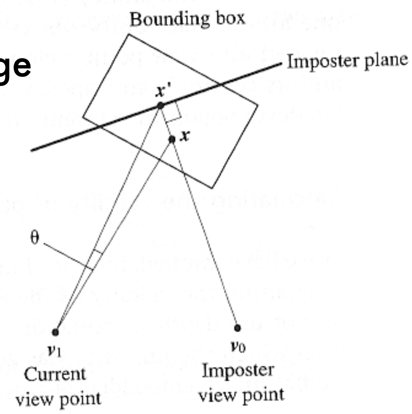
Image-based Rendering

Image-based Rendering (IBR)

- Goal of all IBR techniques
  - Decouple Rendering Time from Scene Complexity
- Represent geometry as imagery
  - Captured from real scenes
    - Enable imagery not possible with local illumination models
  - Pre-computed from geometry
- Use imagery during rendering

Simple 2D IBR: Planar Imposters

- Replace geometry with texture-mapped billboard
  - Rely on frame coherence
- Use until error is too large
  - How to compute error?
- Question
  - Where/what size is plane?



Varying Rendering Resources

- Partition objects based on distance to viewer
  - Render into different layers
  - Composite layers back-to-front
- Render closer objects more often
  - Devote more rendering resources to them!
- Metrics to determine validity of each layer
  - Geometric, photometric, sampling

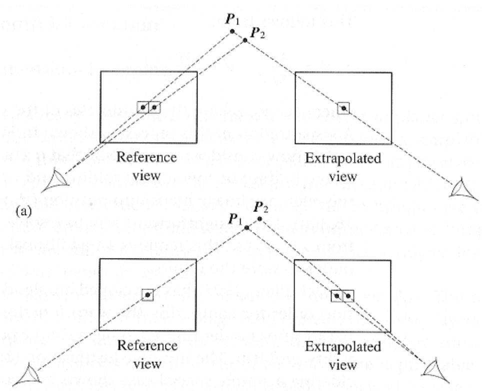
Adding and Using Depth Information

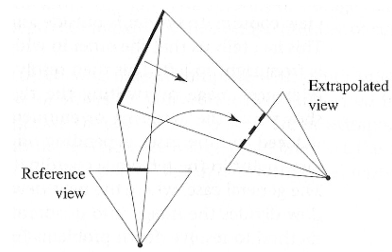
- Three approaches
  - One depth value per imposter (previous technique)
    - Doesn't support motion parallax, planar nature perceptible
  - Save z-value per pixel
    - 3D Image Warping
  - Multiple image and depth values per pixel
    - Layered Depth Images (LDI)

3D Image Warping (Image + Z-buffer)

- Can think of it as sparse data volume
  - Contains visible pixels for a single view
- Can generate a new view
  - Account for 3D structure: parallax, hidden surfaces
- Should be close to old one
  - Necessary data will will not be there if eye moves too much

Problems with 3D Image Warping



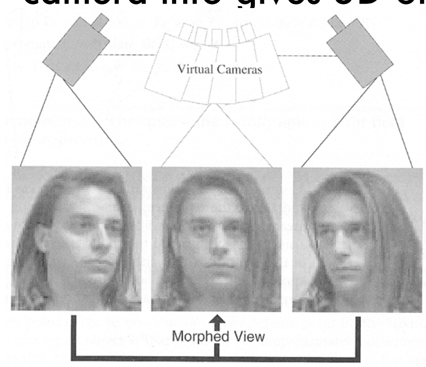


Layered Depth Images

- Improve on 3D Image Warping by adding addition data for each pixel
  - All surfaces intersected by ray through pixel
    - Save surface normals at each pixel
  - Data size linear with average depth complexity
- Can model a scene with a set of LDIs
  - Becomes a 4D dataset, rather than 3D

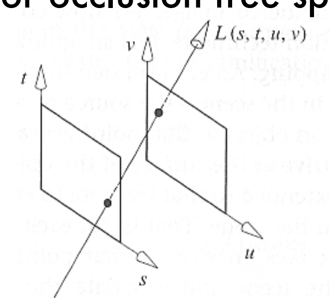
View Interpolation

- Take a collection of depth images
  - Interpolate between any pair of them
- Depth + camera info gives 3D of each pixel



4D Approaches: The Lumigraph or Light Field Rendering

- Store the radiance in all directions from each point in space
  - 5D: $L(x,y,z,\alpha,\beta)$
- Simplify: store only for occlusion free space
  - 4D: $L(s,t,u,v)$



Rendering a Light Field

- Intersect view ray with planes to get (s,t,u,v)
- Set pixel to L(s,t,u,v)