TA Section 7

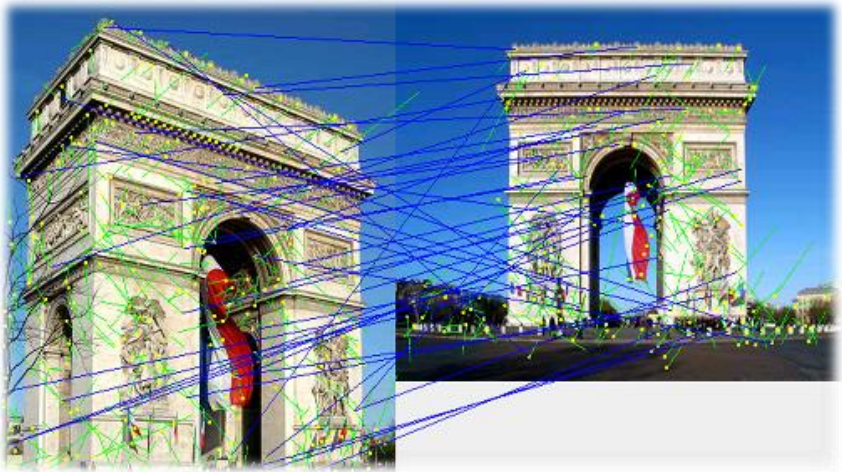
Problem Set 3

- SIFT (Lowe 2004)
- Shape Context (Belongie et al. 2002)
- Voxel Coloring (Seitz and Dyer 1999)

Distinctive Image Features from Scale-Invariant Keypoints

David Lowe 2004

- Still the best keypoint descriptor a decade after publication



SIFT Keypoint Descriptor

- Vector representation of a point in an image

We want:

- The same point in different images generates a similar descriptor
- Different points generate different descriptors

SIFT Keypoint Descriptor

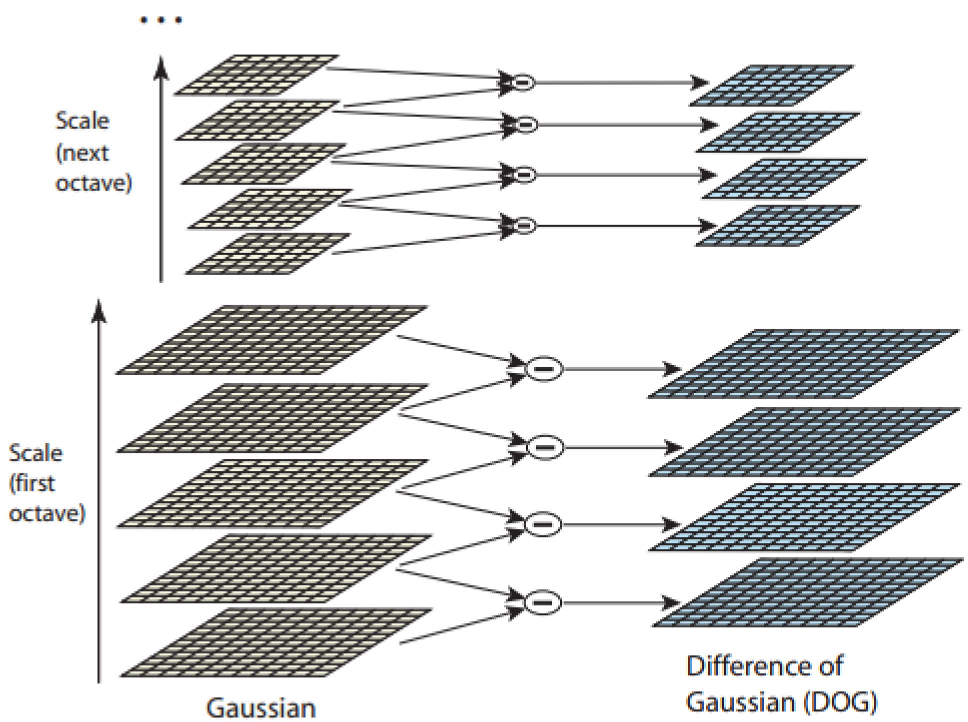
- Vector representation of a point in an image

