CS664 Lecture #15: Multi-camera stereo

Some material taken from:

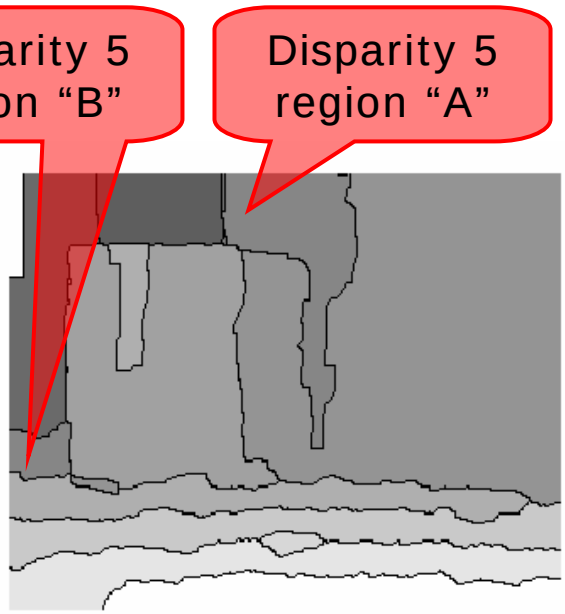
- **Steve Seitz, University of Washington**

http://www.cs.washington.edu/homes/seitz/

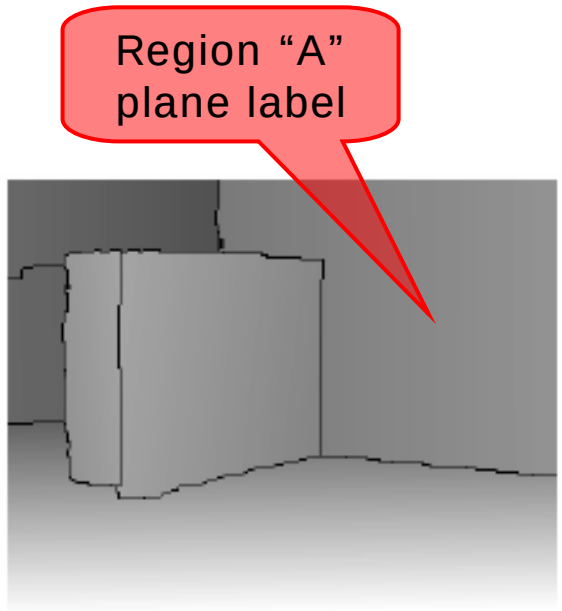
Announcements

- Pick a vision paper to write a report on by the next week (10/25)
  - Sources: CVPR, ICCV, ECCV, PAMI, IJCV
  - Email your choice to rdz@cs
- 1-page report should discuss the paper, list its assumptions, give some advantages and disadvantages
  - Report due on 11/15
- Next quiz: 10/25 (Tuesday)
  - Coverage through this lecture

Graph cuts + EM for sloped surfaces



Graph cut solution
(integer labels)



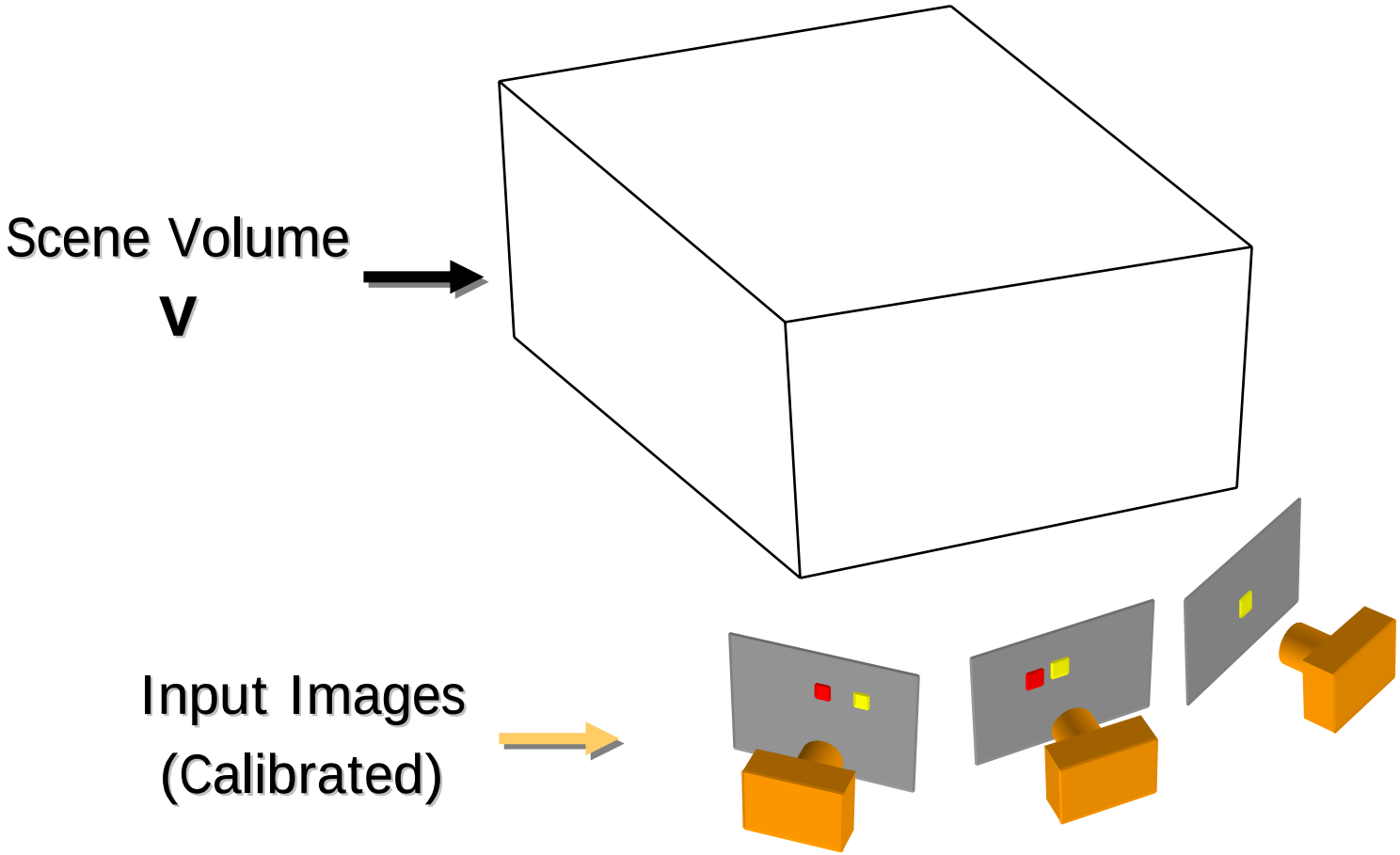
Graph cut solution
(plane labels)

Multicamera stereo

- Obvious generalization of stereo:
  - More than two cameras
- We saw this a little in “voxel occupancy”, which we solved with a binary graph cut
- There is a lot of work on this topic, and it naturally involves other important ideas
- We’ll look at some simple, elegant methods *not* based on energy minimization
  - And see how energy minimization can be used to improve them!

Multiview stereo

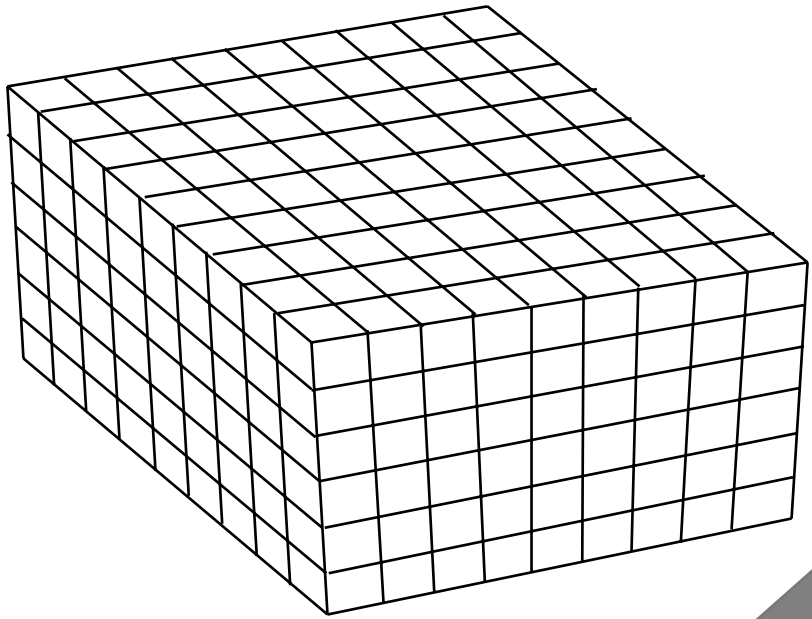
Volumetric stereo



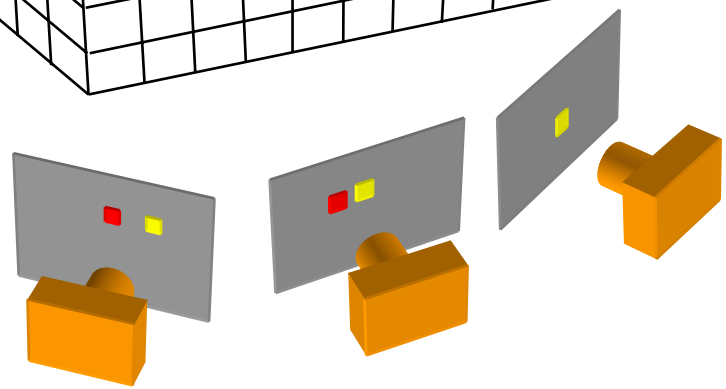
Goal: Determine occupancy, "color" of points in V

