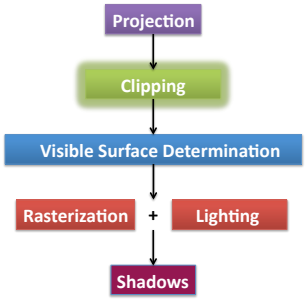
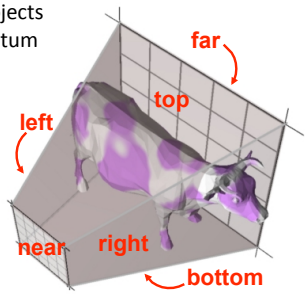
Clipping

Rasterization Pipeline Reminder



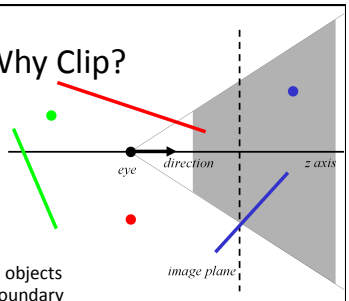
Clipping

- Eliminate portions of objects outside the viewing frustum
- View Frustum
  - boundaries of the image plane projected in 3D
  - a near & far clipping plane
- User may define additional clipping planes



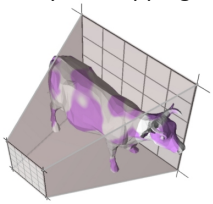
Why Clip?

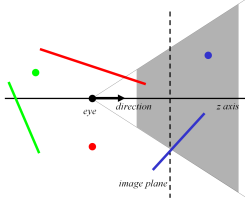
- Avoid degeneracies
  - Don't draw stuff behind the eye
  - Avoid division by 0 and overflow
- Efficiency
  - Don't waste time on objects outside the image boundary
- Other graphics applications (often non-convex)
  - Hidden-surface removal, Shadows, Picking, Binning, CSG (Boolean) operations (2D & 3D)



Clipping Strategies

- Don't clip (and hope for the best)
- Clip on-the-fly during rasterization
- Analytical clipping: alter input geometry



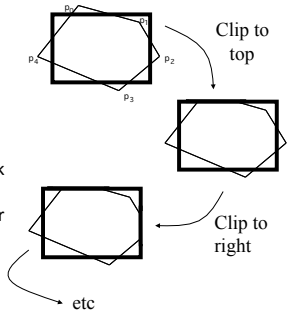


Clipping Summary

- It's the process of finding the exact part of a polygon lying inside the view volume
- To maintain consistency, clipping of a polygon should result in a polygon, not a sequence of partially unconnected lines
- We will first look at 2 different 2D solutions and then extend one to 3D

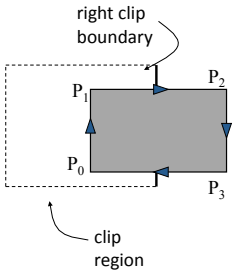
Sutherland-Hodgman Algorithm

- Clip the polygon against each boundary of the clip region successively
- Result is possibly NULL if polygon is outside
- Can be generalised to work for any polygonal clip region, not just rectangular



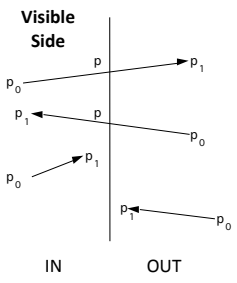
Clipping To A Region

- To find the new polygon
  - iterate through each of the polygon edges and construct a new sequence of points
  - starting with an empty sequence
  - for each edge there are 4 possible cases to consider



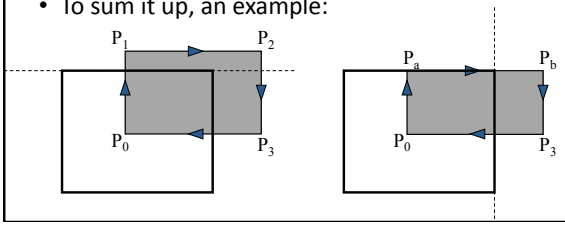
Clipping a polygon edge against the boundary

- Given an edge P0,P1 we have 4 case. It can be:
  - entering the clip region, add P and P1
  - leaving the region, add only P
  - entirely outside, do nothing
  - entirely inside, add only P1
- Where P is the point of intersection



