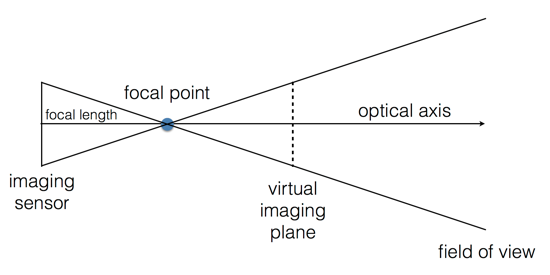
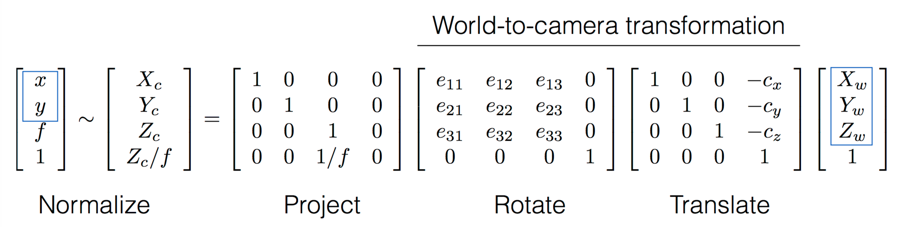
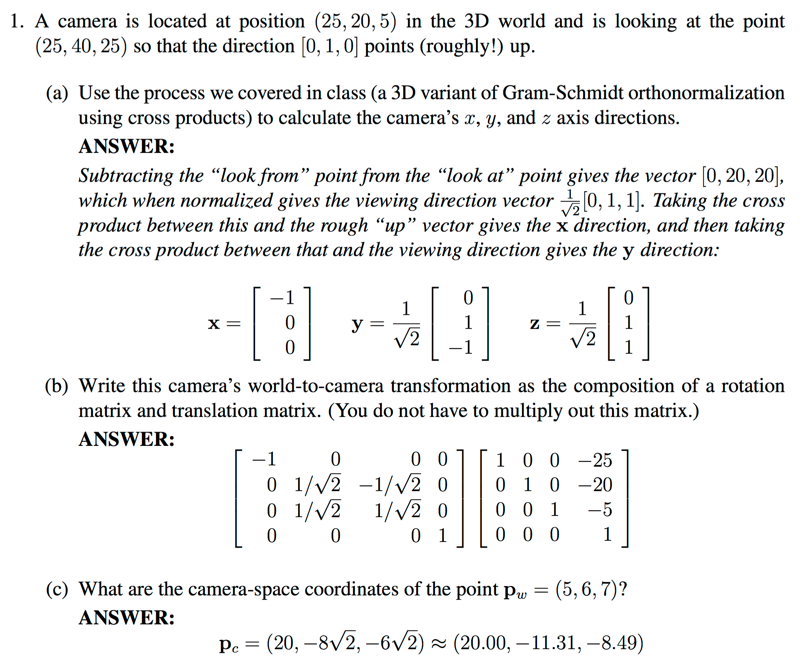
Orthographic Projection: dropping a dimension. 4D to 3D, etc…

Perspective Projection: pinhole camera:

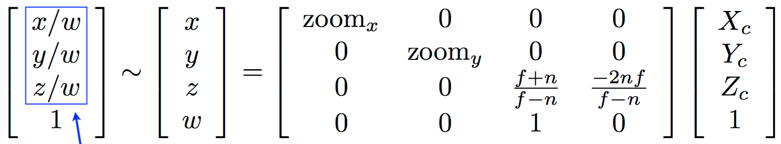
3D Geometry Pipeline:

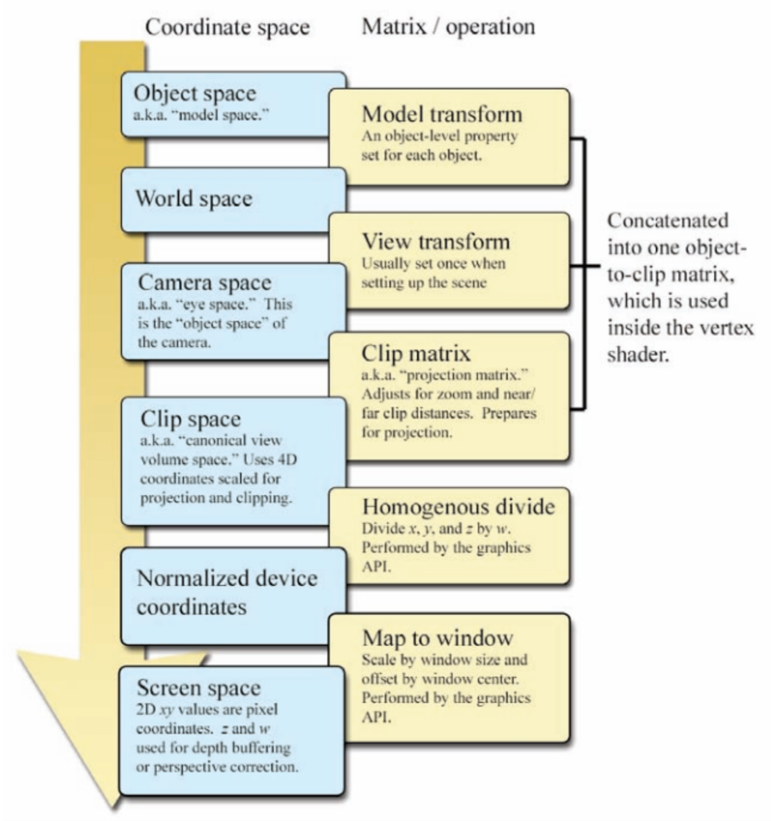


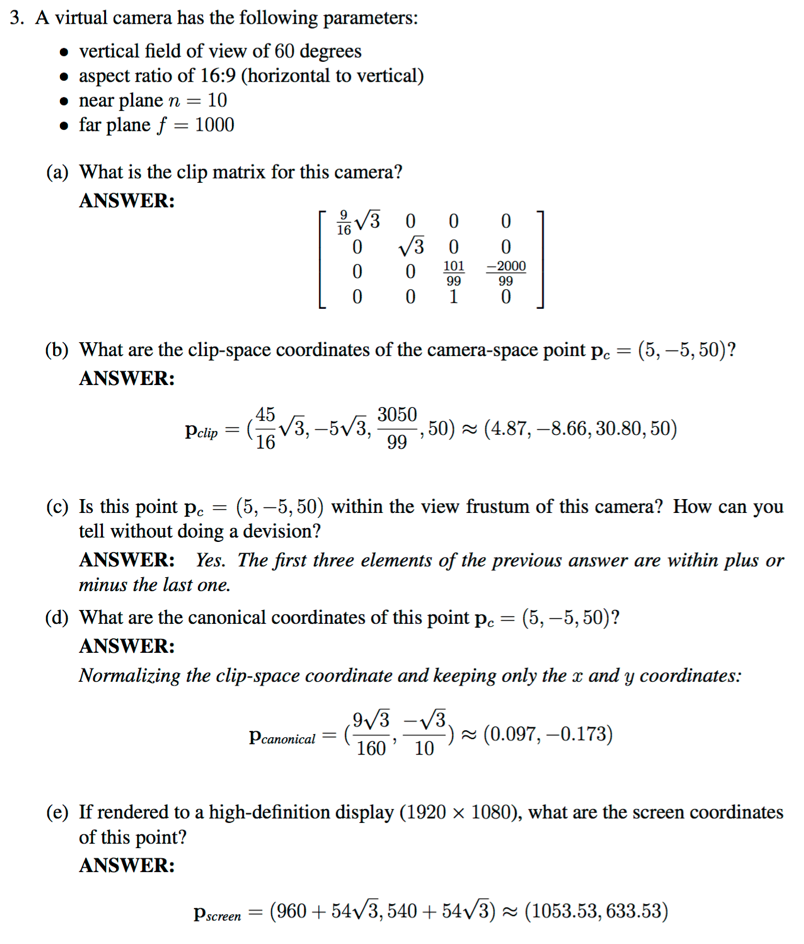
World to Camera: Translate, then Rotate.