Discrete formulation: Voxel coloring

Discretized
Scene Volume



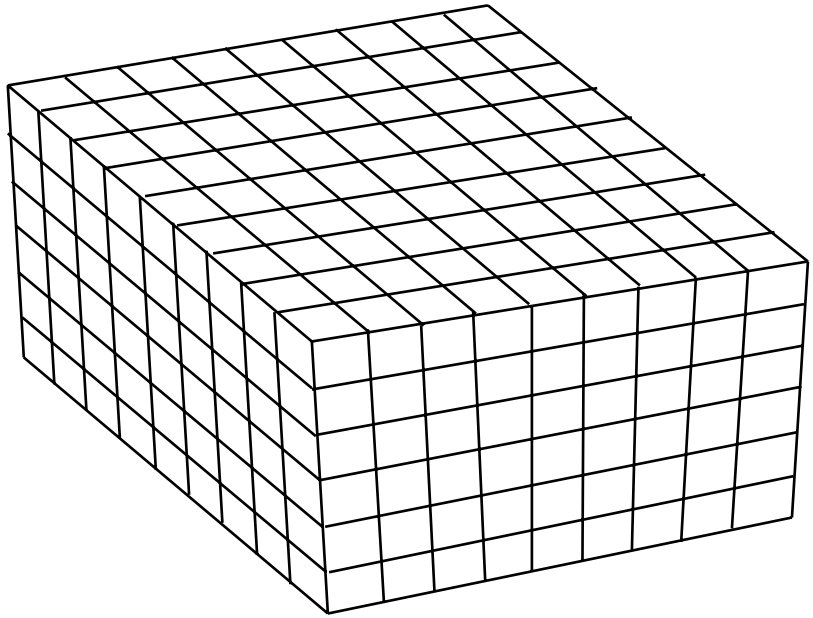
Input Images
(Calibrated)

