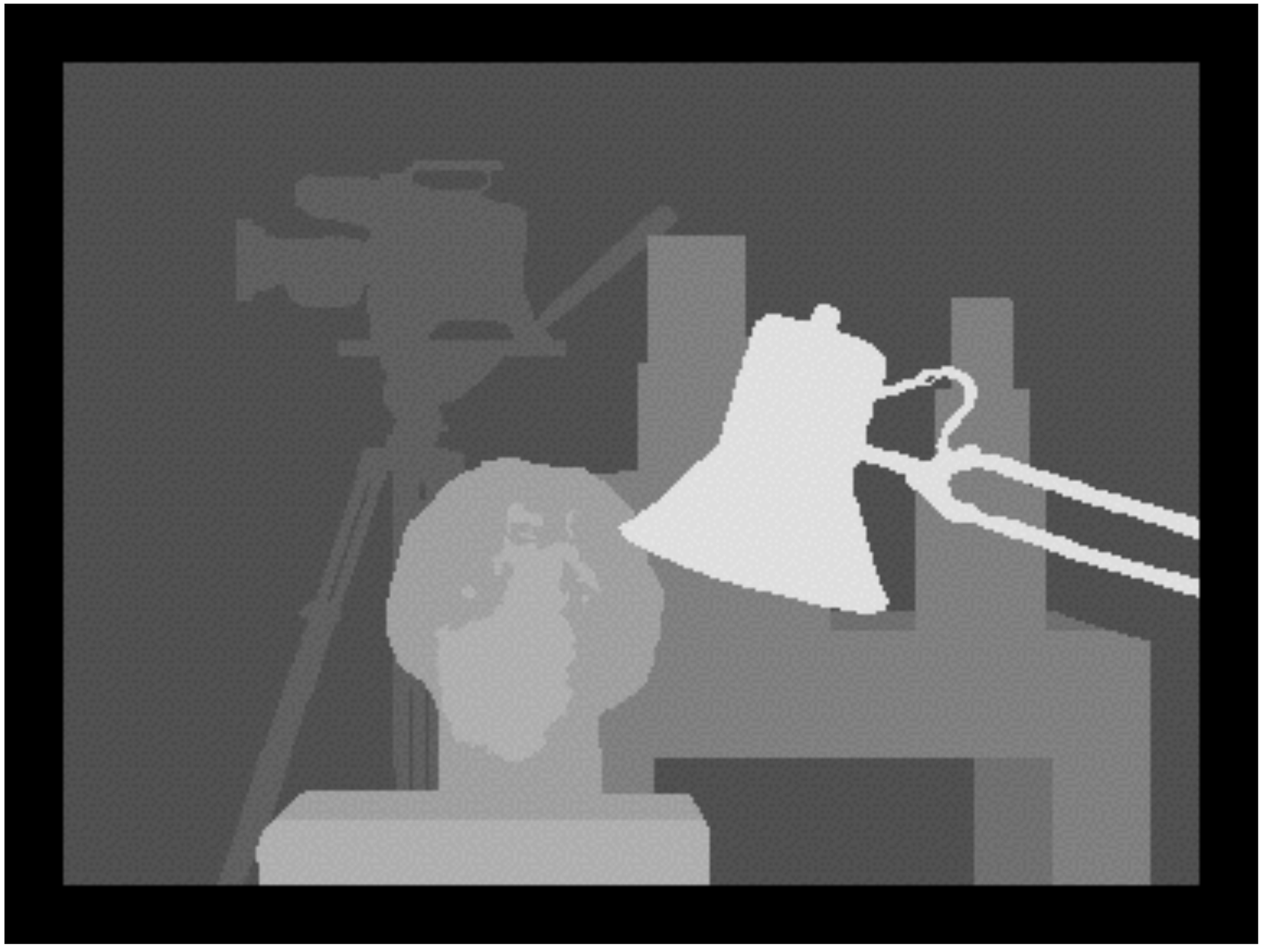
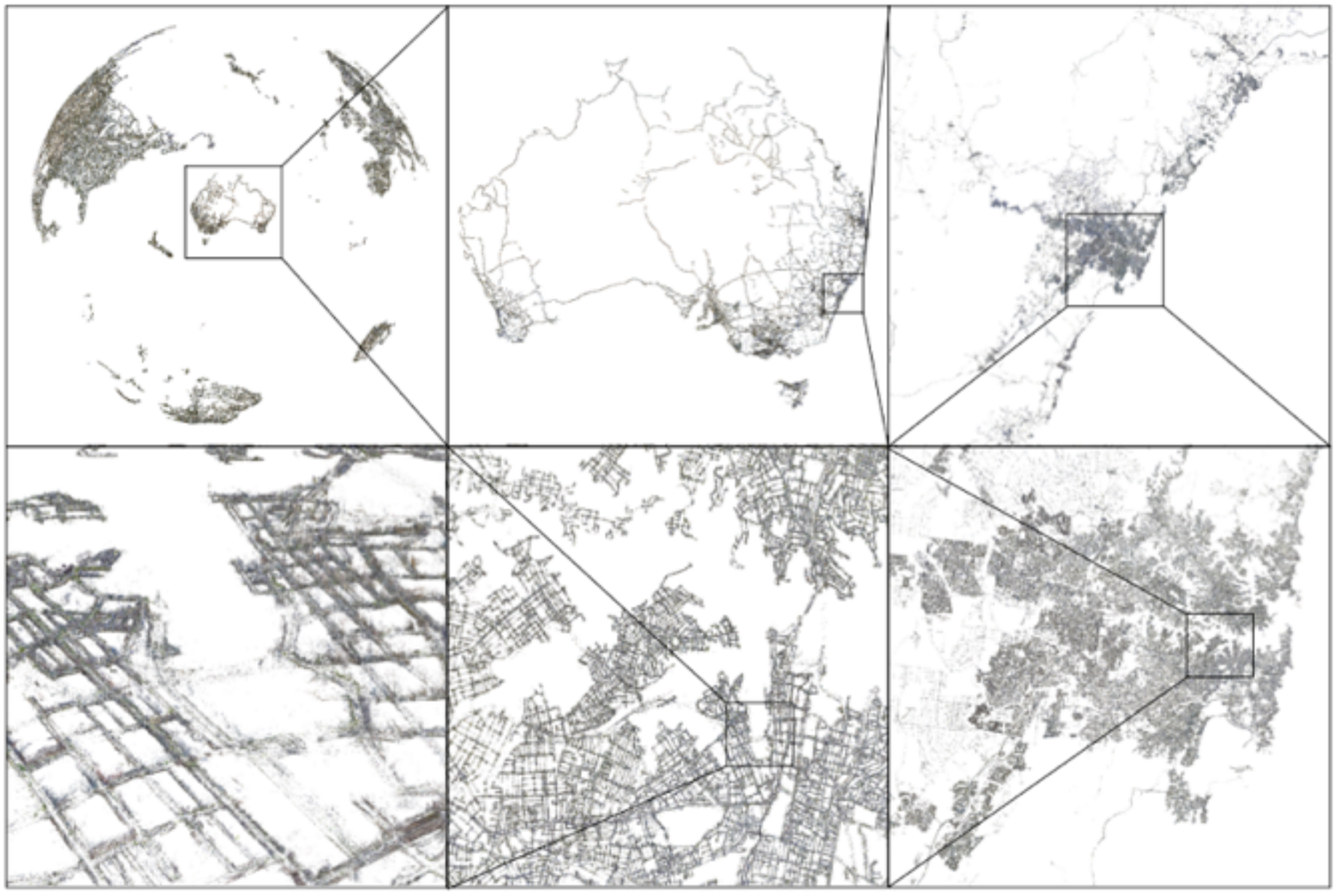
3D Reconstruction with Computer Vision
Meeting 24: Stereo Redux



