Visibility

CS 355: Interactive Graphics and Image Processing

Two Parts of 3D Rendering

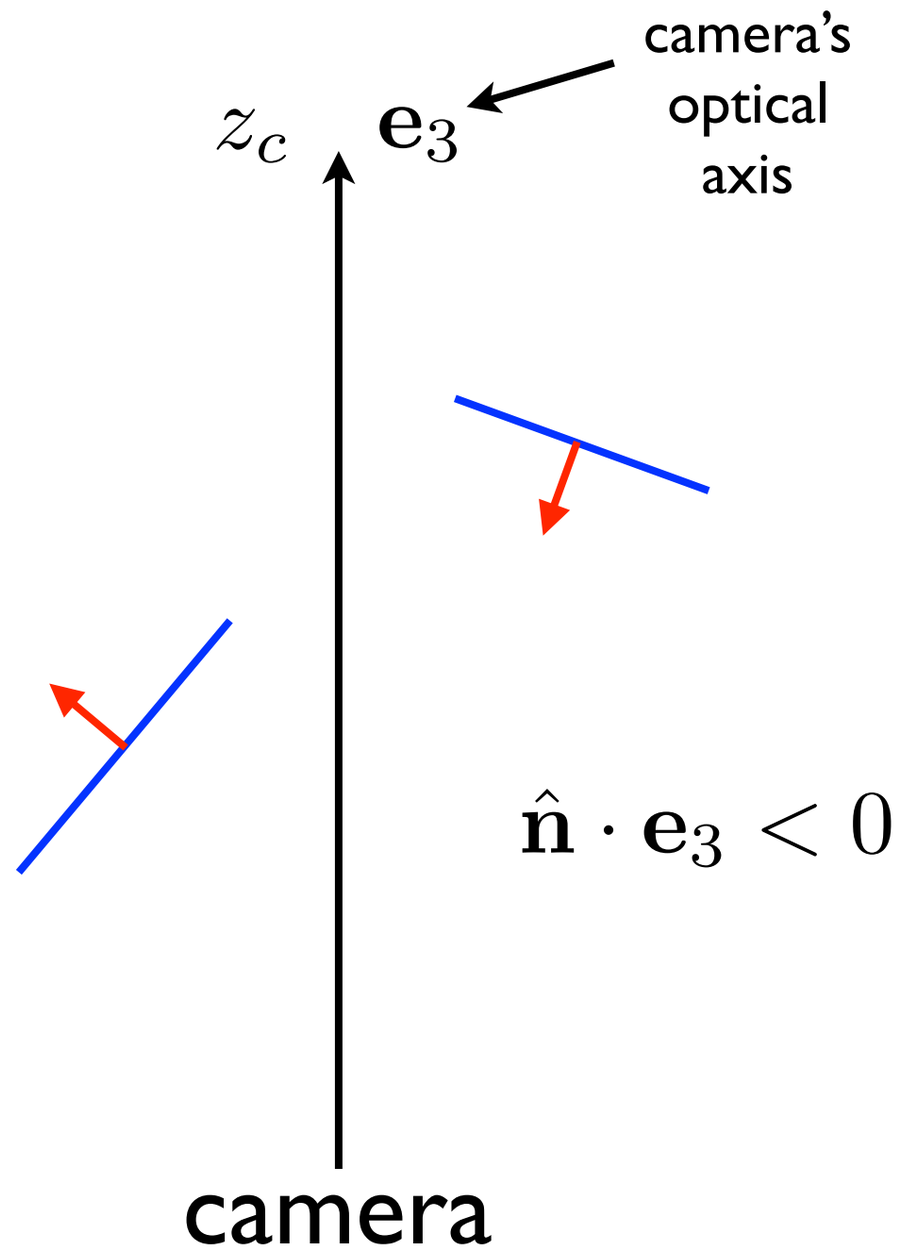
- Two key parts of 3D rendering:
  - What is visible where? (*Visibility*)
  - What light is coming from there? (*Lighting*)

Visibility

- Parts of visibility:
  - ✓ Where is everything relative to the camera? (world-to-camera transformation)
  - ✓ What is within the field of view? (clip matrix)
- What is in front of what else?

Back-Face Culling

- Simple idea:
  - Faces that point *towards* camera may be seen
  - Faces that point *away* from camera cannot be seen

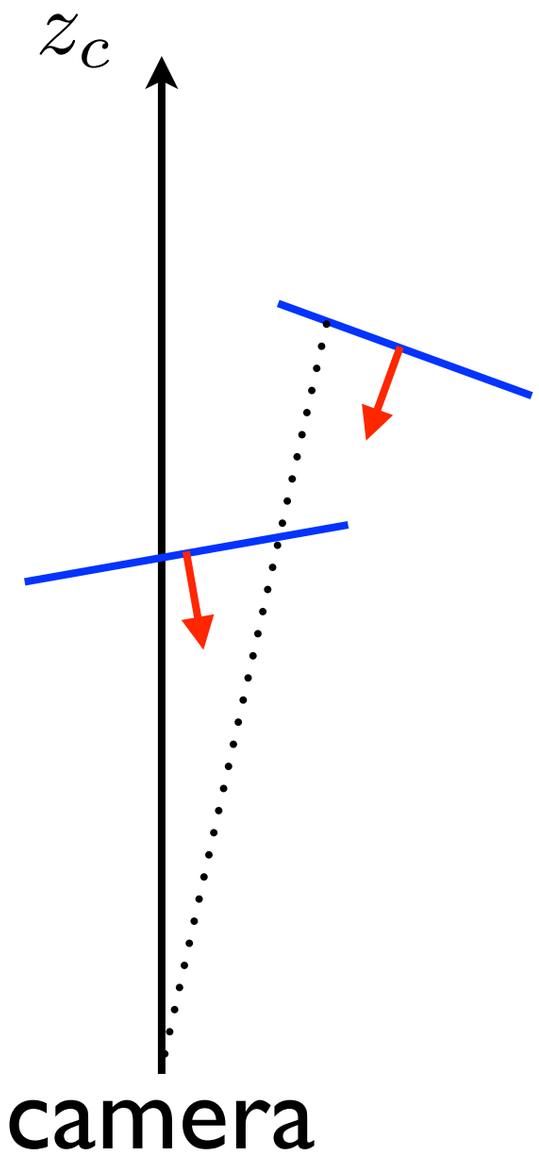


Occlusion Testing

- See what is in front of what else

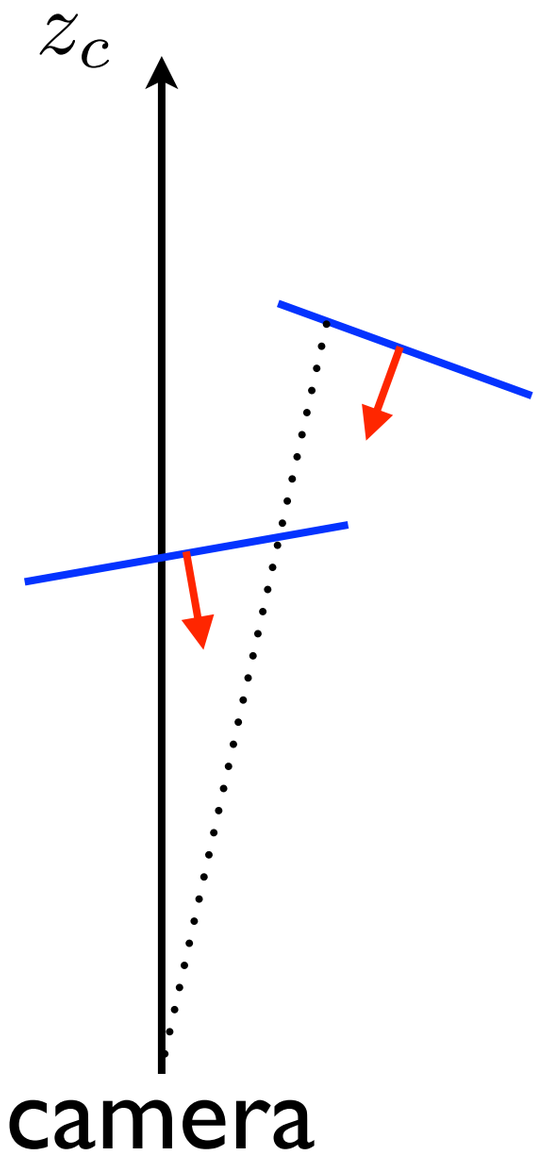
- Test for things that fall on the same camera position

- If opaque, one object *occludes* the other



Occlusion Testing

- Three common ways:
  - Ordered rendering (painter's algorithm)
  - Image space testing (z-buffering)
  - Ray casting



Painter's Algorithm

- Simple idea:
  - Render from back to front (decreasing z)
  - Draw things over top of others on screen
  - Last one drawn wins!
- Big problem:
  - Polygon depth isn't strictly ordered
  - Intersecting
  - Mutually overlapping