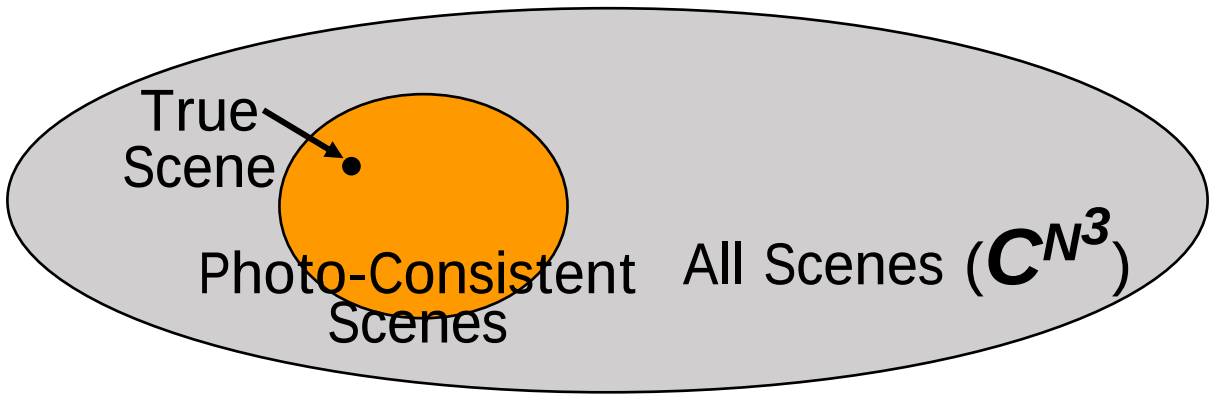


Goal: **Assign RGBA values to voxels in V**
photo-consistent **with images**

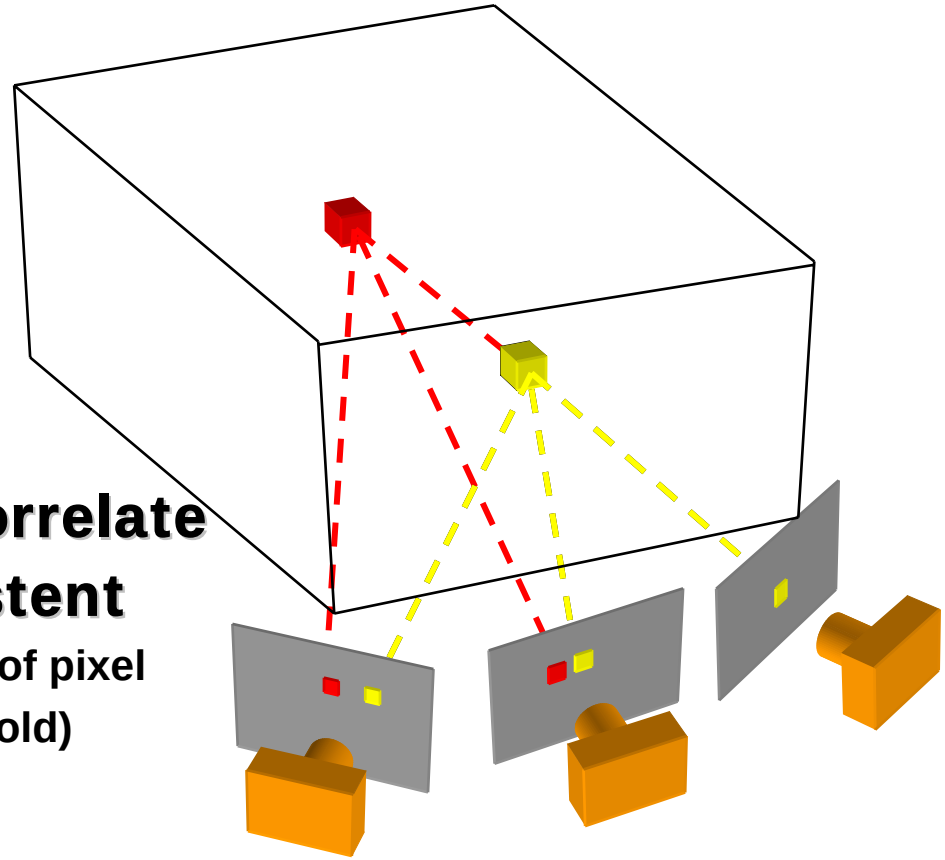
Complexity and computability

Discretized
Scene Volume
N^3 voxels
C colors





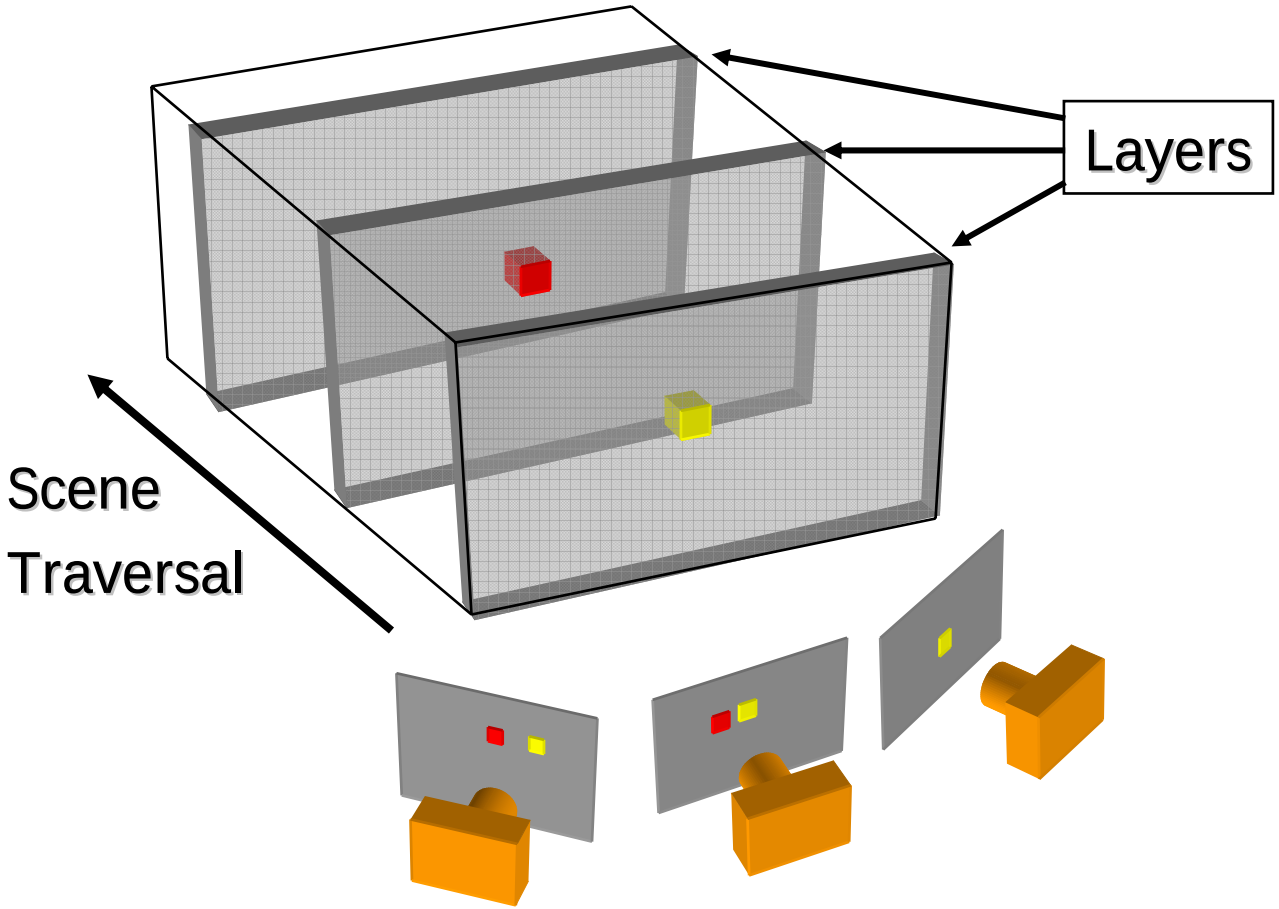
Voxel coloring (Seitz)



1. **Choose voxel**
2. **Project and correlate**
3. **Color if consistent**
   (standard deviation of pixel colors below threshold)

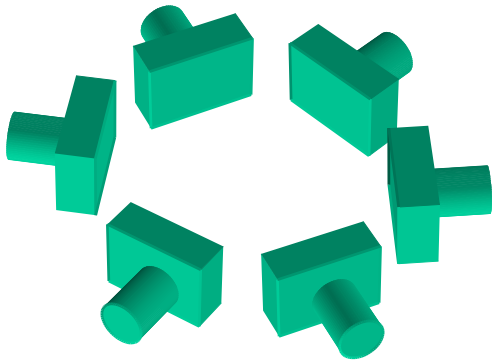
Visibility Problem: **in which images is each voxel visible?**

Depth ordering: occluders first!