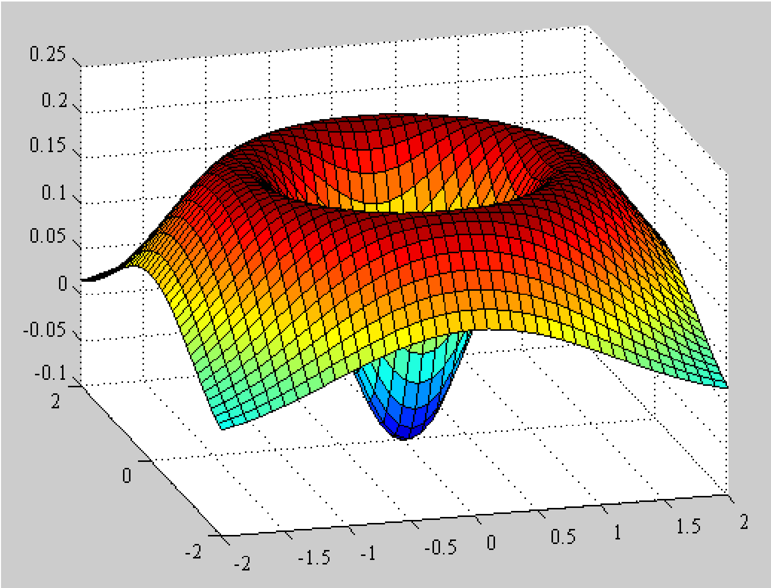
We want:

- The same point in different images generates a similar descriptor
- Different points generate different descriptors

Finding Keypoints

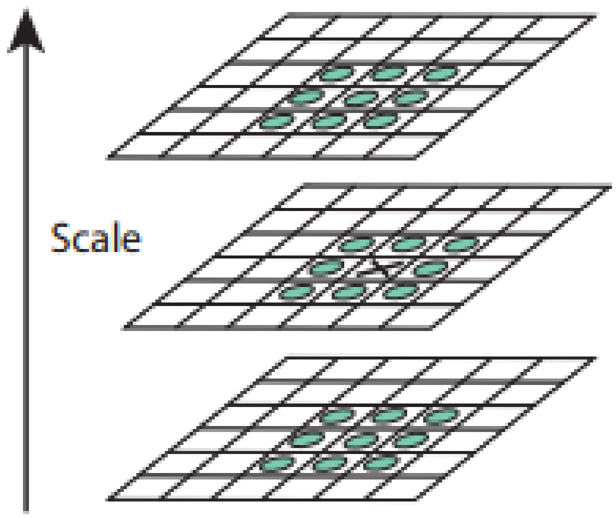
First get scale invariance – Difference of Gaussians





Finding Keypoints

We want to know not just the position, but some sense of 'scale' of a keypoint.



Extrema of DoG pyramid (in x,y,scale space) become keypoint candidates

Localizing the Keypoint

For further accuracy, we fit a 3D quadratic to the DoG values using a Taylor approximation.

The extremum of the quadratic defines the location of the keypoint accurate to fractions of a pixel.

Lowe shows this dramatically increases the match quality.

Eliminating Poor Keypoints

We want keypoints to be easily localizable in scale space so we want well-defined extrema

Can use the Hessian of the quadratic approximation to determine curvature in x and y directions.

Reject keypoints where the curvature in one direction is much smaller than the curvature in the other.

Computing Gradients

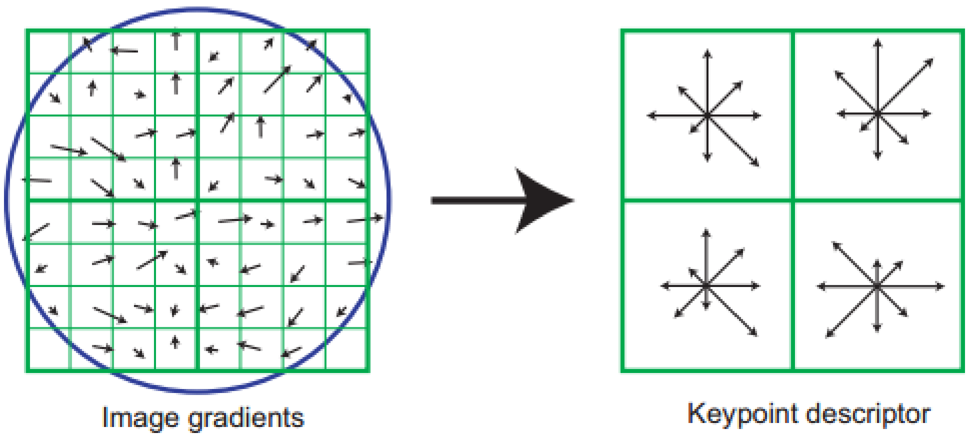
$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Gradients are computed using the keypoint scale, ensuring scale invariance.