This technique isn't used much anymore, but the idea shows up in lots of places

Z-Buffering

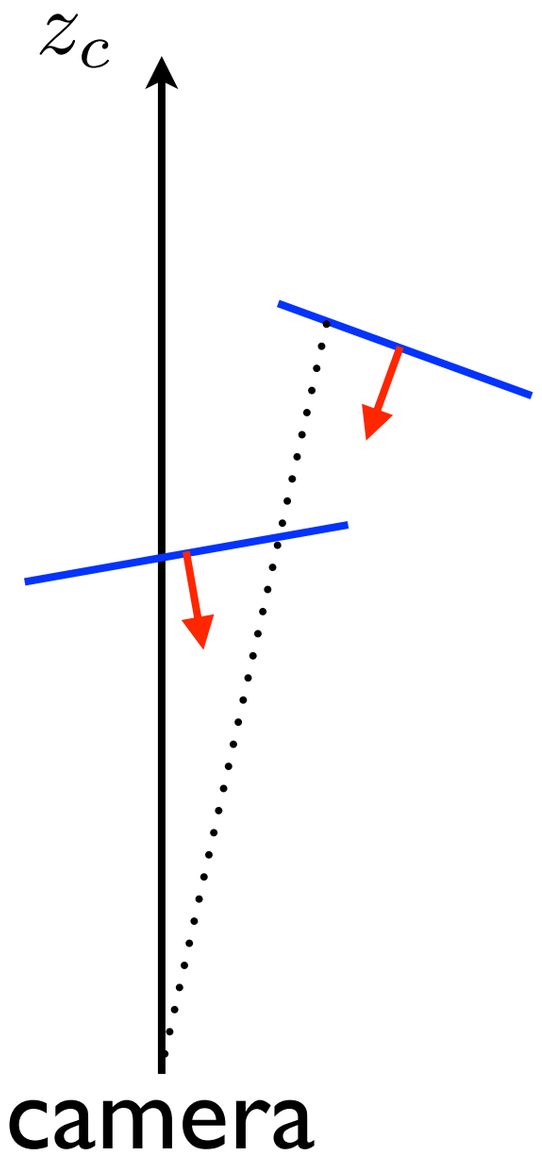
canonical screen coordinates

$$\begin{bmatrix} x/w \\ y/w \\ z/w \\ 1 \end{bmatrix} \sim \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} \text{zoom}_x & 0 & 0 & 0 \\ 0 & \text{zoom}_y & 0 & 0 \\ 0 & 0 & \frac{f+n}{f-n} & 0 \\ 0 & 0 & \frac{-2nf}{f-n} & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

normalized depth

Z-Buffering

- Keep an image buffer that stores the depth of what is currently rendered at each pixel
- Render in any order
- Drawn new stuff
  only if closer

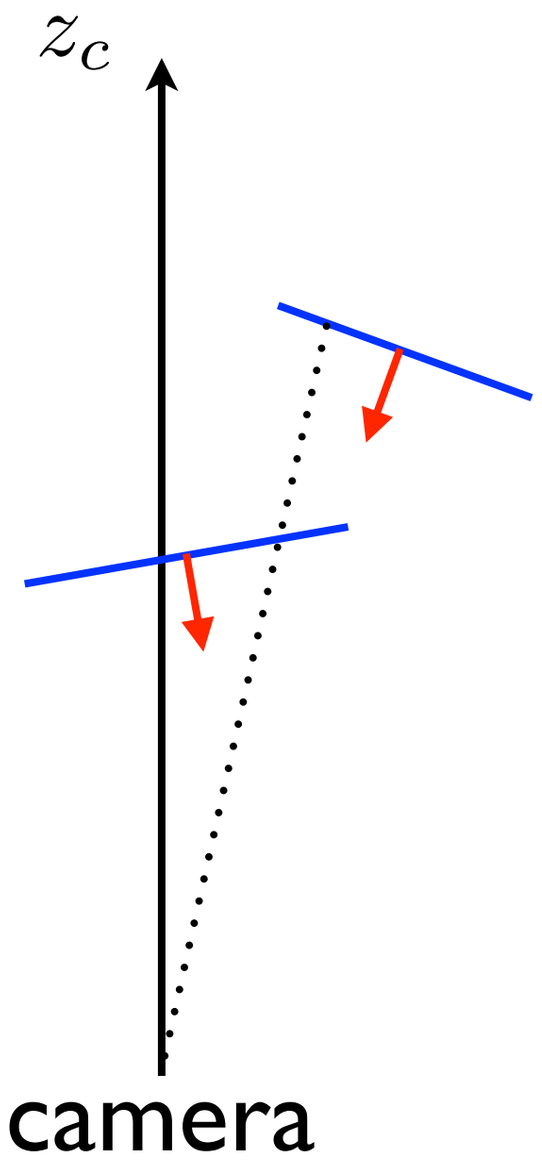


Z-Buffering

- Issue:
  Quantization of the finite-precision z buffer

- Round-off error may be an issue (most use floating point)