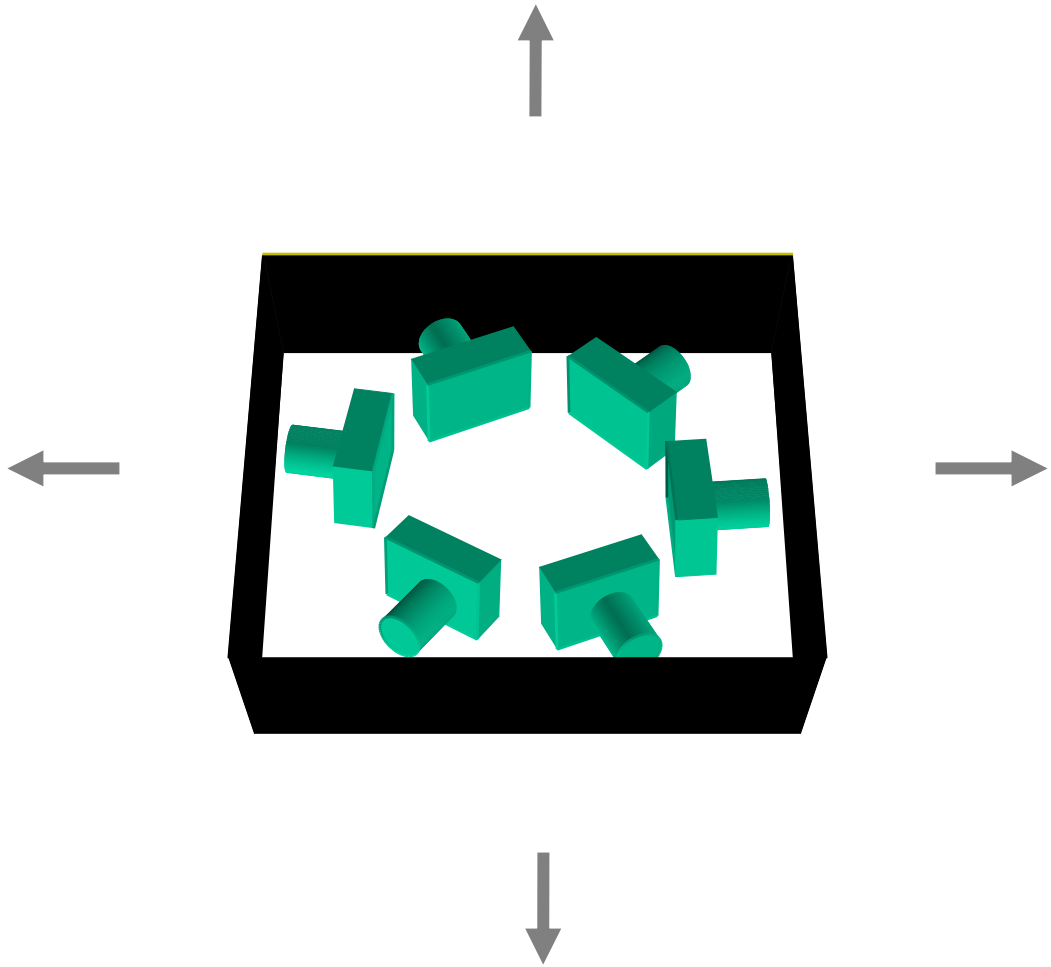


Condition: depth order is the ***same for all input views***

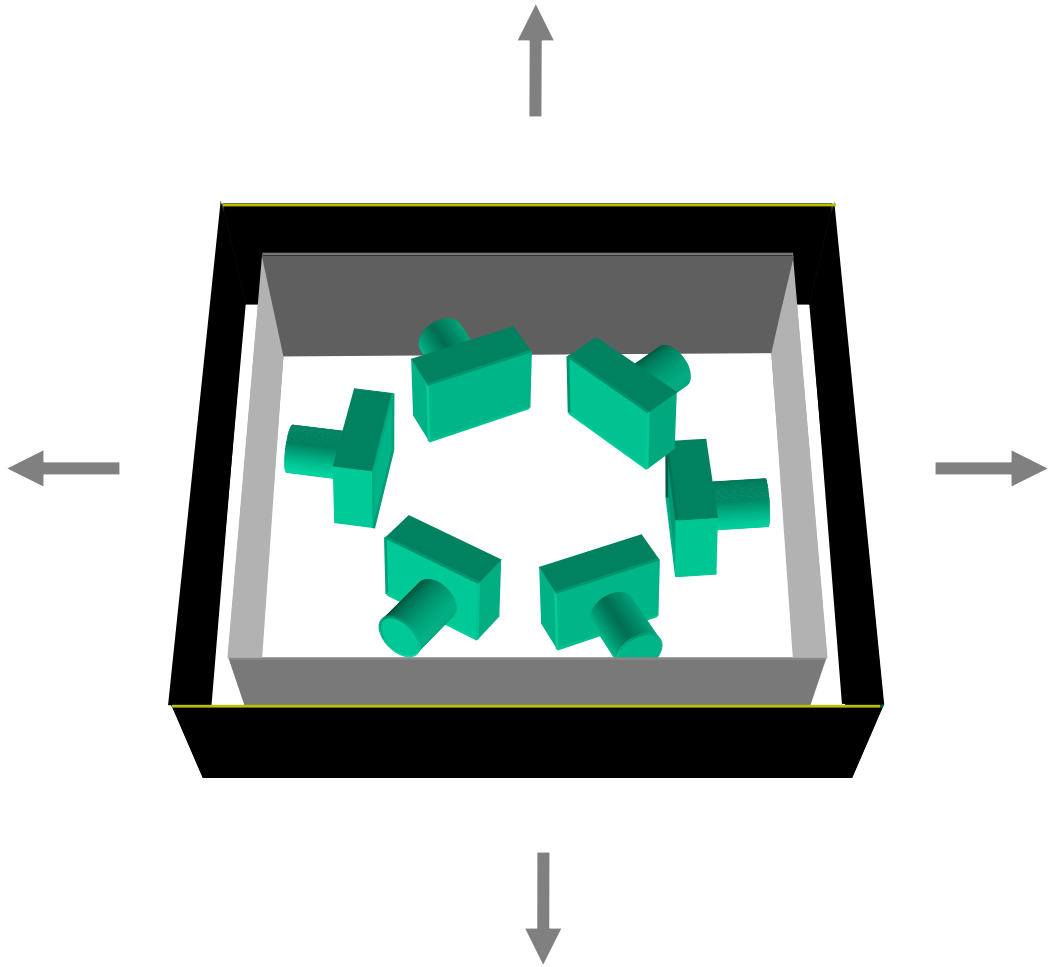
Panoramic Depth Ordering



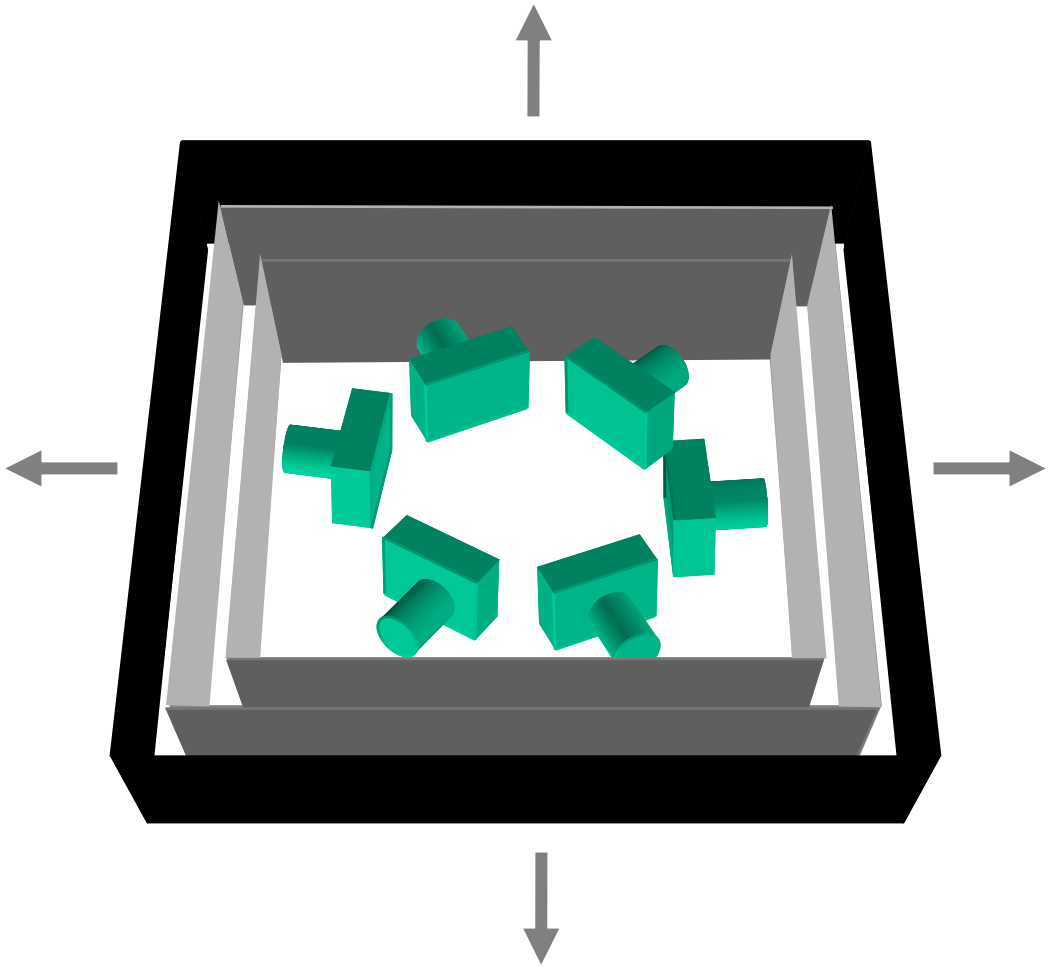
- Cameras oriented in many different directions
- Planar depth ordering does not apply



Layers radiate outwards from cameras

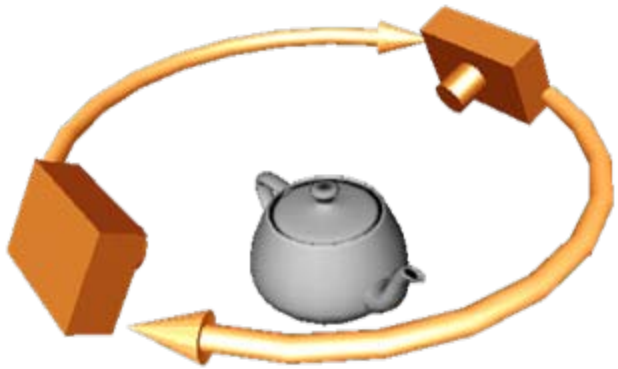


Layers radiate outwards from cameras



Layers radiate outwards from cameras

Compatible Camera Configurations



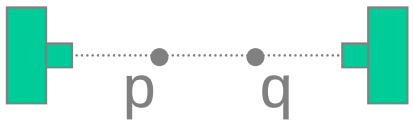
- *Inward-Looking*
  - *cameras above scene*
- *Outward-Looking*
  - *cameras inside scene*

Voxel Coloring Results

Dinosaur Reconstruction
72 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

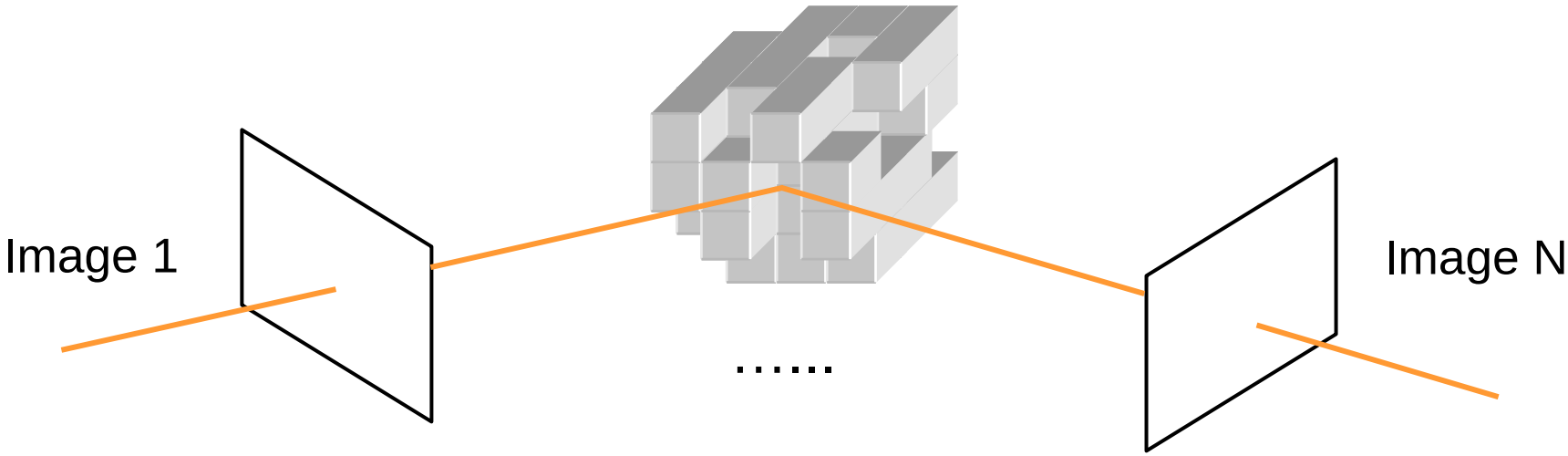
Flower Reconstruction
70 K voxels colored
7.6 M voxels tested
7 min. to compute
on a 250MHz SGI

Limitations of Depth Ordering



- A view-independent depth order may not exist
- Need more powerful general-case algorithms
  - Unconstrained camera positions
  - Unconstrained scene geometry/topology

