**Line Clipping**

**Outline**
- Clipping problem
- Point-in-window test
- Line clipping algorithms
  - Cohen-Sutherland algorithm
  - Parametric algorithm

**Clipping Problem**
- Remove an item or part of an item that is outside a window
- The item can be:
  - Point, line, or polygon
- The window can be:
  - Rectangle
  - Arbitrary polygon
- Application:
  - Display part of a large drawing on screen

**Point-in-Window Test**
- How to define a rectangular window?
  - The coordinates of two corners of the window
    - Bottom left & top right
    - Top left & Bottom right
- How to determine whether a point is inside a window?
  - \( l \leq x \leq r \)
  - \( b \leq y \leq t \)

**Line Clipping**
- Both endpoints inside:
  - Accept
- One endpoint inside; one outside:
  - Partially accept
  - Need calculate intersection
- Both endpoints outside:
  - Cannot decide
  - Part of the line may be inside

**Cohen-Sutherland Algorithm**
- Divide plane into 9 regions
- Each region is assigned a 4-bit code
  - 1st bit: \( y > t \)
  - 2nd bit: \( y < b \)
  - 3rd bit: \( x > r \)
  - 4th bit: \( x < l \)
- Check the codes for the two endpoints of a line
Trivially Accept

- Both endpoints are in region 0000
- Both are in the window
- Whole line is inside
- Trivially accept

Trivially Reject

- Codes for the two endpoints are 1001 & 1010
- 1001 AND 1010 = 1000
- Both endpoints are above the window
- Trivially reject

Partially Accept

- 0001 AND 0000 = 0000
- Cannot decide
- 0001 XOR 0000 = 0001
- Intersect with left edge
- Intersection with left splits it into 2 parts
  - Left part has code 0001 & 0001
    - Reject
  - Right part has code 0000 & 0000
    - Accept

Non-trivial Reject

- 1000 AND 0010 = 0000
- Cannot decide
- 1000 XOR 0010 = 1010
- Intersect with top & right edges
- Intersection with top splits it into 2 parts
  - Top part has codes 1000 & 1010
    - Reject
  - Bottom part has code 0010 & 0010
    - Reject

Recursive Case

- 1000 AND 0101 = 0000
- Cannot decide
- 1000 XOR 0101 = 1101
- Intersect with top, bottom, & left edges
- Intersection with top splits it into 2 parts
  - Top part are rejected
  - Bottom part has codes 0000 & 0101
    - Cannot decide
    - Do it again

Pseudocode for Cohen-Sutherland Algorithm

```c
void LineClipping(code endpoint1, code endpoint2) {
  if (endpoint1 == 0000 && endpoint2 == 0000) {
    Accept and draw the current line segment;
    Return;
  } else if (endpoint1 & endpoint2 != 0000) {
    Reject the current line segment;
    Return;
  } else {
    Compute endpoint1 XOR endpoint2;
    Select a boundary for intersection calculation;
    intersect1 = code at one side of boundary;
    intersect2 = code at the other side of boundary;
    LineClipping(endpoint1, intersect1);
    LineClipping(intersect2, endpoint2);
  }
}
```
Parametric Algorithm
• Use the parametric line function:
  • $x = x_0 + k \cdot (x_1-x_0)$
  • $y = y_0 + k \cdot (y_1-y_0)$
• Find the range of $k$ that is inside the window:
  • $k_{in} \leq k \leq k_{out}$
• No need for recursion, but always requires 4 intersection calculations

Intersection Calculation
• 4 intersections exist in general cases
• At intersections:
  • $l = x_0 + k_l \cdot (x_1-x_0)$
  • $r = x_0 + k_r \cdot (x_1-x_0)$
  • $t = y_0 + k_t \cdot (y_1-y_0)$
  • $b = y_0 + k_b \cdot (y_1-y_0)$
• Therefore:
  • $k_l = (l - x_0) / (x_1-x_0)$
  • $k_r = (r - x_0) / (x_1-x_0)$
  • $k_t = (t - y_0) / (y_1-y_0)$
  • $k_b = (b - y_0) / (y_1-y_0)$

Potential Entering/Leaving Point
• 2 intersections for the horizontal zone
  • One is horizontal entering point
    • $H_{in} = \min(k_l, k_r)$
    • The other is horizontal leaving point
    • $H_{out} = \max(k_l, k_r)$
  • 2 intersections for the vertical zone
    • $V_{in} = \min(k_t, k_b)$
    • $V_{out} = \max(k_t, k_b)$

Real Entering/Leaving Point
• The real entering point:
  • Should be the larger of the two entering points
  • Should also $\geq 0$
    • $k_{in} = \max(0, H_{in}, V_{in})$
• Similarly, the real leaving point:
    • $k_{out} = \min(1, H_{out}, V_{out})$

Outside Cases
• If $k_{in} > k_{out}$
  • The line segment leaves the window before it enters
  • Whole line segment is outside the window
  • Fully reject

Inside Portion
• If $k_{in} < k_{out}$
  • Range $[k_{in}, k_{out}]$ is inside the window
  • The coordinates of the endpoints of the inside portion can be calculated using:
    • $x_s = x_0 + k_{in} \cdot (x_1-x_0)$
    • $y_s = y_0 + k_{in} \cdot (y_1-y_0)$
    • $x_e = x_0 + k_{out} \cdot (x_1-x_0)$
    • $y_e = y_0 + k_{out} \cdot (y_1-y_0)$
Special Cases

• How about horizontal/vertical lines?
  • Edges parallel to the line can be ignored

• Two situations:
  • Line is outside the window
  • The inside portion can be calculated using two intersections