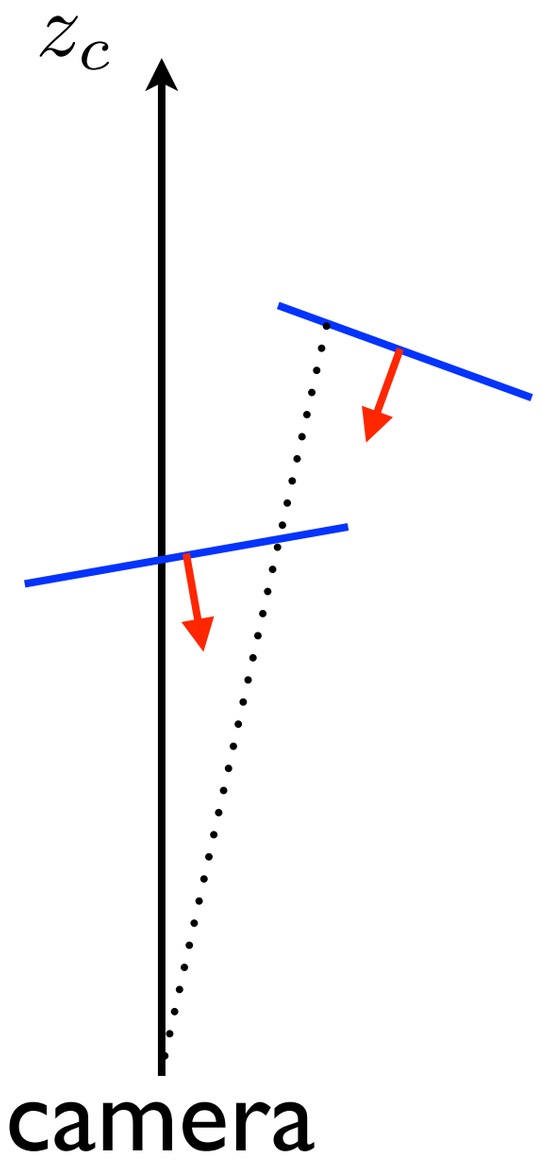
- Nonlinear by depth (coarser farther away)



Z-Buffering

- Issue:
  Discretization of the
  image z buffer

- Hard to do antialiasing
  (partial painting of
  pixels on boundaries)



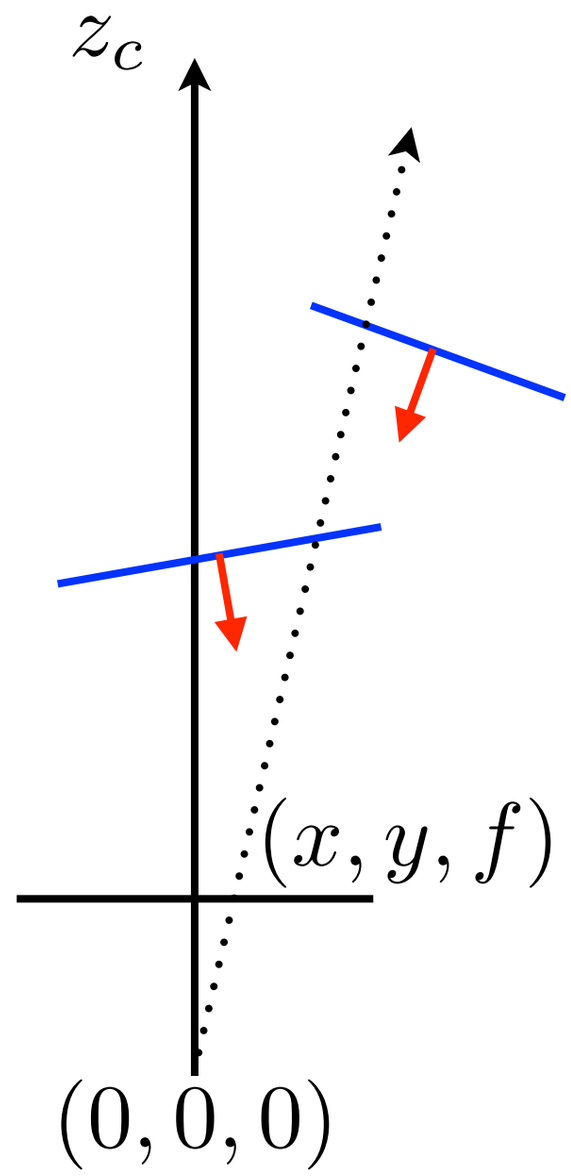
Ray Casting

- Idea: shoot a ray out from the camera's focal point through the pixel location

- What does it hit first?