Clipping tests: (left x < -w) (right x > w) (bottom y < -w) (top y > w)

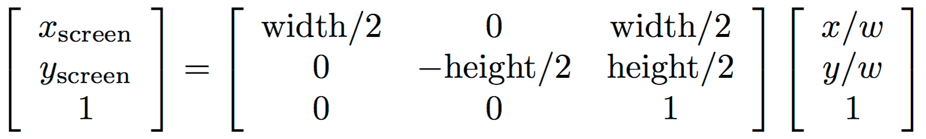
(near z < -w) (far z > w)

Clip Matrix:





To Screen Space:



Z-Buffering: testing for visibility, in front or behind an object.

Keep an image buffer that stores the depth of what is rendered at each pixel.

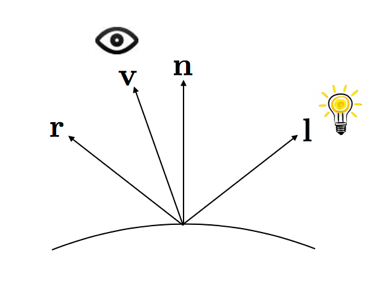
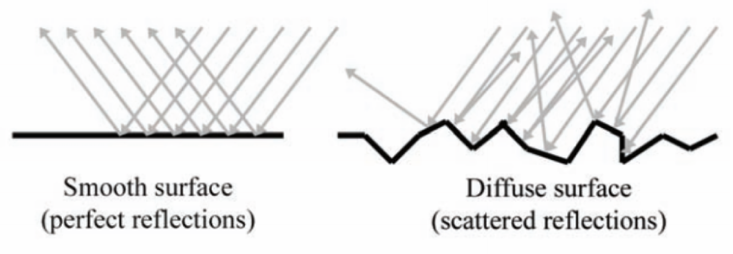
Render in any order. Draw new stuff if closer.

Issue: Round-off error. Nonlinear by depth (coarser farther away).

Issue: Hard to antialias

Ray Casting:

Shoot a ray out from the camera’s focal point, what does it hit first?