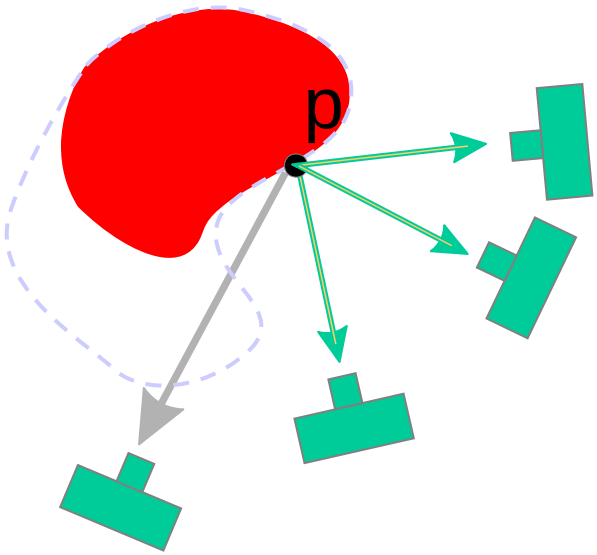
Space Carving Algorithm



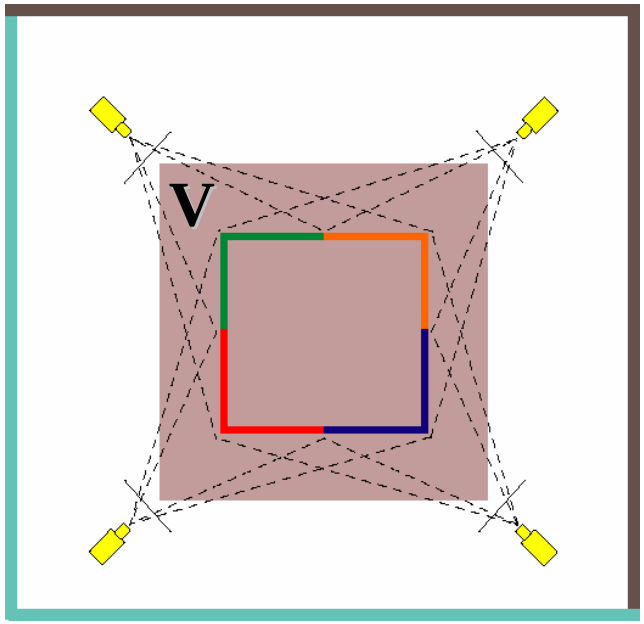
- Initialize to a volume V containing the true scene
- Choose a voxel on the current surface
- Project to visible input images
- Carve if not photo-consistent
- Repeat until convergence

Convergence

- Consistency Property
  - The resulting shape is photo-consistent
    - all inconsistent points are removed
- Convergence Property
  - Carving converges to a non-empty shape
    - a point on the true scene is *never* removed



Which shape do you get?



True Scene

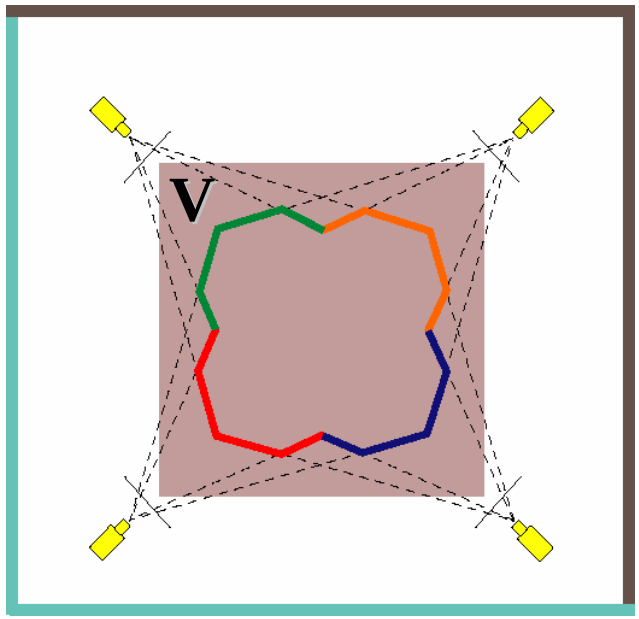


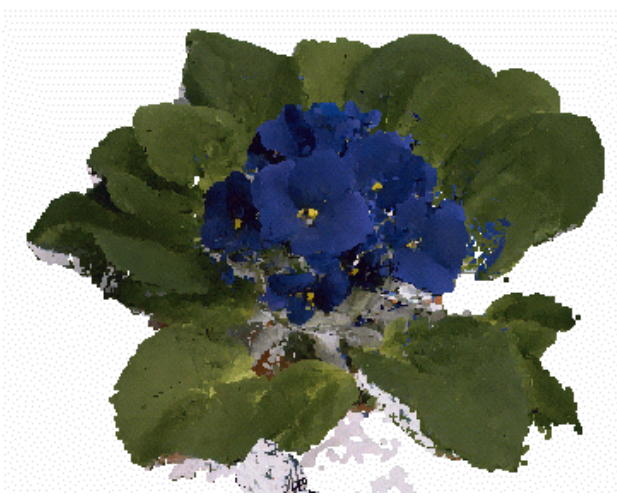
Photo Hull

- The Photo Hull *is the UNION of all photo-consistent scenes in V*
  - It is a photo-consistent scene reconstruction
  - Tightest possible bound on the true scene

Space Carving Results: Violet

Input Image (1 of 45)

Reconstruction



Reconstruction

Reconstruction

Space Carving Results: Hand

Input Image
(1 of 100)

Views of Reconstruction

Why use energy minimization?

- Very similar to voxel occupancy example
  - Early hard decisions can be wrong
    - Difficult to recover from them
- No use of spatial smoothness!
  - Almost nothing in vision actually works unless it uses spatial smoothness
  - No one really understands why voxel coloring and space carving are exceptions
  - Very active area of research

Multi-Camera Scene Reconstruction via Graph Cuts
(Kolmogorov & Zabih, ECCV ’02)

Comparison with stereo

- Much harder problem than stereo
- In stereo, most scene elements are visible in both cameras
  - It is common to ignore occlusions
- Here, almost no scene elements are visible in all cameras
  - Visibility reasoning is vital

Key issues

- Visibility reasoning
- Incorporating spatial smoothness
- Computational tractability
  - Only certain energy functions can be minimized using graph cuts!
- Handle a large class of camera configurations
- Treat input images symmetrically

