Who's who

- Application program
- Graphics pipeline
- End user

Who's who for today

- User
- Graphics pipeline

"Vertex based" primitives

- Points
- Lines
- Triangles
- Convex polygons

Graphics pipeline

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

Vertex Transformations
graphics pipeline overview

• simplified pipeline
• general pipeline

projection

standard orthographic view volume

simplified graphics pipeline
graphics pipeline: clip

eliminate "outside" primitive

viewport transformation

standardized world coordinates

image coordinates

scan conversion

simplified graphics pipeline

user's description of scene in standardized, homogenous world coordinates

viewport transformation

scan conversion

3d clipping overview

successively clip primitive against each bounding plane of the view volume

bounding planes of canonical view volume

eliminate outside bounding plane
clipping algorithm

given a clipping plane and a graphics primitive
return "in-side primitive"

clipping algorithms

- vertex
- line segment
- polygon

primitives to clip

- vertex
- line segment
- polygon

vertices to clip

bounding planes of canonical view volume

convenient description of bounding plane

1. point on plane q
2. inward-pointing normal n

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
clipping algorithm

• vertex
• line segment
• polygon

line segment clipping

use test for vertex clipping

1. Classify endpoints \( p_0 \) & \( p_1 \) as in or out

line segment clipping

IN

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

\( \bullet p_0 \)
\( \bullet p_1 \)

IN

Case \( p_0 \) & \( p_1 \) out:
return null

\( p_0 \)
\( \bullet p_1 \)

color at \( p' \):
we'll come back
to this when we
talk about color
models

do you know how to compute \( p' \)?
what should happen
to \( w \) component?

line segment clipping

p_1
\( \bullet p_0 \), \( p' \)

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

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

- clip the line with endpoints (-10,-10,-11) and (2,2,1) 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

out-code optimization

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

out-code optimization

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

out-code optimization

what does it mean if the $i^{th}$ bit of both endpoints is 0?

clipping algorithm

- vertex clipping
- line clipping
- polygon clipping
polygon clipping

1. classify vertices

2. compute intersection points of intersecting edges

&

write out new polygon

• 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

if \( v_0 \) is in then write \( v_0 \)
for \( i = 0 \ldots 3 \)
  - 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
  - case \( v_i \) out and \( v_{i+1} \) in: write intersection point and \( v_{i+1} \)
example

\begin{itemize}
\item \textbf{i=1}
  \begin{itemize}
  \item case $v_i$ \& $v_{i+1}$ in:
    \begin{itemize}
    \item write $v_{i+1}$
    \end{itemize}
  \item case $v_i$ \& $v_{i+1}$ out:
    \begin{itemize}
    \item do nothing
    \end{itemize}
  \item case $v_i$ in and $v_{i+1}$ out:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \item case $v_i$ out and $v_{i+1}$ in:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \end{itemize}
\end{itemize}

\begin{itemize}
\item \textbf{i=2}
  \begin{itemize}
  \item case $v_i$ \& $v_{i+1}$ in:
    \begin{itemize}
    \item write $v_{i+1}$
    \end{itemize}
  \item case $v_i$ \& $v_{i+1}$ out:
    \begin{itemize}
    \item do nothing
    \end{itemize}
  \item case $v_i$ in and $v_{i+1}$ out:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \item case $v_i$ out and $v_{i+1}$ in:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \end{itemize}
\end{itemize}

\begin{itemize}
\item \textbf{i=3}
  \begin{itemize}
  \item case $v_i$ \& $v_{i+1}$ in:
    \begin{itemize}
    \item write $v_{i+1}$
    \end{itemize}
  \item case $v_i$ \& $v_{i+1}$ out:
    \begin{itemize}
    \item do nothing
    \end{itemize}
  \item case $v_i$ in and $v_{i+1}$ out:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \item case $v_i$ out and $v_{i+1}$ in:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \end{itemize}
\end{itemize}

\begin{itemize}
\item output: $u$, $v_3$
\end{itemize}

\begin{itemize}
\item \textbf{i=2}
  \begin{itemize}
  \item case $v_i$ \& $v_{i+1}$ in:
    \begin{itemize}
    \item write $v_{i+1}$
    \end{itemize}
  \item case $v_i$ \& $v_{i+1}$ out:
    \begin{itemize}
    \item do nothing
    \end{itemize}
  \item case $v_i$ in and $v_{i+1}$ out:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \item case $v_i$ out and $v_{i+1}$ in:
    \begin{itemize}
    \item write intersection point
    \end{itemize}
  \end{itemize}
\end{itemize}

\begin{itemize}
\item output: $u$, $v_3$, $w$
\end{itemize}

exercise

\begin{itemize}
\item clip the triangle with vertices (-10,0,0), (0,0,0), and (0,10,0) using the following order of bounding planes
  \begin{itemize}
  \item near (team 1)
  \item far (team 2)
  \item left (team 3)
  \item top (team 1)
  \item bottom (team 2)
  \item right (team 3)
  \end{itemize}
\item each team should write the vertices after each of its steps on the board
\end{itemize}