Lighting: Direct, Indirect, Ambient

The surface normal

The lighting direction

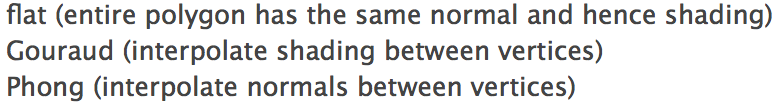
The viewing direction

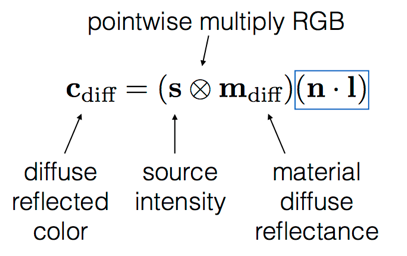
The reflected light direction

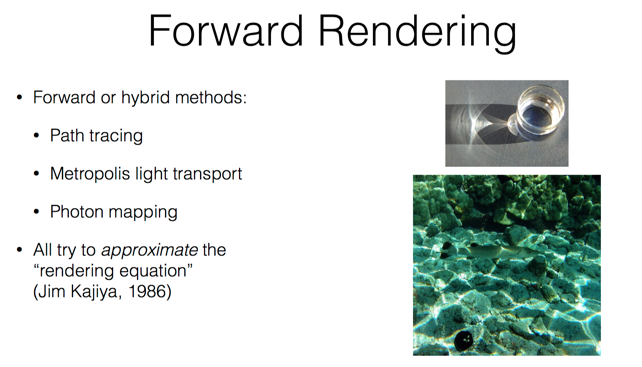
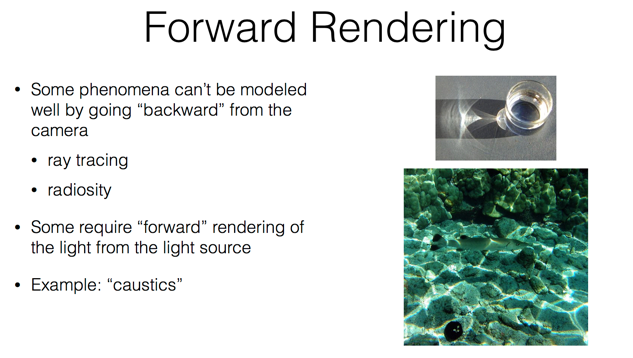
Irradiance: how much light per unit area

Diffuse:

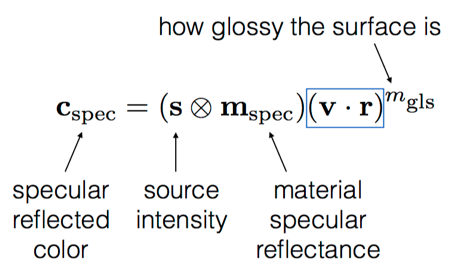
Perfectly diffuse is Lambertian

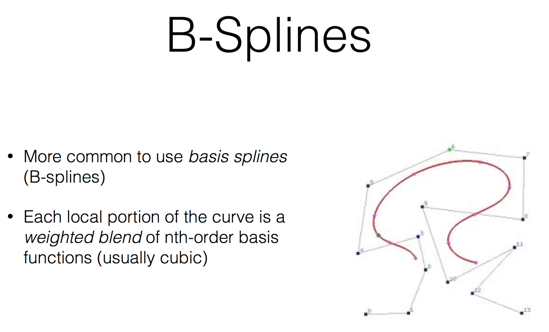
 Lighting direction matters, not viewing direction

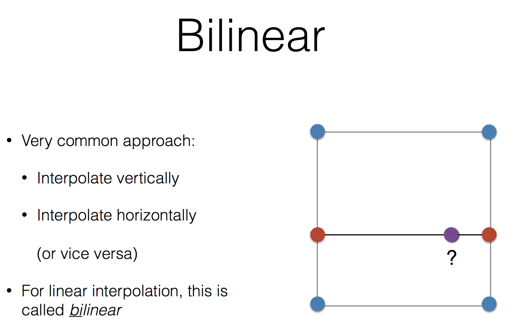




Specular:







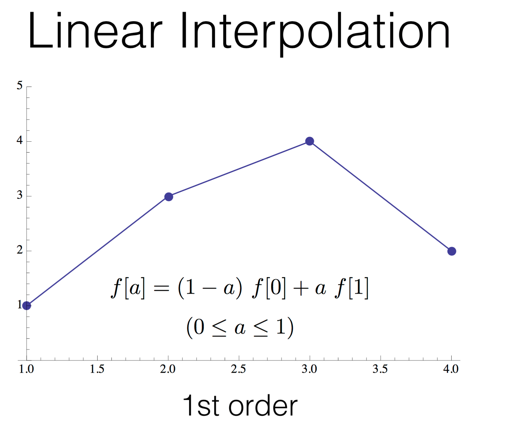
Umbra = area in full shadow

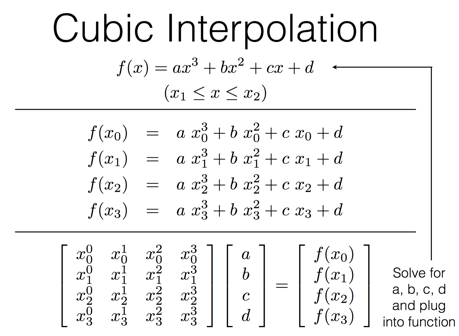
Penumbra = area in partial shadow

Ray Tracing:

Shoot a ray out through the pixel on the imaging plane.