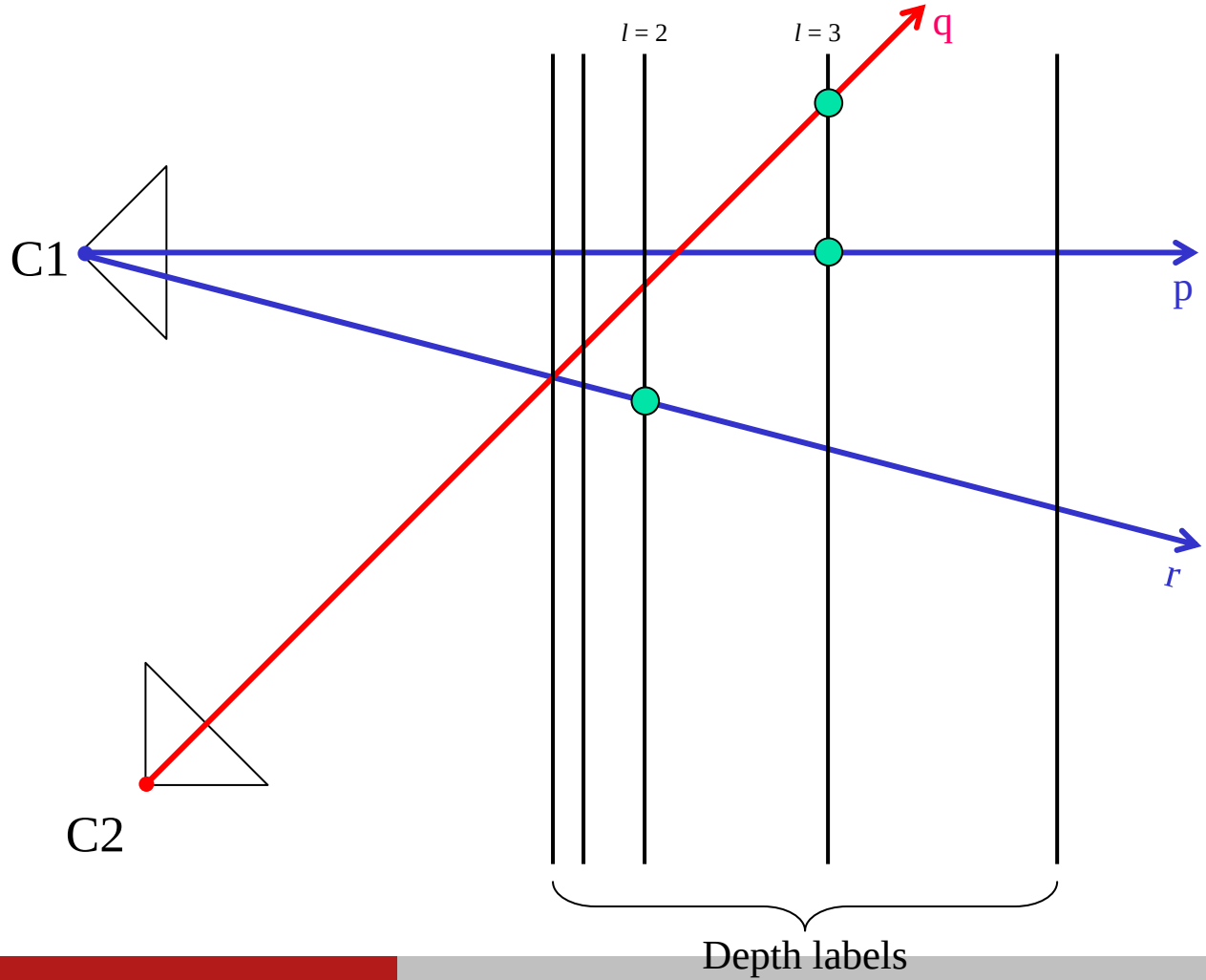
Approach

- Problem formulation
  - Discrete labels, not voxels
  - Carefully constructed energy function
- Minimizing the energy via graph cuts
  - Local minimum in a strong sense
  - Use the regularity construction
- Experimental results
  - Strong preliminary results

Problem formulation

- Discrete set of labels corresponding to different depths
  - For example, from a single camera
- Camera pixel plus label = 3D point
- Goal: find the best configuration
  - Labeling for each pixel in each camera
  - Minimize an energy function over configurations
  - Finding the exact minimum is NP-hard

Sample configuration

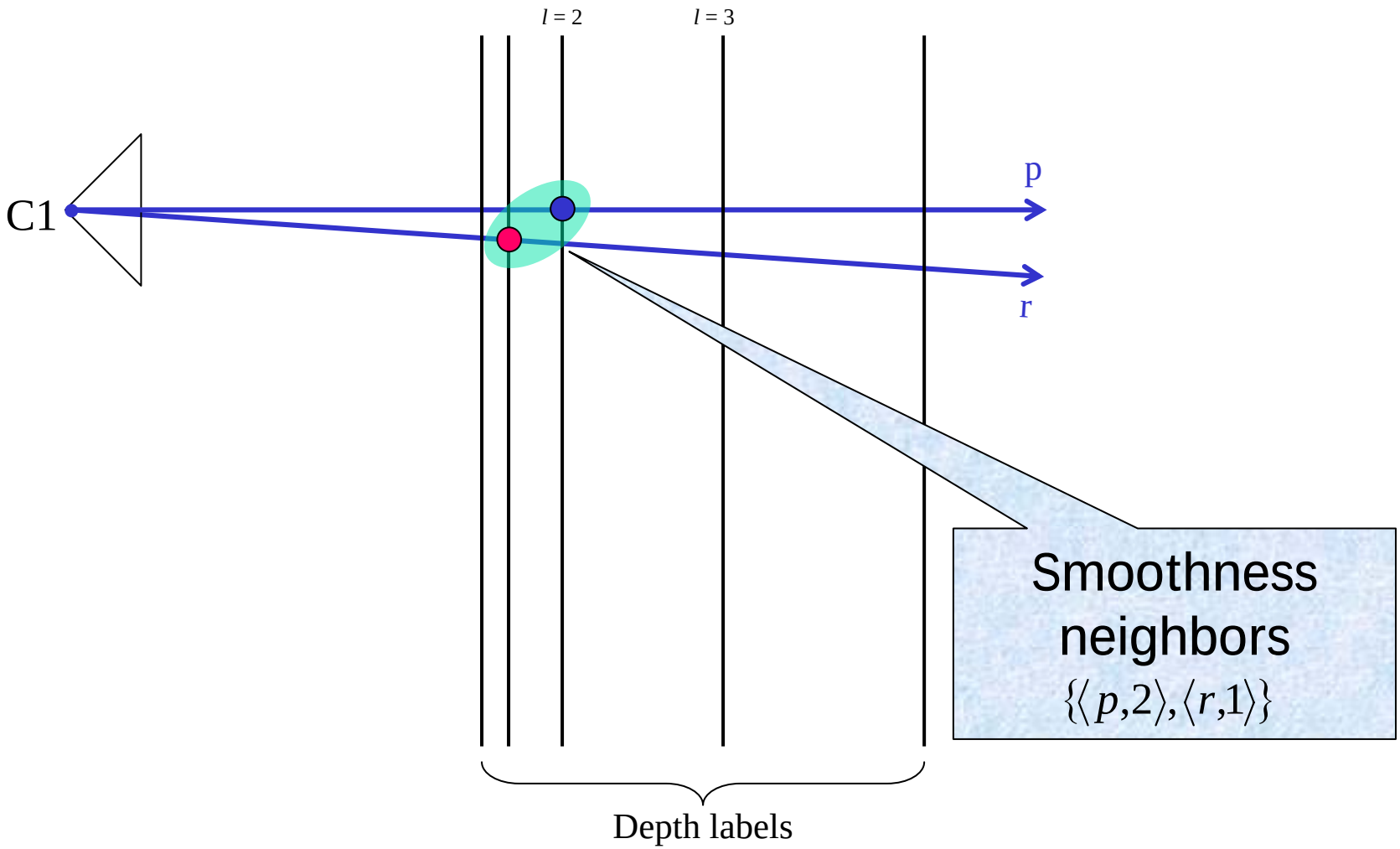


$\{\langle p,3\rangle,\langle r,2\rangle,\langle q,3\rangle\}$
configuration

Energy function has 3 terms: smoothness, data, visibility

- Neighborhood systems involve 3D points
- Smoothness: spatial coherence (within camera)
- Data: photoconsistency (between cameras)
  - Two pixels looking at the same scene point should see similar intensities
- Visibility: prohibit certain configurations (between cameras)
  - A pixel in one camera can have its view blocked by a scene element visible from another camera

