a unit vector:

$$|\hat{n}_{\text{avg}}| = \sqrt{\hat{n}_{\text{avg,x}}^2 + \hat{n}_{\text{avg,y}}^2},$$

and then normalize:

$$\hat{n}_{\text{avg}} = \frac{\hat{n}_{\text{avg}}}{|\hat{n}_{\text{avg}}|}.$$

6. The final normal can be used to offset the point outward or inward based on the desired direction (outer border or hole). The offset point is given by:

$$\text{offsetPoint} = (x_{\text{curr}} + \hat{n}_{\text{avg,x}} \times \text{radius}, \quad y_{\text{curr}} + \hat{n}_{\text{avg,y}} \times \text{radius}),$$

The above steps ensure that the normals are correctly computed for consecutive edges and used to generate the appropriate offset points for both outer borders and holes.

4 Scanline Tool

Scanlines are horizontal lines intersecting a shape. For each scanline $y = y_k$, we compute the x-coordinates of the intersections with the edges of the shape. From these intersection points, we determine the segments of the scanline that lie inside the shape.

- **Intersections with Scanline**:

  A scanline $y = y_k$ intersects an edge of the shape defined by two endpoints $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ if and only if:

  $$\min(y_1, y_2) \leq y_k \leq \max(y_1, y_2).$$

  The x-coordinate of the intersection is calculated using the line equation:

  $$x = x_1 + \frac{y_k - y_1}{y_2 - y_1}(x_2 - x_1).$$

  This gives a set of intersection points $(x_1, y_k), (x_2, y_k), \ldots, (x_n, y_k)$ for the scanline $y = y_k$.

- **Sorting and Removing Duplicates**:

  The intersection points are sorted in ascending order of x-coordinates:

  $$x_1 < x_2 < \cdots < x_n.$$

  Any duplicate x-coordinates (e.g., caused by overlapping edges) are removed, leaving a set of unique intersection points.

- **Determining Inside Segments**:

  To determine which segments of the scanline lie inside the shape, we leverage the **Even-Odd Rule**:

  - Start from the leftmost intersection point (x_1, y_k).
  - Alternate between "outside" and "inside" the shape as you move between consecutive intersection points.

  If the sorted intersection points are $(x_1, y_k), (x_2, y_k), \ldots, (x_n, y_k)$, the scanline segments that are inside the shape are:

  $$\text{Segments} = \{[(x_1, y_k) \to (x_2, y_k)], [(x_3, y_k) \to (x_4, y_k)], \ldots, [(x_{n-1}, y_k) \to (x_n, y_k)]\}.$$

  Here, the segments alternate between being "inside" and "outside" the shape.

- **Final Representation**:

  A scanline $y = y_k$ is represented as a collection of line segments:

  $$\bigcup_{i=1,3,\ldots} [(x_i, y_k) \to (x_{i+1}, y_k)].$$

  These segments correspond to the portions of the scanline that lie inside the shape.