Light Fields

Light Fields

By Levoy and Hanrahan, SIGGRAPH 96

Representation for sampled plenoptic function

- stores data about visible light at various positions and directions

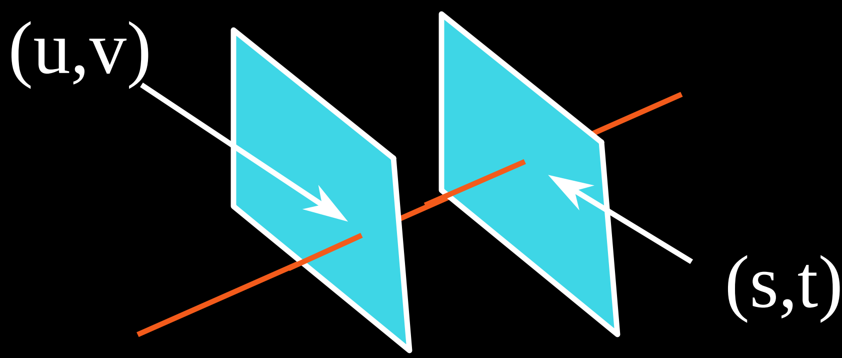
Created from set of images

Resamplings employ data from lots of different images

Light Field Dimensionality

- Position and direction for each sample is a 5D space
- For empty space (no occlusion), space reduced to 4D
  - sample is constant along a line
  - light field defined on 4D space of directed lines

Slab Representation



- **Define two parallel planes**
  - ***uv*-plane and *st*-plane**
- **Light field defined as L(*u*,*v*,*s*,*t*)**
  - **(r,g,b) for each (*u*,*v*,*s*,*t*) tuple**
- **Use multiple slabs to cover larger space**

Sampling

Typically create regular sampling of uv- and st-planes

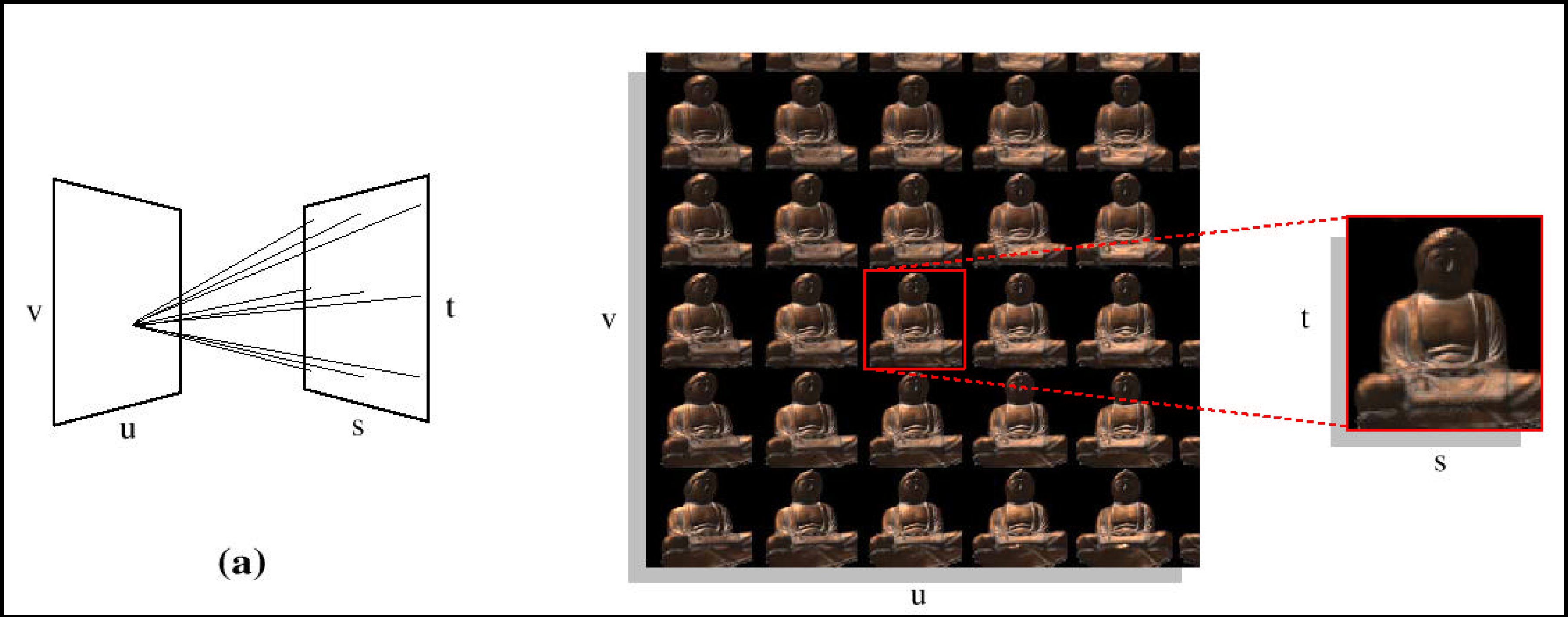
Place eye point at (u,v) on the uv-plane

Generate image with each corresponding to a point on the st-plane

- each pixel for image (u,v) supplies sample (u,v,x,y)
- using skewed perspective matrix, (x,y) = (s,t)

Data looks like 2D array of 2D images

Visualization of Light Field



from Levoy and Hanrahan, "Light Field Rendering," *Proceedings of SIGGRAPH 96*, page 34.