Smoothness neighborhood



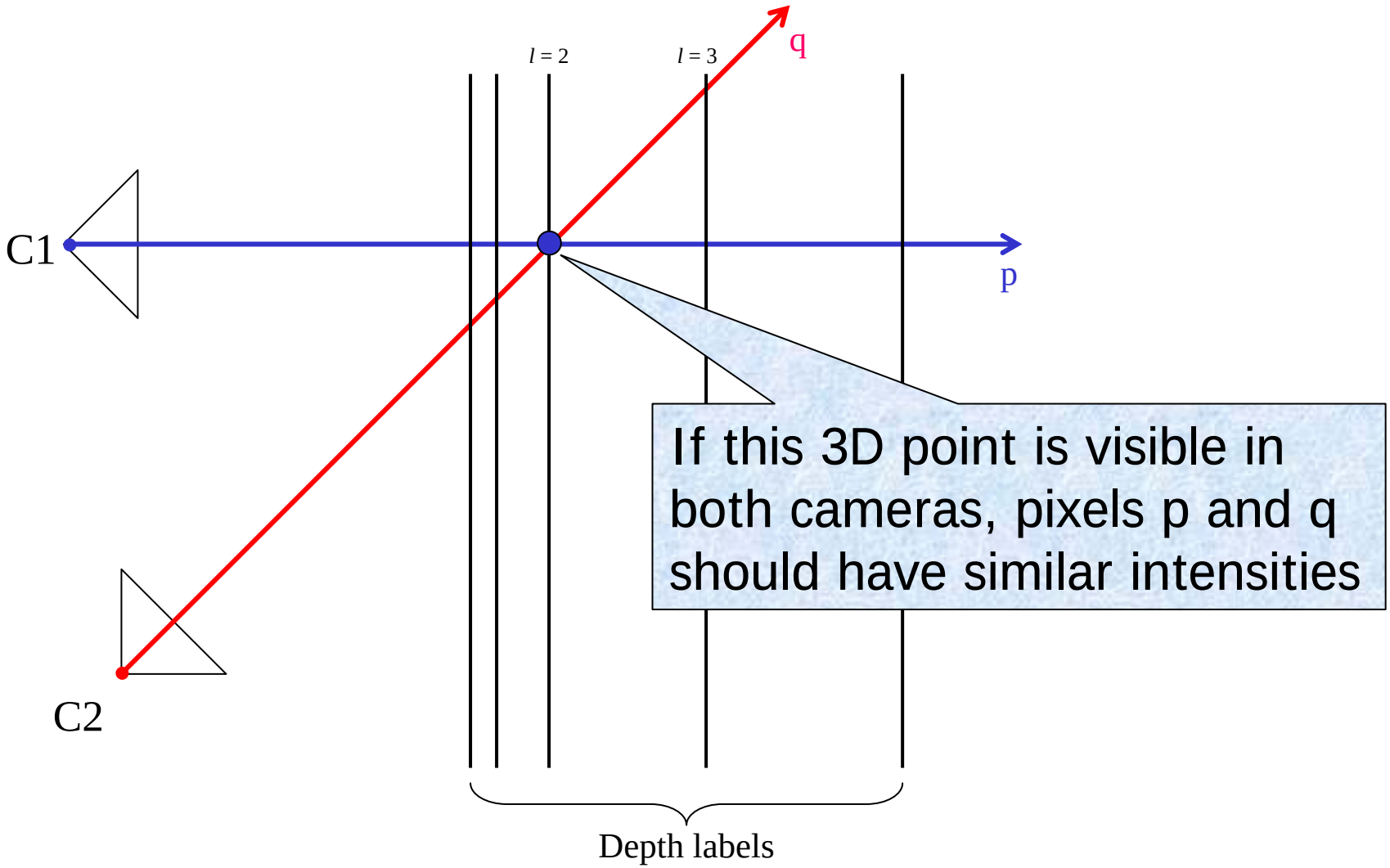
Smoothness term

- Smoothness neighborhood involves pairs of 3D points from the same camera
  - We'll assume it only depends on a pair of labels for neighboring pixels
    - Usual 4- or 8-connected system among pixels
- Smoothness penalty for configuration f is

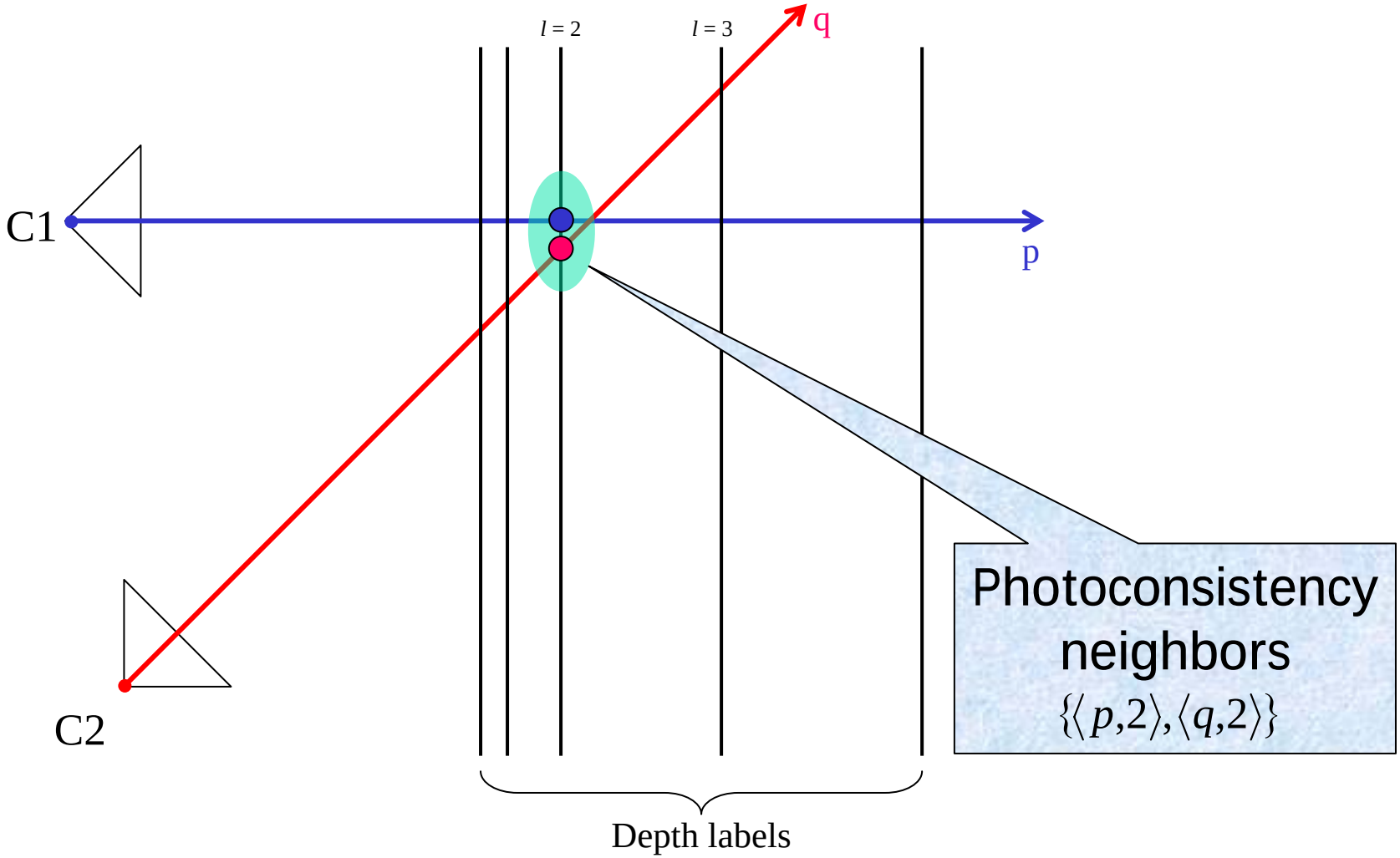
$$\sum_{\{p,q\} \in N_{smooth}} V(f_p, f_q)$$

  - V must be a metric, i.e. robustified L_1 (regularity)