Ie we compute gradients in a blurred image.

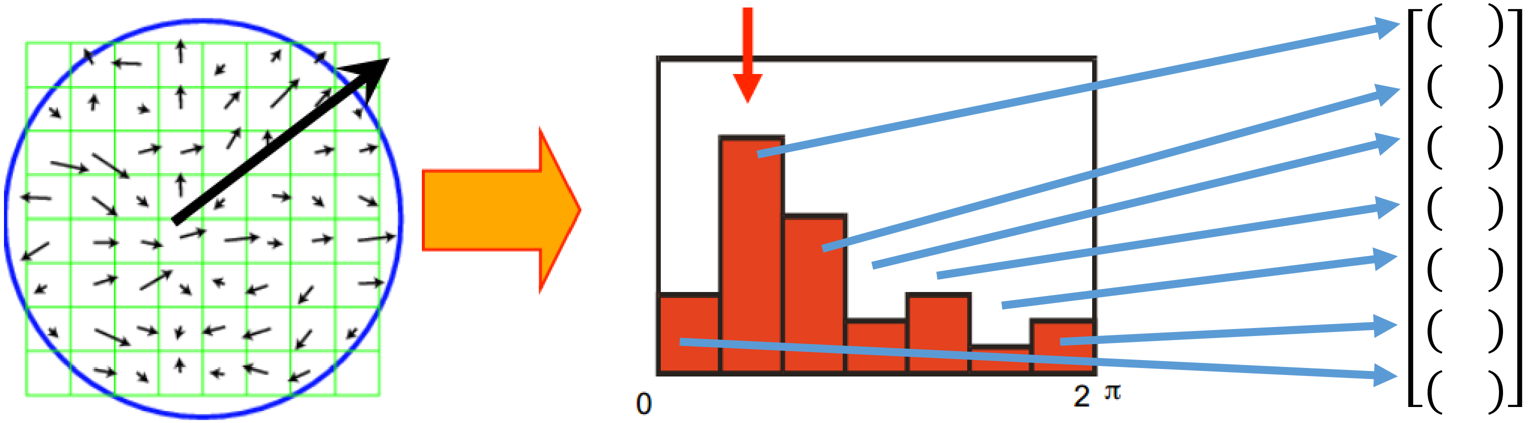
Constructing the Descriptor



Build histograms of orientations for cells around the keypoint, weighted by gradient magnitude and distance from center.

16 cells of 4x4 pixel patches times 8 gradient bins per cell equals 128 values per descriptor.