Slides by Richard Szeliski and others

CS 378 Fall 2014, UT Austin, Bryan Klingner, 18 November

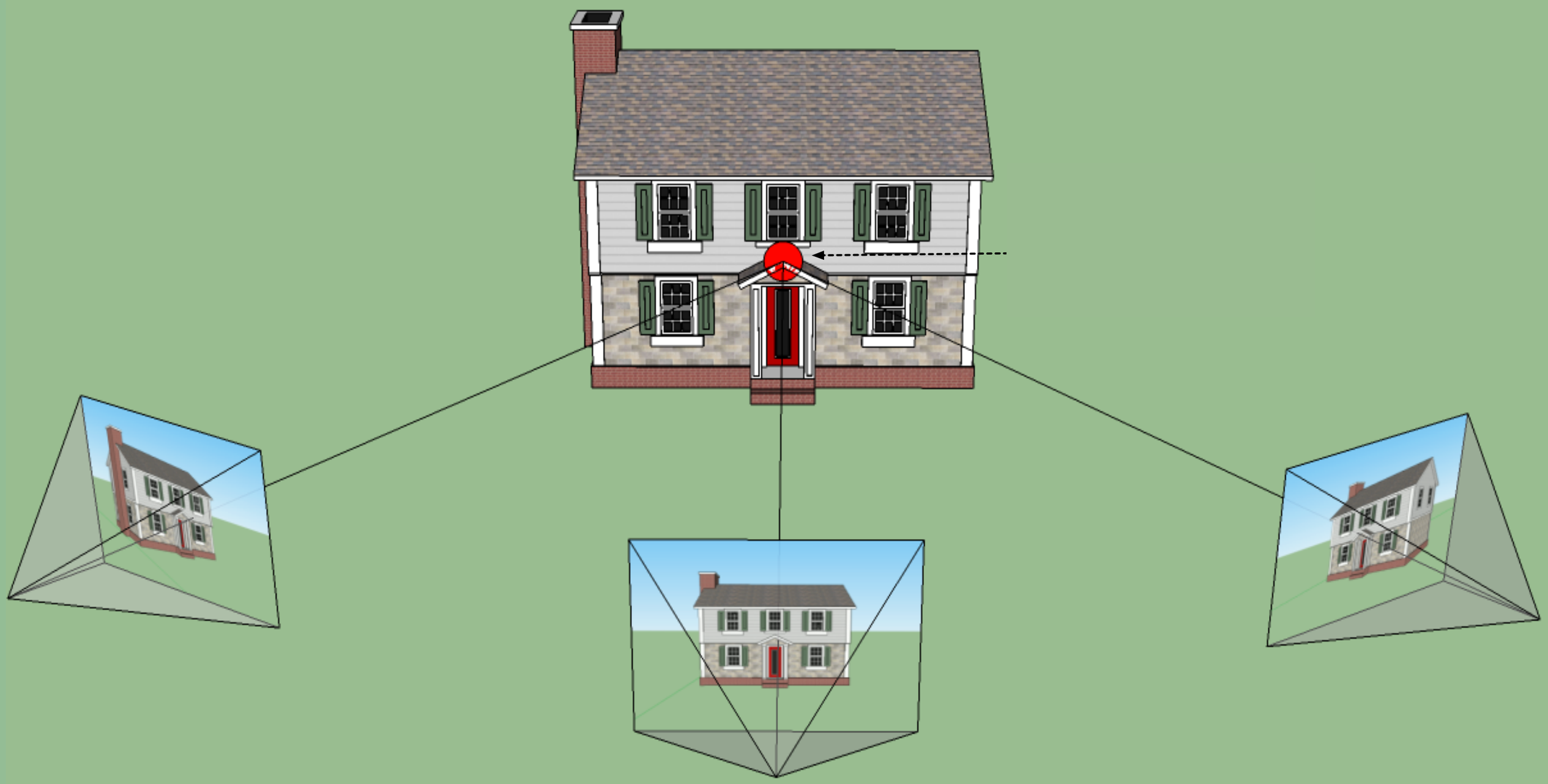
Final Project



- **Progress report 2 Due Thursday, 20 November**
- **Final report Due Tuesday, 2 December**

- **In-class presentations Tuesday, 25 Nov., Tuesday 2 Dec., and Thursday 4 Dec.**

Idea: 3D from multiple images of a scene



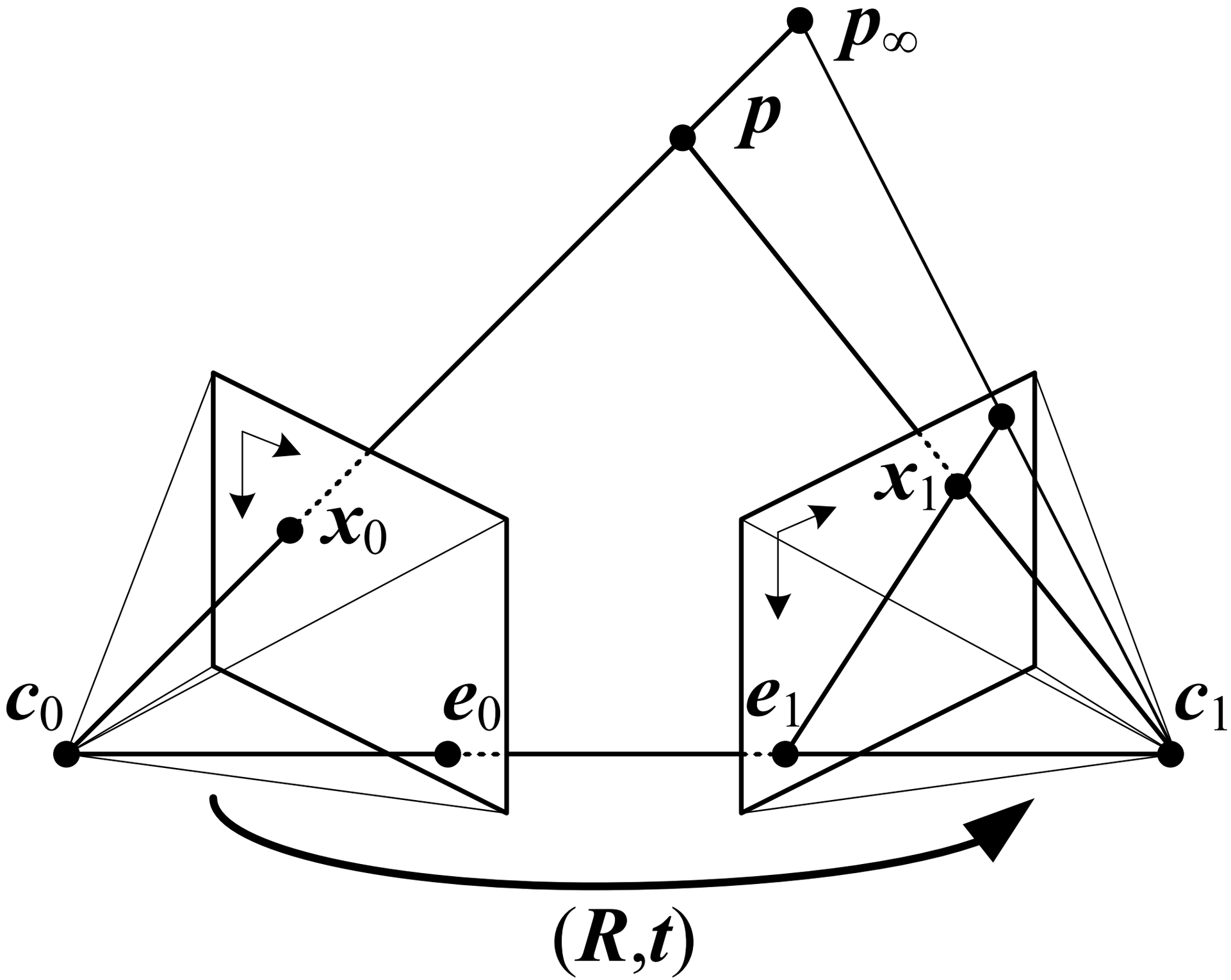
Stereo Matching

Given two or more images of the same scene or object, compute a representation of its shape

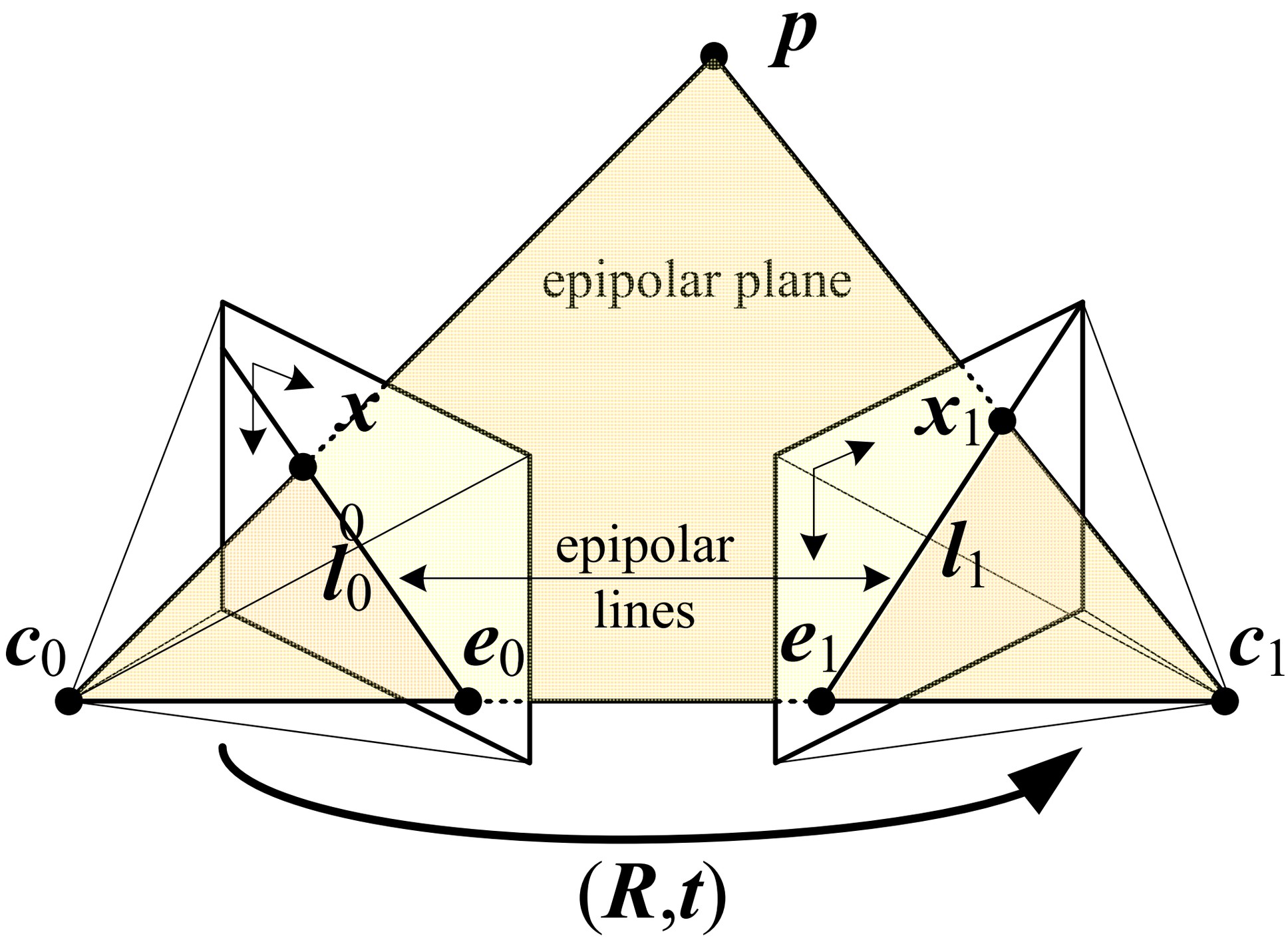
What are some possible representations?

- depth maps
- volumetric models
- 3D surface models
- planar (or offset) layers

Epipolar Geometry



Epipolar Geometry



Stereo: epipolar geometry

for ***two*** images (or images with collinear camera centers), can find epipolar lines

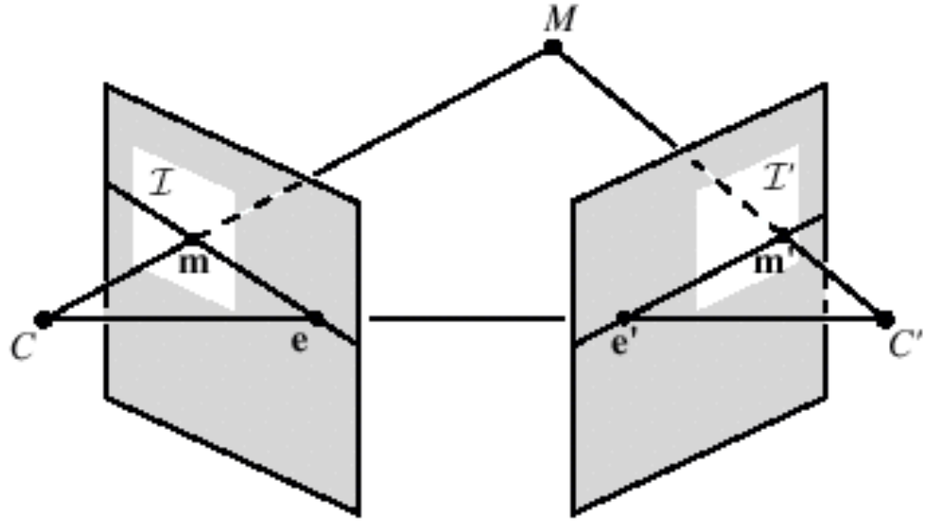
epipolar lines are the projection of the ***pencil*** of planes passing through the centers

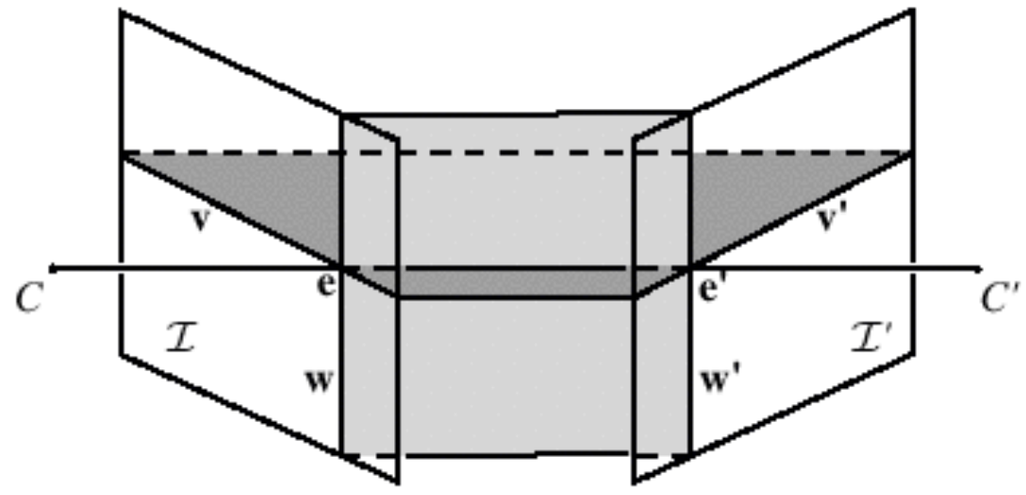
Rectification: warping the input images (perspective transformation) so that epipolar lines are horizontal

Rectification

Project each image onto same plane, which is parallel to the epipole

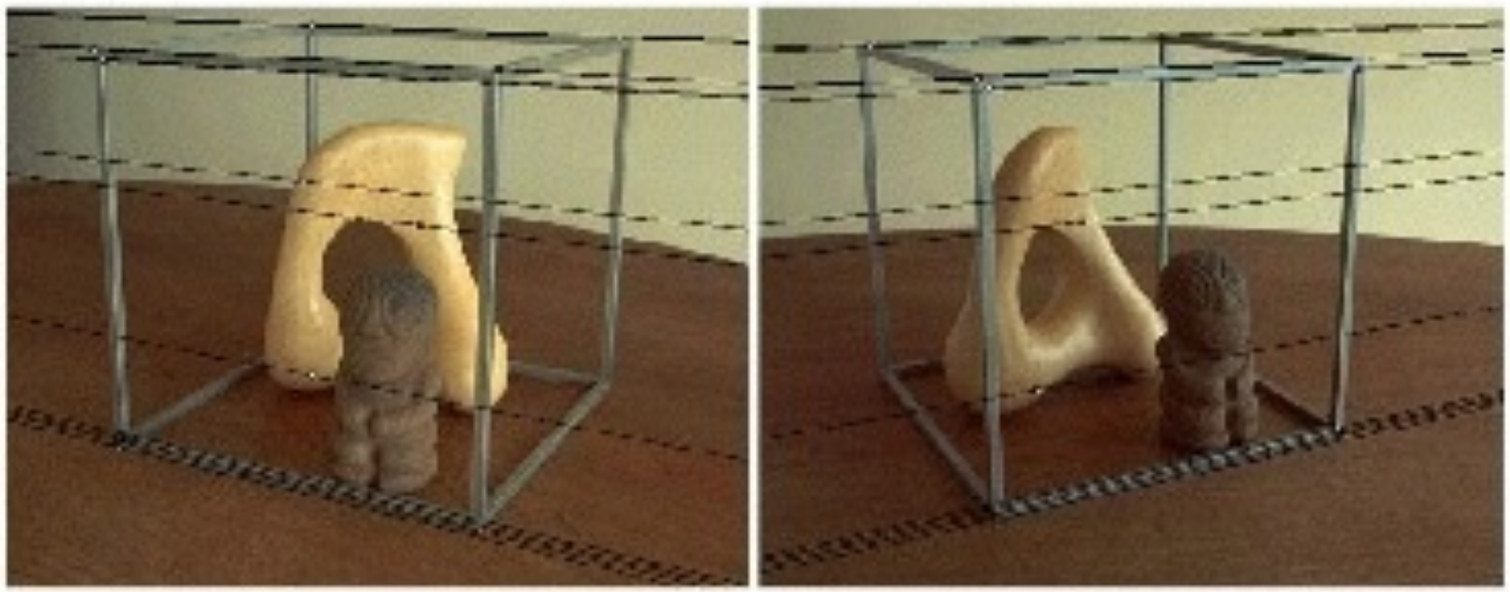
Resample lines (and shear/stretch) to place lines in correspondence, and minimize distortion





[Loop and Zhang, CVPR'99]

Rectification

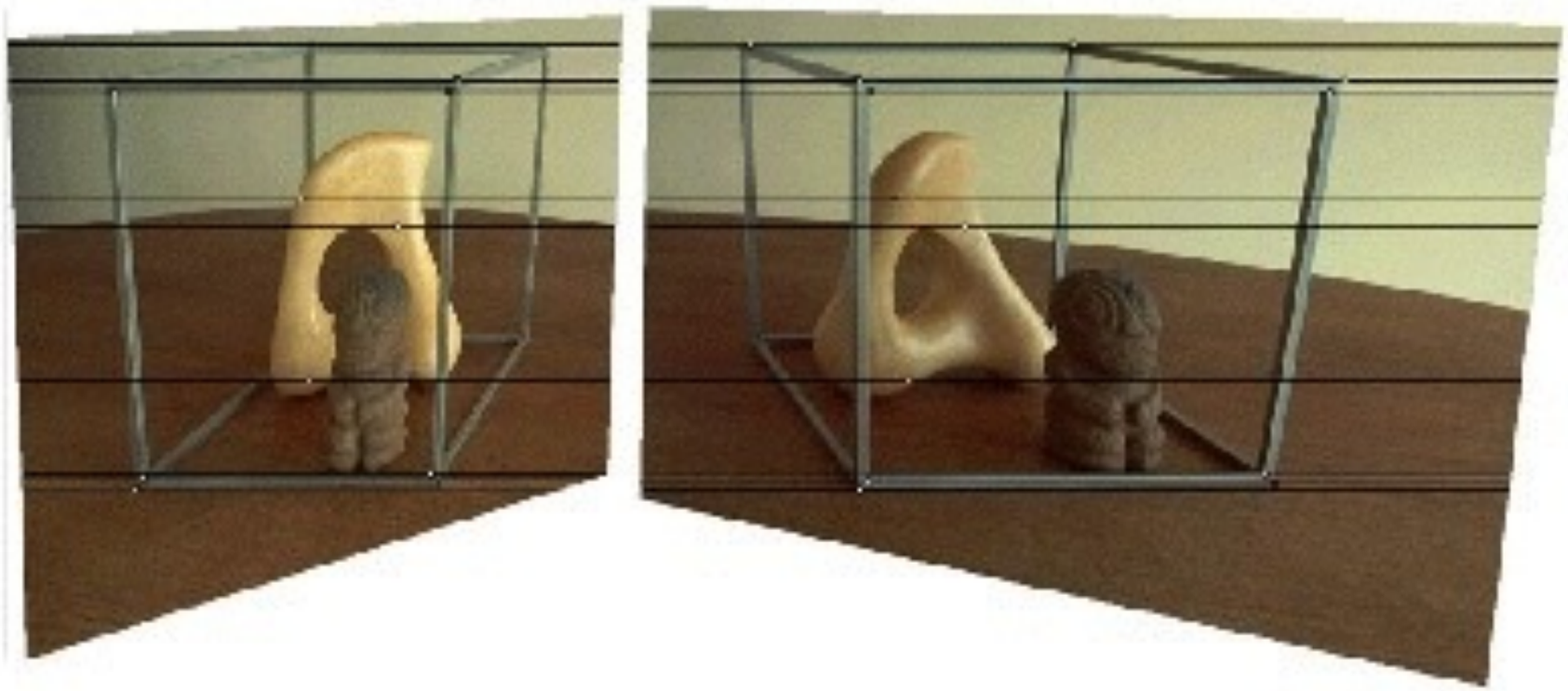


(a) Original image pair overlayed with several epipolar lines.

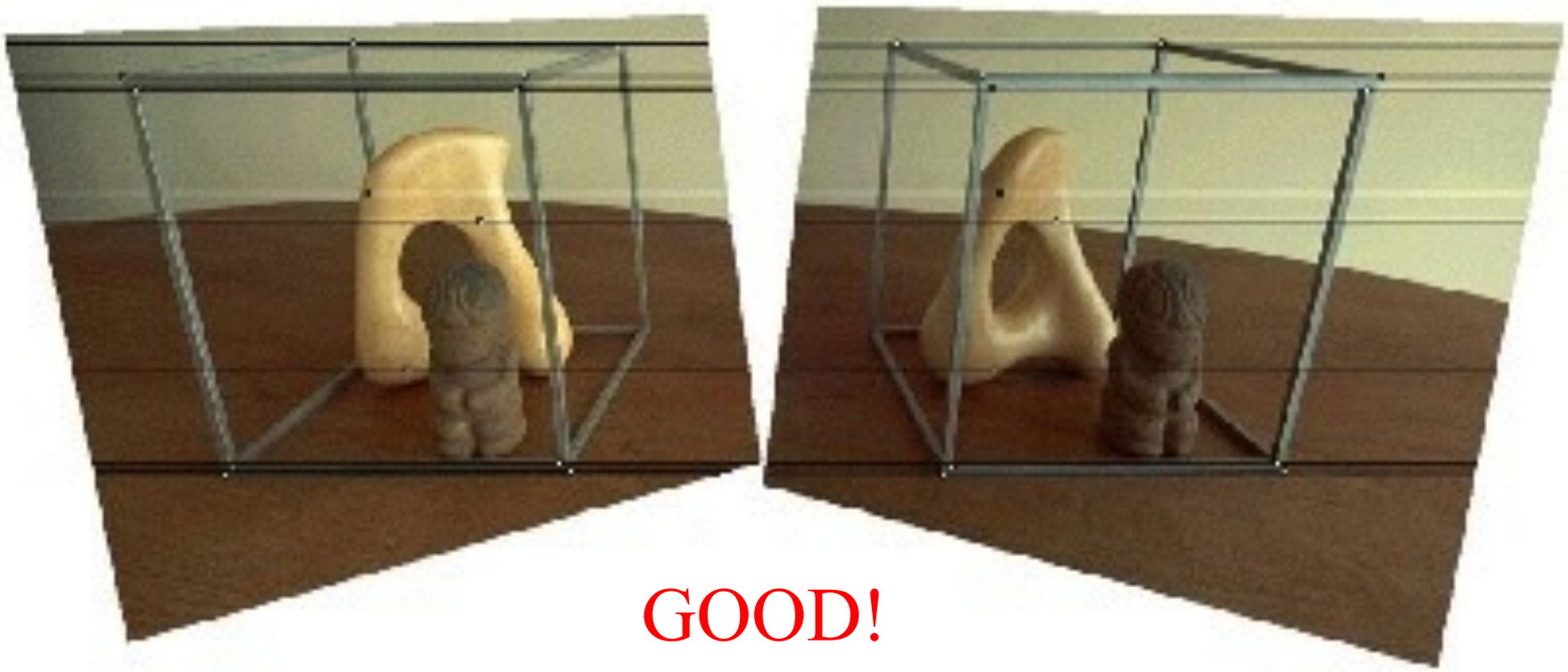
BAD!

(b) Image pair transformed by the specialized projective mapping $\mathbf{H}_p$ and $\mathbf{H}'_p$. Note that the epipolar lines are now parallel to each other in each image.

Rectification



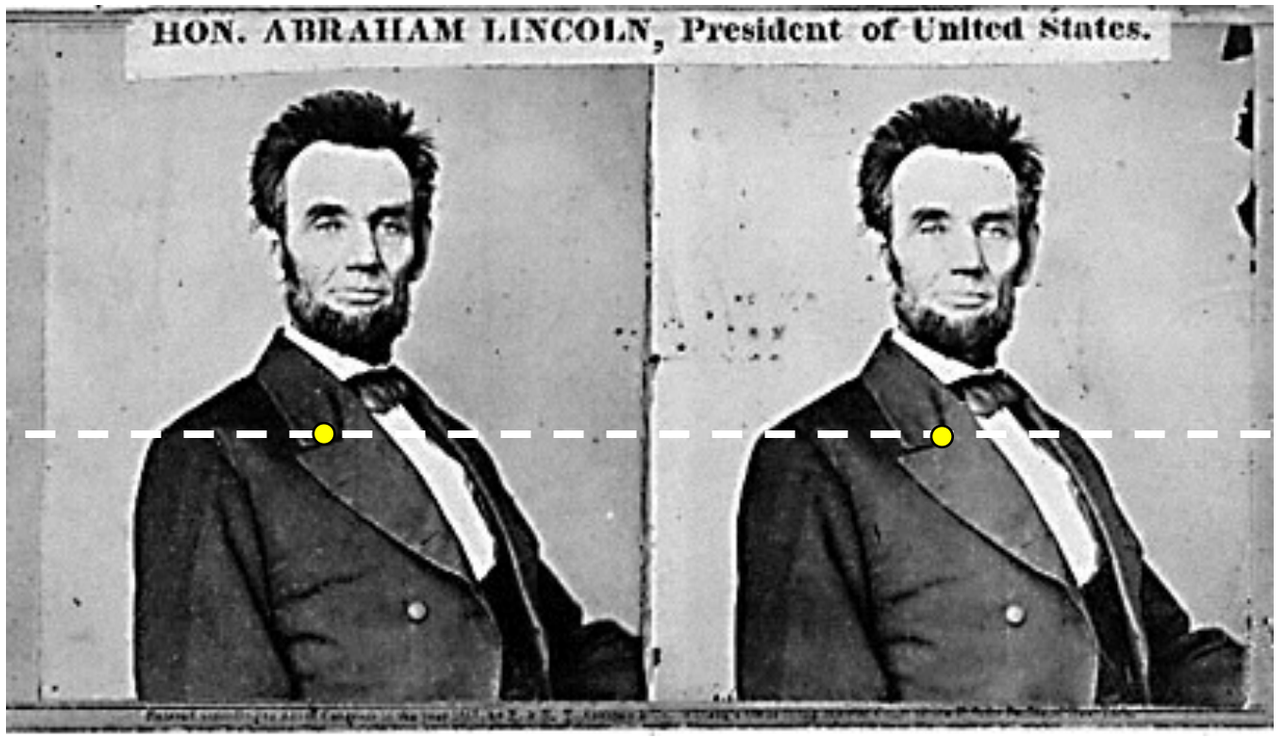
(c) Image pair transformed by the similarity $\mathbf{H}_r$ and $\mathbf{H}'_r$.. Note that the image pair is now rectified (the epipolar lines are horizontally aligned).



(d) Final image rectification after shearing transform $\mathbf{H}_s$ and $\mathbf{H}'_s$. Note that the image pair remains rectified, but the horizontal distortion is reduced.

GOOD!

Your basic stereo algorithm



For each epipolar line

For each pixel in the left image

- compare with every pixel on same epipolar line in right image
- pick pixel with minimum match cost

Improvement: match ***windows***

- This should look familar...