Photoconsistency constraint



Photoconsistency neighborhood



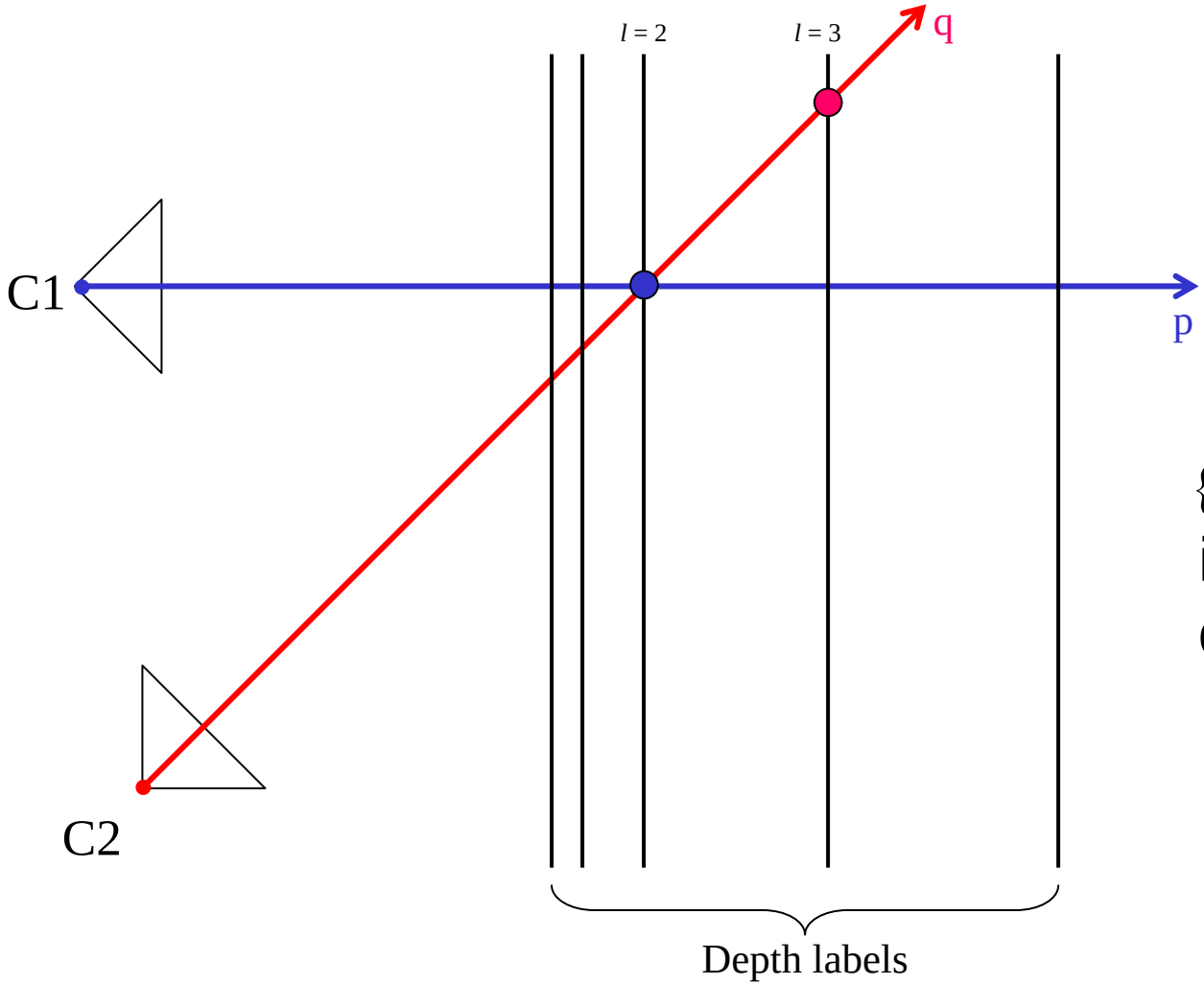
Data (photoconsistency) term

- Photoconsistency neighborhood N_{photo}
  - Arbitrary set of pairs of 3D points (same depth)
  - Implementation: $\{\langle p,f_p\rangle,\langle q,f_q\rangle\} \in N_{photo}$ if the projection of $\langle p,f_p\rangle$ on C2 is nearest to q
- Our data penalty for configuration f is

$$\sum_{\substack{\{\langle p,f_p\rangle,\langle q,f_q\rangle\} \\ \in N_{photo}}} \min\left[\left(I_p - I_q\right)^2 - K, 0\right]$$

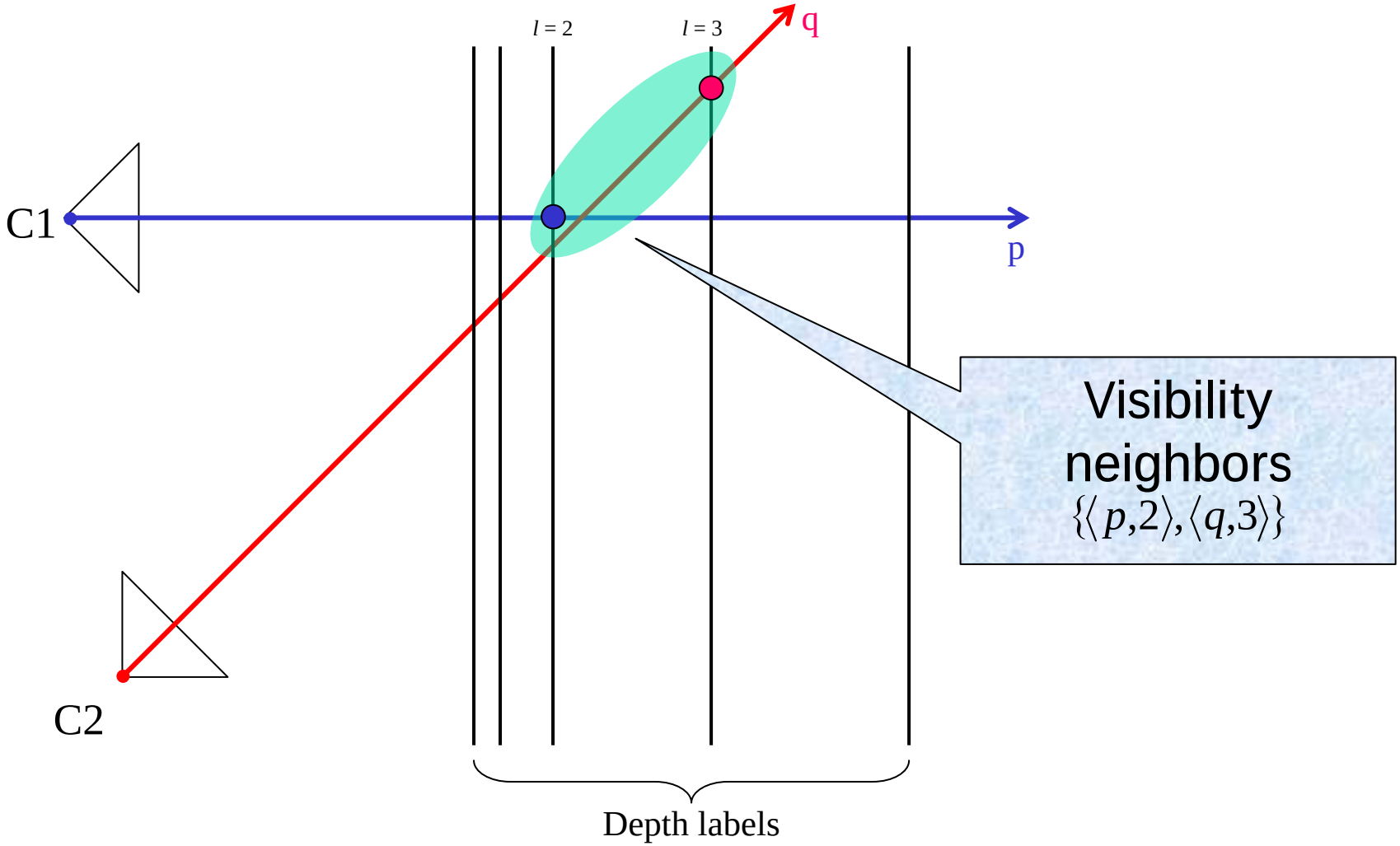
  - Negative for technical reasons (regularity)

Visibility constraint



$\{\langle p,2\rangle,\langle q,3\rangle\}$
is an impossible configuration

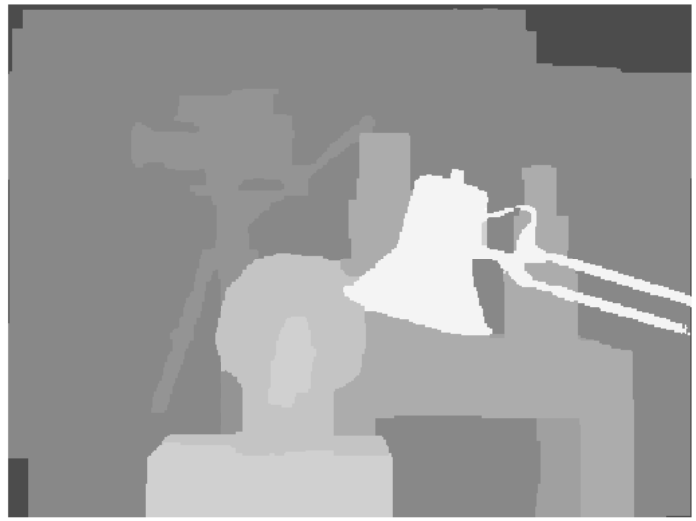
Visibility neighborhood



Visibility term

- Visibility neighborhood N_{vis} is all pairs of 3D points that violate the visibility constraint
  - Arbitrary set of pairs of points at different depths
    - Needed for regularity
  - The pair of points come from different cameras
  - Current implementation: based on the photoconsistency neighborhood
- A configuration containing any pair of 3D points in the visibility neighborhood has infinite cost

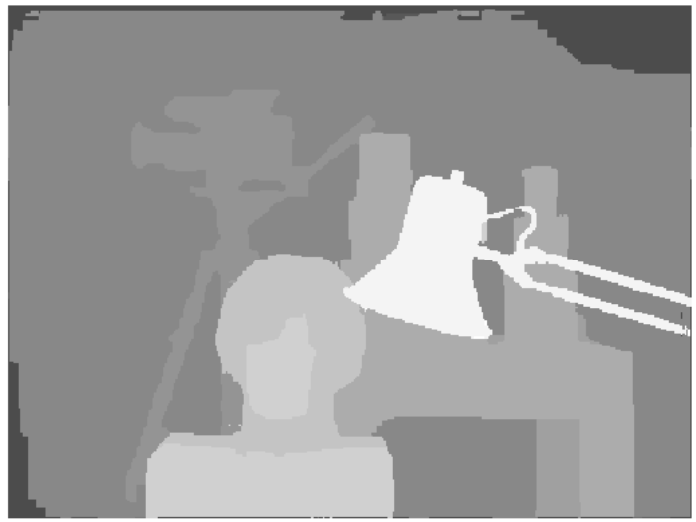
Tsukuba images



Our results, 4 interactions

Comparison

Best results [SS ’02]



Our results, 10 interactions