might decrease, or neither, or segment is discarded.

When each clipping line/plane is processed, no might increase or no

Initially, \( n_1 = 0 \) and \( n_2 = 1 \).

must be stored and updated 4 times (6 times for 3D).

To process one line segment, two parameters values I

Fairly easy, assigned for homework.

2D/3D Line-Clip Algorithm:

Easy to understand, it extends to 3D.

Bottom, Right, Left, bits = 1.

Chip an entire line away when its endpoints share any one or more Top,

the Top clipping line, etc.

Two endpoints share the "Top" bit = 1 when they are both above

2D/3D Line Clipping HB 6-7 and 7-11, Cohen-Sutherland:

CSci 422/502 Lecture 33

University at Albany Computer Science Dept.
Bresenham's algorithm rasterizes line segments. How can we rasterize a

the pixel positions that might be affected by the primitive.

Rastorization: The generation of colored fragments corresponding to

2D if that hadn't been done earlier.

What happens after clipping? First, do perspective division to convert to

culling region edge or vertex with a polygon face.

New vertex: Intersection of a polygon edge with a clipping region face, or a

vertices can be added.

When a polygon is clipped, original vertices can be removed and new

We emphasize the pipelineable Sutherland-Hodgman algorithm.

Polygon Fill Area or 3D polygon face clipping. HB 6-8 and 7-11:
It's easiest when the polygon is convex. See Fig. 4-23 and text.

Bresenham-like techniques are applied to each relevant polygon side.

To calculate the x-coordinates for the ENDS of the NEXT row,

**Scan-Line Algorithms** locate affected pixels row-by-row.
Intuitively, a convex set has no "holes" or separate components. We can also say "$S$ is convex" can be expressed: For all $\mathbf{p}_1$ and $\mathbf{p}_2 \in S$ then $\mathbf{p} \in S$. If $\mathbf{p} \in S$ then $\mathbf{p}_1 \mathbf{p}_2$ is in $S$. If $\mathbf{p}_1 \mathbf{p}_2$ is in $S$ then the entire line segment $\mathbf{p}_1 \mathbf{p}_2$ is in $S$. A set $S$ is convex means that whenever points $\mathbf{p}_1$ and $\mathbf{p}_2 \in S$, the entire straight line segment $\mathbf{p}_1 \mathbf{p}_2$ is in $S$.
So, line segment $P_1P_2$ is in both, in other words, $P_1$ is in $S^1 \cup S^2$.

because $S^1$ is convex and $S^2$ is convex.

- $P_1$ and $P_2$ are in $S^2$. So line segment $P_1P_2$ is in $S^2$.
- $P_1$ and $P_2$ are in $S^1$. So line segment $P_1P_2$ is in $S^1$.

Whenever $P_1$ and $P_2 \in S^1 \cup S^2$.

The intersection $(S^1 \cup S^2) = (S^1 \cap S^2)$ is convex whenever two sets $S^1$ and $S^2$ are both convex.

What's cool about convex sets is that the intersection of two convex sets is convex. This means that the intersection of $S^1$ with the clipping window $S^2$ (which is a convex region) is also convex.

University at Albany Computer Science Dept.
4n ≥ 2
huge number of vertices
neuts n and (2) has a
disconnected compo-
(1) has a huge number
gets with 4n + 4 vertices
ected non-convex poly-
sction of these 2 con-

Example: The Inter-

\[ n \]

is in 2

intersection

fingers each

have n

51 and 52

2

1

2

1
\[
\begin{align*}
\left\{ \begin{array}{c}
0 & < \alpha + z \cdot C + B \cdot x + y \cdot z + \beta \\
\text{for which } & Ax + Bx + Cy = d
\end{array} \right. \\
\end{align*}
\]

A \textbf{halfspace} is the set of points in or on one side of a plane.

(They include their boundaries, topologically closed.)

\textbf{Every convex polygon or polygonal region is the intersection of halfspaces.}

\textbf{So, clipping a filled non-convex polygon is a problem because many}

\textbf{disconnected components may result.}

\textbf{It is computationally cumbersome to figure which sequences of vertices}

\textbf{should be filled polygons.}

\textbf{It is computationally cumbersome to figure which sequences of vertices}

\textbf{should be filled polygons.}

\textbf{It is computationally cumbersome to figure which sequences of vertices}

\textbf{should be filled polygons.}

\textbf{It is computationally cumbersome to figure which sequences of vertices}

\textbf{should be filled polygons.}
Intersection of a polygon edge with a clipping plane. Output vertices are sometimes input vertices and other times the input plus repetition. Polytetra edge goes: (Out or In) → (Out or In).

Halftone, so it's a convex polygon.

Input: Sequence of points describing the polygon after clipping the particular plane. It is the intersection of the input polygon with a halftone plane. Halftone rules in figure 6-26 apply to successive pairs of vertices in the halftone plane, so it’s a convex polygon.

Output: Sequence of points describing the convex polygon in positive (CW) order. Repeat the first.

Use 4 stages, one for each clipping plane. For EACH stage:

See HB 6-8. Idea for 2D:

Graphics card is a pipeline algorithm. So, it's a good choice in an OpenGL engine in a Sutherland-Hodgeman clipping (1) works only on convex polygons and (2)
Right and Top Clipper Outputs are the same.

Bottom Clipper Output:

Left Clipper Output:

desired output:

\[(1', 2', 2'', 2')\]