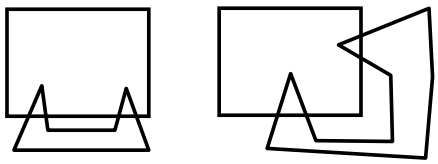
Still the Sutherland-Hodgman

- We can determine which of the 4 cases and also the point of intersection with just if statements
- To sum it up, an example:



Problems with Sutherland-Hodgman

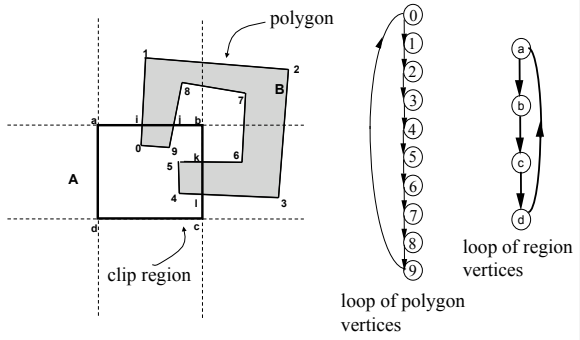
- What if you clip these?



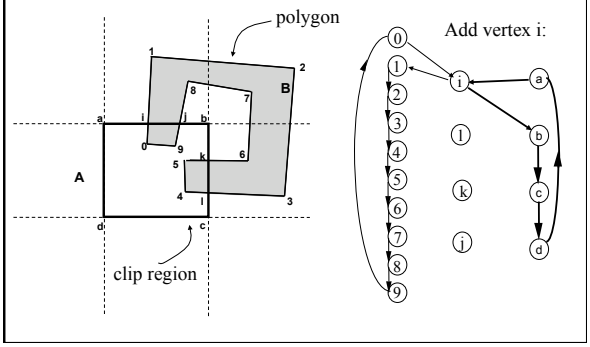
Weiler-Atherton Algorithm

- When we have non-convex polygons then the algorithm above might produce polygons with coincident edges
- This is sometimes OK for rendering but is not for other applications (e.g. shadows)
- The Weiler-Atherton algorithm produces separate polygons for each visible fragment

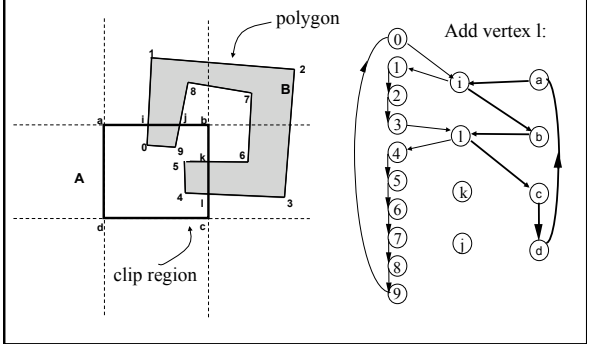
Weiler-Atherton Algorithm



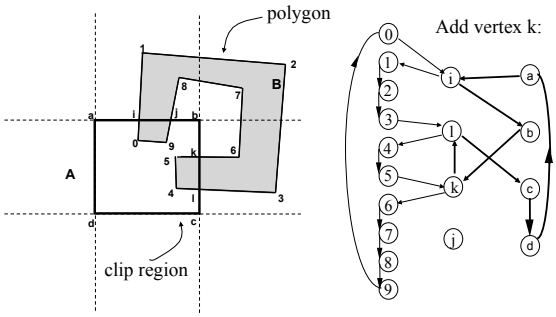
Find the intersection vertices and connect them in the two lists



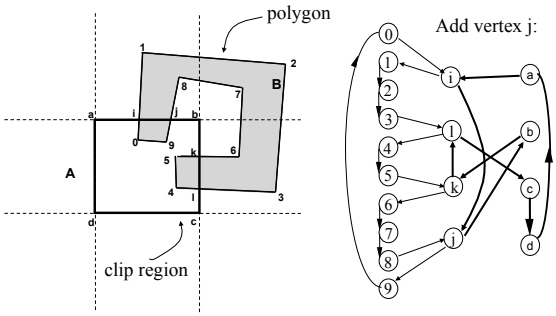
Find the intersection vertices and connect them in the two lists



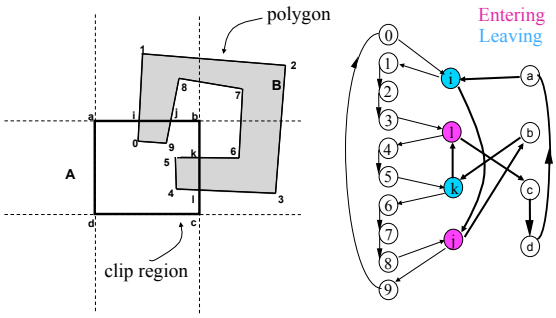
Find the intersection vertices and connect them in the two lists



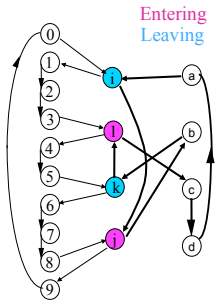
Completed Loop



Classify each intersection vertex as Entering or Leaving

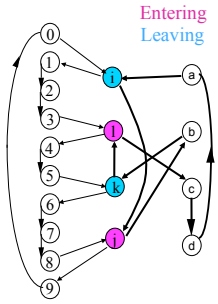


Capture clipped polygons



- Start at an entering vertex
- If you encounter a leaving vertex swap to right hand (clip polygon) loop
- If you encounter an entering vertex swap to left hand (polygon) loop
- A loop is finished when you arrive back at start
- Repeat whilst there are entering vertices

Capture clipped polygons



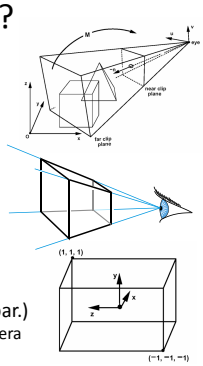
- Loop 1:
  - L, 4, 5, K
- Loop 2:
  - J, 9, 0, I

Clipping Polygons in 3D

- The Sutherland-Hodgman can easily be extended to 3D
  - the clipping boundaries are 6 planes instead of 4 lines
  - intersection calculation is done by comparing an edge to a plane instead of edge to edge

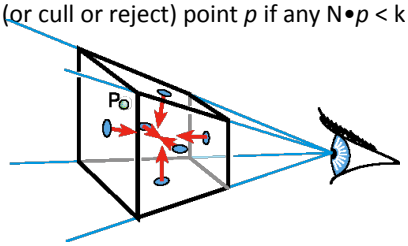
When to clip?

- Before perspective transform in 3D space
  - Use the equation of 6 planes
  - Natural, not too degenerate
- In homogeneous coordinates after perspective transform (Clip space)
  - **Before** perspective divide (4D space, weird *w* values)
  - Canonical,independent of camera
  - The simplest to implement in fact
- In the transformed 3D screen space after perspective division (canonical par.)
  - Problem: objects in the plane of the camera



Clipping with respect to View Frustum

- Test against each of the 6 planes
  - Normals oriented towards the interior
- Clip (or cull or reject) point *p* if any N•*p* < k



Clipping in Homogeneous Coord.

- The Sutherland-Hodgman can also be used for clipping in **4D before dividing** the points by the *w* coordinate
- This can have the advantage that is even more general, it even allows for the front clip plane to be behind the COP

Clipping in Canonical Parallel Space

- The view volume is defined by: $-1 \le x \le 1$, $-1 \le y \le 1$, $0 \le z \le 1$
- Testing for the 4 cases is fast, for example for the x = 1 (right) clip plane:
  - x0 ≤ 1 and x1 ≤ 1 entirely inside
  - x0 ≤ 1 and x1 > 1 leaving
  - x0 > 1 and x1 ≤ 1 entering
  - x0 > 1 and x1 > 1 entirely outside

In Reality

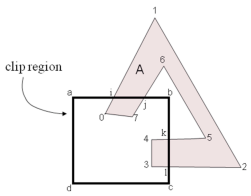
- GPUs used to implement 4D clipping
- Nowadays they seem to do 2D homogenous clipping…

Clipping Recap

- Sutherland-Hodgman is simple to describe but fails in certain cases
- Weiler-Atherton clipping is more robust but more complex
- Both extend to 3D but we need to consider projection and end up clipping in 4D

Exercise

- The following shape (A) is to be clipped against a rectangular clipping area. Apply the Weiler-Atherton clipping method and determine the clipped shape.



- The vertices of the clipping area are labelled a, b, c, d. The vertices of the shape are numbered from 0 to 7. The intersections of the shape's edges and the clipping rectangle are labelled i, j, k, l.