Generating Samples

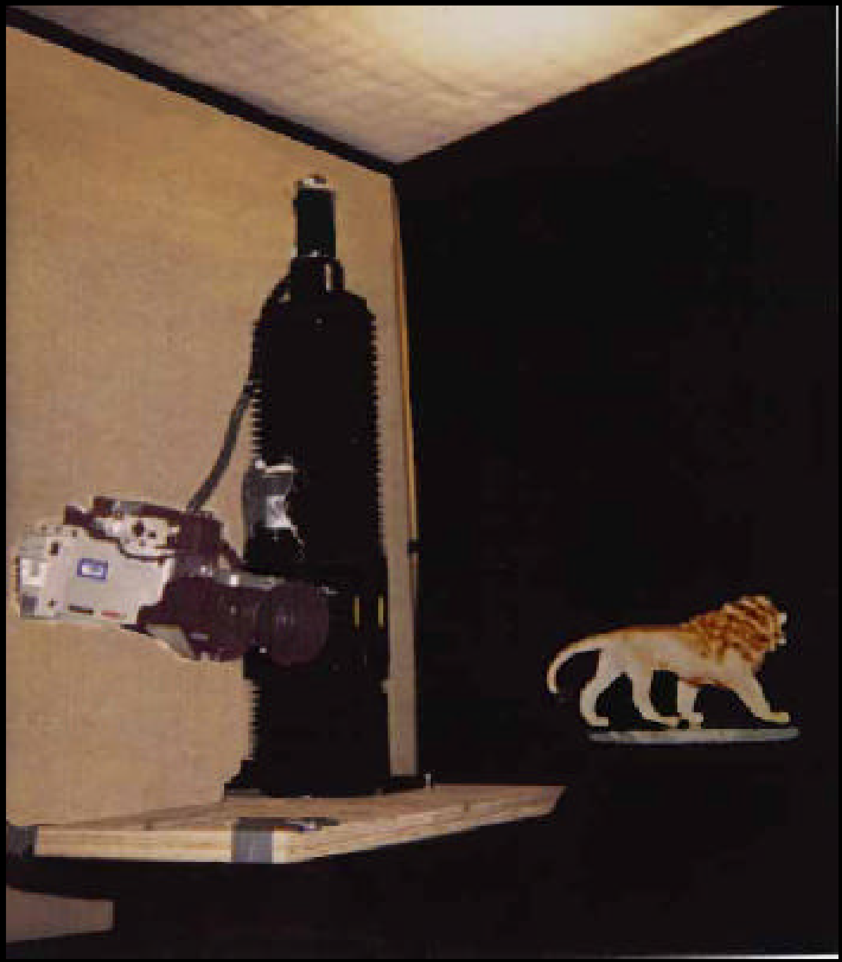
Using rendered images

- Place eye at (u,v)
- Skew projection to cover proper (s,t) range
- Generate image

Using real photographs (looking inward)

- Computer-controlled camera on planar gantry
- Camara tilts to center on object
- (s,t) resampled from (x,y)
- Object platform (and lighting) rotates to capture different slabs

Stanford Light Field Gantry



from Levoy and Hanrahan, “Light Field Rendering,” *Proceedings of SIGGRAPH 96*, page 36.

Resampling

Foreach pixel in the rendered image

- **compute line coordinates (intersections with uv- and st-planes**
- **Apply nearest neighbor, bilinear, or quadralinear sampling to generate value of pixel from nearby lines in light field**

Computing Line Parameters

Possible using ray/plane intersection

Faster using “texture mapping” to take advantage of plane coherence

- Store (u,v) coordinates in texture map
- Render uv-plane as textured rectangle
- Look up (u,v) coordinates for each pixel
- Repeat for (s,t) coordinates

Anti-aliasing

- Pre-filter data to remove aliases
- Integrate over range of eye points to filter (u,v)
- Apply lens aperture to filter (s,t)
- Filter size should be consistent with sample spacing

Compression

- Light fields can be BIG (gigabytes)
- Want to transmit over internet
- Want to fit in memory
- Need random access during reconstruction
- Compression can be slow, decompression must be fast

Two Stage Compression/Decompression

Lossy vector quantization (VQ) compression

- Decompose data into small chunks, described as vector
- Train with data to generate codebook (containing codewords to represent)
- Store index of best codeword for each vector

Lossless entropy coding (using gzip)

Decompression

Decompress entropy coding (gunzip) on loading to memory

- entropy coding doesn't allow random access

Decompress vector quantization (fast lookup) for each line sample on the fly

May compress 24:1 for VQ, 5:1 for gzip, total of 120:1

Live Demo (Stanford implementation)

Videos

- **Levoy and Hanrahan. "Light Field Rendering."** ***Proceedings of SIGGRAPH 96.***
- **Regan et al., "A Real-time, Low Latency Light Field Renderer",** ***Proceedings of SIGGRAPH 99***
- **Wood et al., "Surface Light Fields for 3D Photography",** ***Proceedings of SIGGRAPH 2000***
- **Isaksenm et al., "Dynamically Reparameterized Light Fields",** ***Proceedings of SIGGRAPH 2000***