Rotation/Illumination invariance

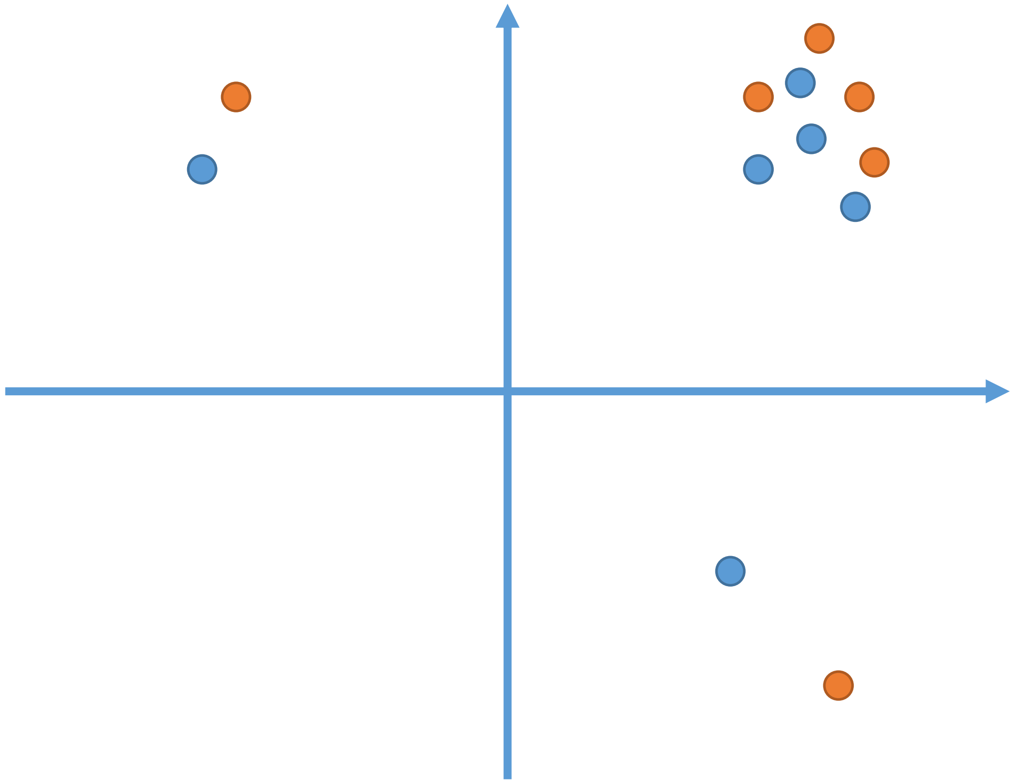


Canonical orientation is peak of orientation histogram over entire patch

This bin is first in feature vector description

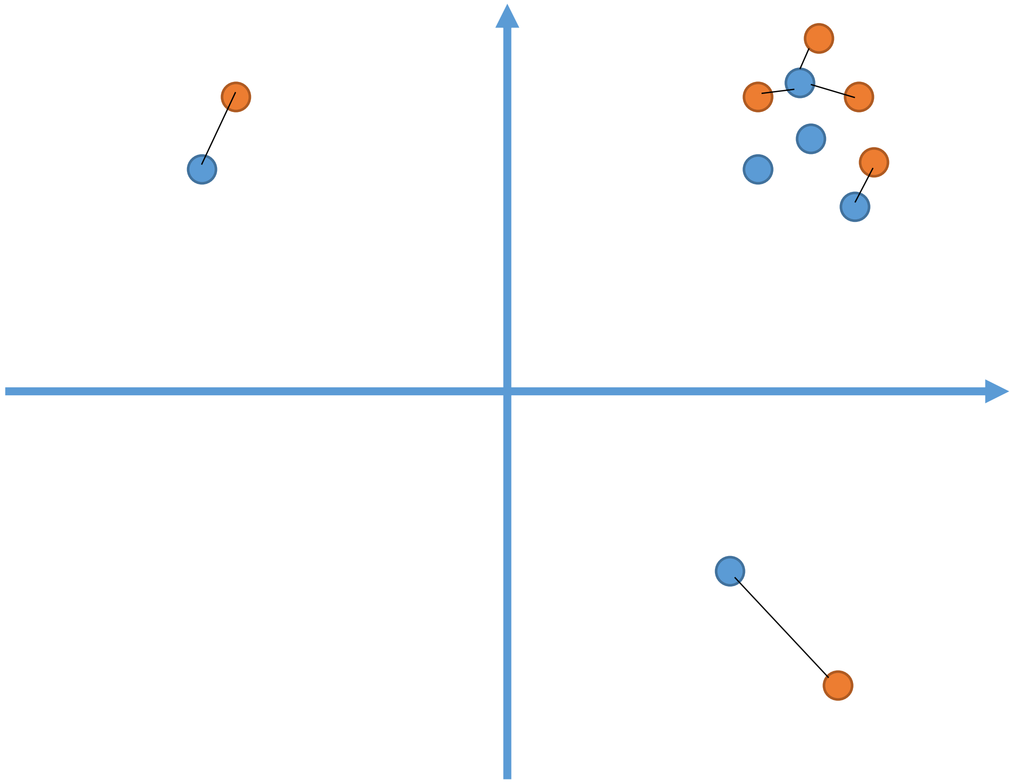
128D vector is normalized for illumination invariance

Keypoint Matching



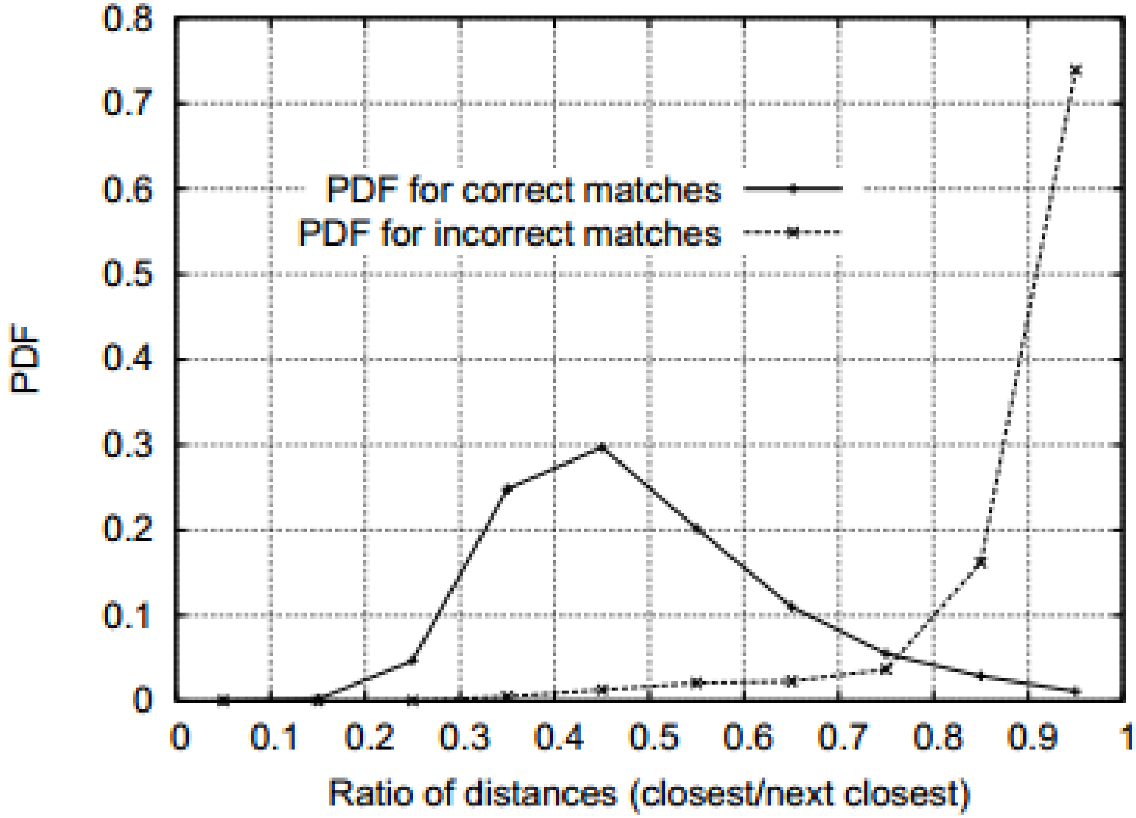
Find nearest neighbors in feature space

Keypoint Matching



Quality of match depends not just on distance between points but how cluttered the local region is.
We compare with distance to 2^{nd} closest neighbor.

Keypoint Matching



Keep match if $||x - n_1|| < 0.8||x - n_2||$

Keypoint Matching

Notes for homework:

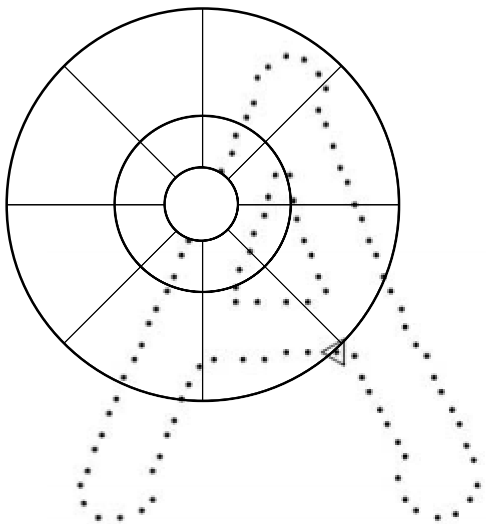
- Do exhaustive (brute-force) search for nearest neighbors, not the Best-Bin-First method described in Lowe’s paper

- A keypoint in one image may match to multiple keypoints in the other, or none.

SIFT Questions?

Shape Matching and Object Recognition via Shape Contexts

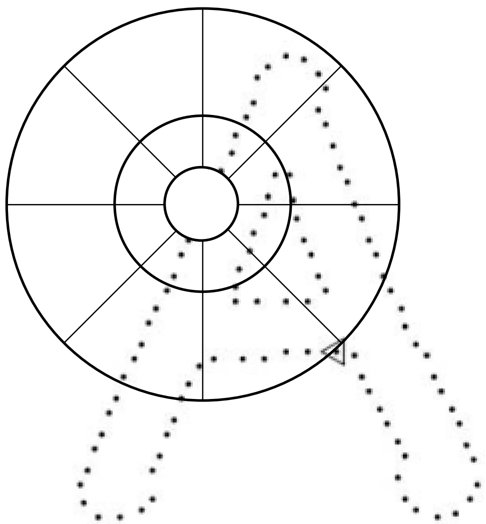
Belongie, Malik and Puzicha 2002



Describe a point on a shape by a histogram of edges around it

Match shapes by finding corresponding points and a deformation to align them

Constructing the Descriptor

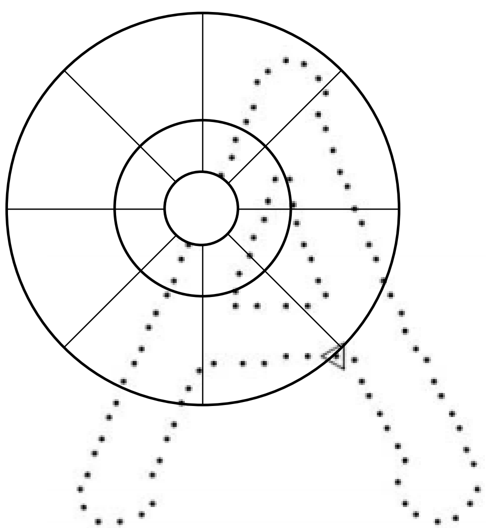


Represent objects as a set of points sampled from the edges

Pick a point to create a descriptor for

Create a (radius, angle) histogram for vectors from the query point to every other point

Shape Context Notes:



Histogram is in log-polar space, so the bins are more fine-grained near the center. Use *logspace* in Matlab

Not rotation invariant – otherwise impossible to tell 6 from 9.

Outliers not included in histograms, still get a shape context histogram for themselves

Can match shape contexts using chi-squared score (not in homework)

Shape Context Notes:

```
This code should populate the the radius bins using
r_bin_edges as bin edges. For each radius bin, find the
corresponding radii that fall within that bin and increment
the corresponding histogram counts.

The array that should be populated is r_array_q.
Note that each bin contains all radii within the bin's radius,
so you are going to be double counting the radii, for example,
the outermost bin will include radii of all of the other bins.
```

Somewhat confusing comment in the code we gave you.

r_array_q is nsamp x nsamp array containing the radius bin index of the point. *One possible way* to compute the array is to count the number of times a point falls within a bins radius.

This will result the outer bin having index 1, and the inner bin having index nbins_r.