When it hits something, recursively shoot ore rays.

 This adds reflections, and refraction through transparent materials

 Good for specular reflections and refraction

Not so good if the camera is moving a lot I guess?

Radiosity:

Involves diffuse light, so it is independent of the viewpoint

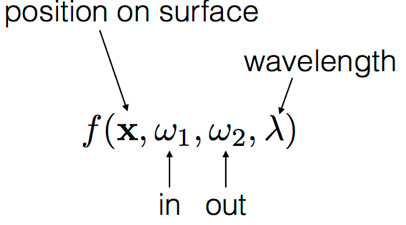
Colored boxes in a room, reflecting their colors.

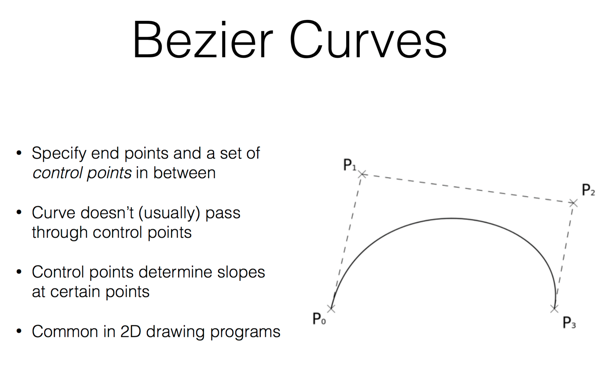
This is great if objects are not moving but camera is

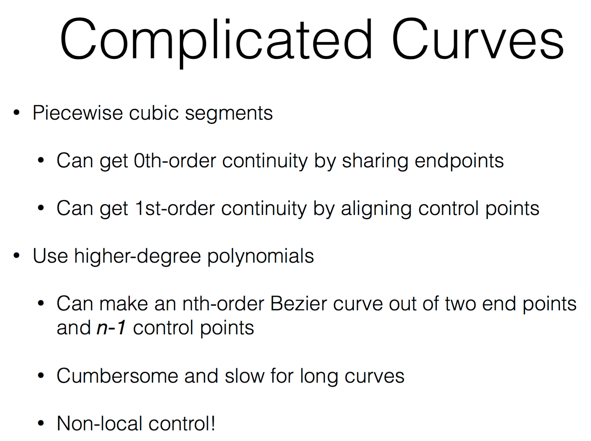
Ray Tracing is good for specular reflections, refraction, etc…

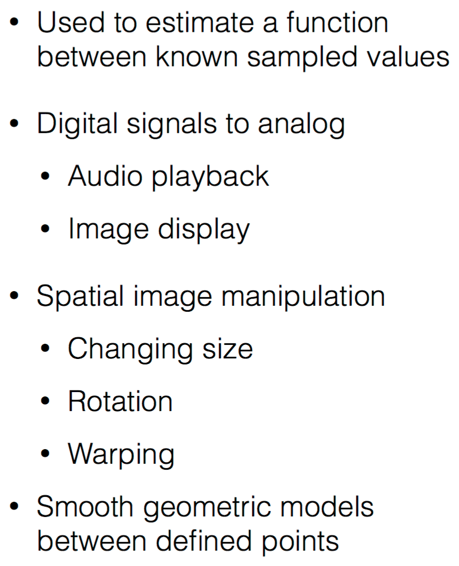
Radiosity is good for diffuse interreflections

You can combine them though

BRDF: Bidirectional Reflectance Distribution Function





Interpolation:

