The graphics pipeline is as follows:

1. Build scene
2. Projection transform
3. Clip
4. Perspective division
5. Scan convert

The remaining pieces are:

- User's description of scene
- Perspective division
- Scan convert

Primitives:
- Vertices
- Line segments
- Polygons

When & how to clip:
- Pre-projection, mid-projection, post-projection
- 3d or 4d

Orthographic projection step simply converts to canonical world:

$$M_{w_2} = M_{w_1}^{-1} M_{w_0}$$

Orthographic: convert world:

Pre-projection clipping = post-projection clipping
3d clipping

successively clip against each bounding plane of the view volume

eliminate outside bounding plane

bounding planes of canonical world

bounding plane description

1. point on plane
2. inward-pointing normal

bounding planes of canonical world

exercise: compute p & n for remaining planes

3d clipping

- vertex clipping
- line segment clipping
- polygon clipping

vertex clipping

\( p \) is in with respect to the clipping plane iff 
\[ n \cdot v \geq 0 \] where
- \( n \) is the inward facing normal
- \( v \) is the vector from \( q \) to \( p \)
3d clipping

- vertex clipping
- line segment clipping
- polygon clipping

use test for vertex clipping

Classify endpoint $p_0$ & $p_1$

Case: $p_0$ & $p_1$ in _____
Case: $p_0$ & $p_1$ out _____
Case: $p_0$ in & $p_1$ out _____
Case: $p_0$ out & $p_1$ in _____

Classify endpoint $p_0$ & $p_1$

Case $p_0$ & $p_1$ in:
return $(p_0, p_1)$

Case $p_0$ & $p_1$ out:
return null

Case: $p_0$ in & $p_1$ out:
return $(p_0, p')$

Case: $p_0$ out & $p_1$ in:
return $(p', p_1)$

Do you know how to compute $p'$?
What should happen to $w$ component?

Color at $p'$:
we’ll come back to this when we talk about color models
exercise 1

- clip the line with endpoints (-10,-10,-12) and (2,2,3) using the following order of bounding planes:
  - near
  - far
  - left
  - right
  - top
  - bottom
- show the endpoints at the beginning of each step
- how many intersection computations did you do? against which planes?

exercise 2

- clip the line with endpoints (-10,-2,0) and (10,-2,0) using the following order of bounding planes:
  - near
  - far
  - left
  - right
  - top
  - bottom
- show the endpoints at the beginning of each step
- how many intersection computations did you do? against which planes?

out-code optimization

eliminate unnecessary intersection computations

endpoint p has out-code $b_0b_1...b_5$:
- $b_i = 0$ if p is inside plane i
- $b_i = 1$ else

what does it mean if the $i$th bit of both endpoints is 1?

what does it mean if the $i$th bit of both endpoints is 0?
3d clipping

- vertex clipping
- line clipping
- polygon clipping

### Polygon Clipping

1. Classify vertices
2. Compute intersection points of intersecting edges
   - write out new polygon

### Example

If \( v_0 \) is in then write \( v_0 \)

for \( i = 0 \ldots n-1 \)

- case \( v_i \) & \( v_{i+1} \) in: write \( v_{i+1} \)
- case \( v_i \) & \( v_{i+1} \) out: do nothing
- case \( v_i \) in and \( v_{i+1} \) out: write intersection point
- case \( v_i \) out and \( v_{i+1} \) in: write intersection point and \( v_{i+1} \)

indices taken modulo \( n \)
example

\(i = 0\)
- case \(v_i \& v_{i+1}\) in:
  - write \(v_{i+1}\)
- case \(v_i \& v_{i+1}\) out:
  - do nothing
- case \(v_i\) in and \(v_{i+1}\) out:
  - write intersection point and \(v_{i+1}\)

\(i = 1\)
- case \(v_i \& v_{i+1}\) in:
  - write \(v_{i+1}\)
- case \(v_i \& v_{i+1}\) out:
  - do nothing
- case \(v_i\) in and \(v_{i+1}\) out:
  - write intersection point and \(v_{i+1}\)

\(i = 2\)
- case \(v_i \& v_{i+1}\) in:
  - write \(v_{i+1}\)
- case \(v_i \& v_{i+1}\) out:
  - do nothing
- case \(v_i\) in and \(v_{i+1}\) out:
  - write intersection point and \(v_{i+1}\)

\(i = 3\)
- case \(v_i \& v_{i+1}\) in:
  - write \(v_{i+1}\)
- case \(v_i \& v_{i+1}\) out:
  - do nothing
- case \(v_i\) in and \(v_{i+1}\) out:
  - write intersection point and \(v_{i+1}\)

output: \(u, v_3, w\)

exercise 3
- clip the triangle with vertices \((-10,2,4),\) \((5,2,-4),\) and \((-10,-2,0)\) using the following order of bounding planes
  - near (team 1)
  - far (team 2)
  - left (team 3)
  - right (team 1)
  - top (team 2)
  - bottom (team 3)
- each team should write the vertices after each of its steps on the board
when & how to clip

- pre-projection, mid-projection, post-projection
- 3d or 4d

let's do orthographic first
now for perspective

hyperbolic interpolation

similar to linear interpolation ...

we need to wave our hands a little

graphics pipeline