- Lots of ray-primitive intersection tests

(Arthur Appel, 1968)



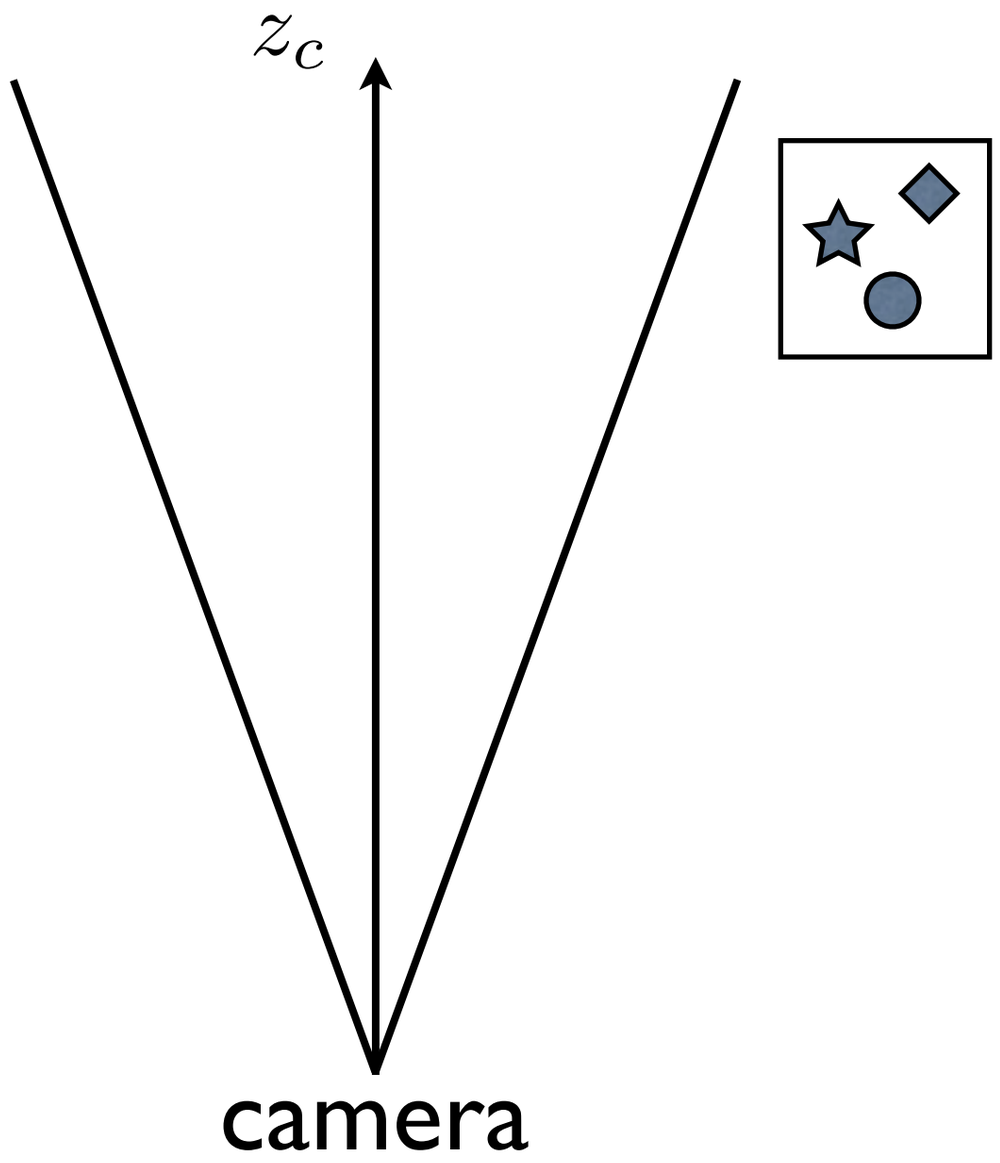
Objects & Bounding Boxes

- Almost all of these ideas can accelerated by grouping primitives and using bounding boxes

- Typically AABBs in original object space

- Transform to a general BB in camera space

- Throw out if all corners are out of view, not visible, etc.



Coming up...

- Lighting