Photorealistic Scene Reconstruction by Voxel Coloring

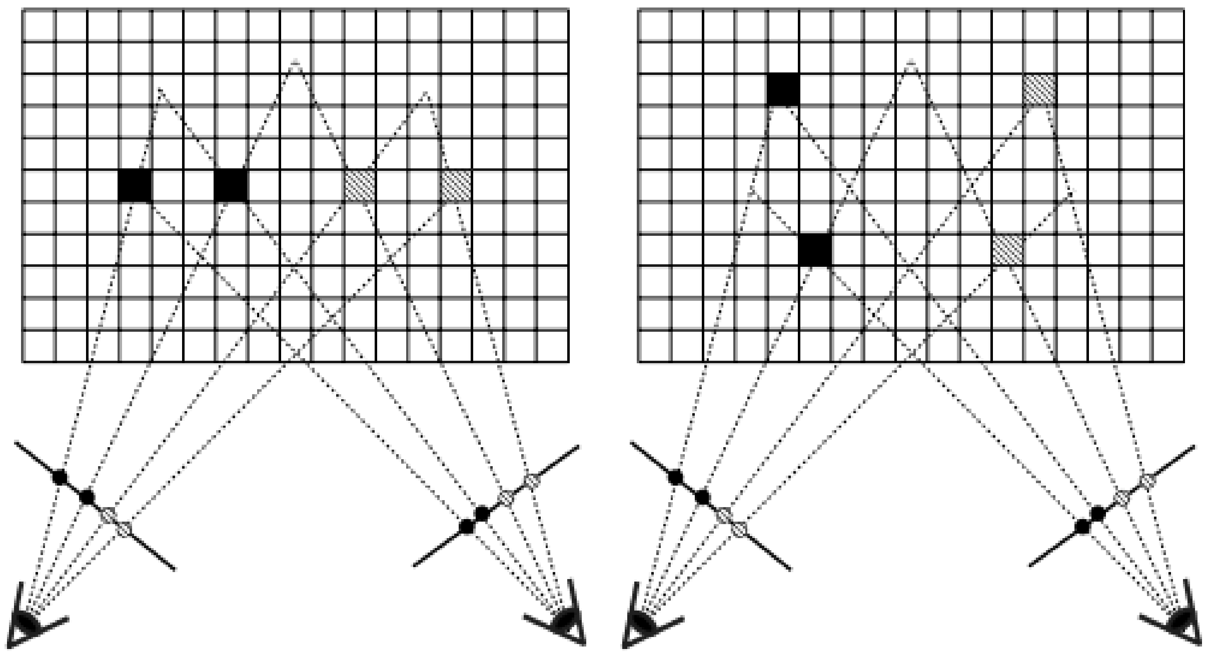
Seitz and Dyer 1999

- Works really well under controlled conditions
- Doesn't really work otherwise!



Voxel Coloring

Find a reconstruction that has consistent shape and color given a collection of images.



Under defined problem! More than one consistent solution. Need more constraints

Ordinal Visibility Constraint

Need to quickly determine if a voxel is not occluded

We find a function $D : \mathbb{R}^3 \rightarrow \mathbb{R}^+$ such that:

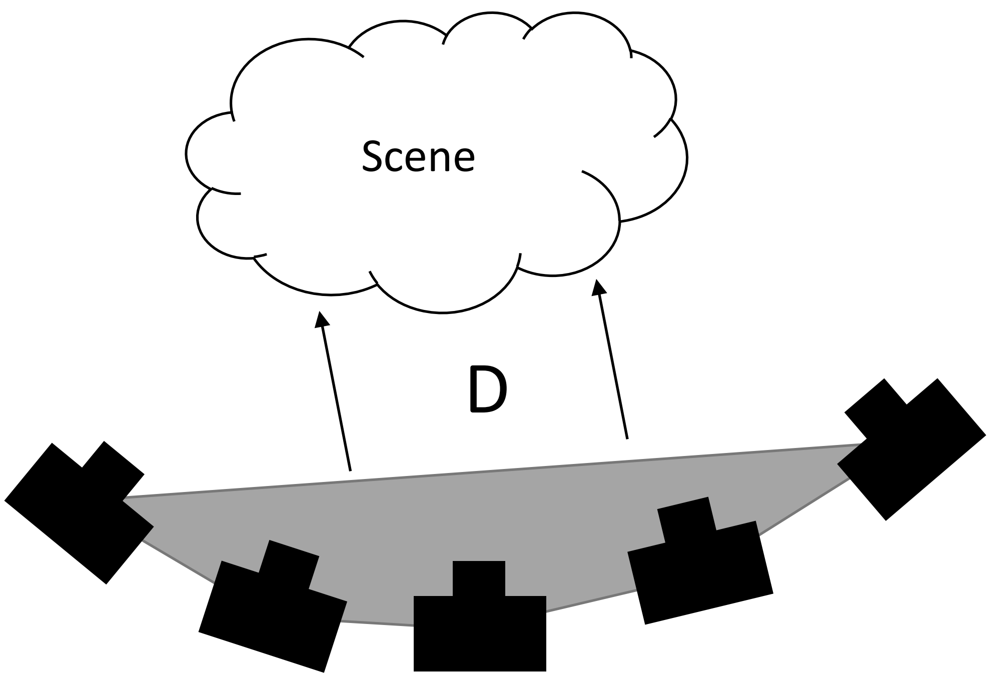
Voxel A occludes B $\implies D(A) < D(B)$

Conversely, if D(P) < D(Q) then Q cannot occlude P

Ordinal Visibility Constraint

D does not exist for all camera setups.

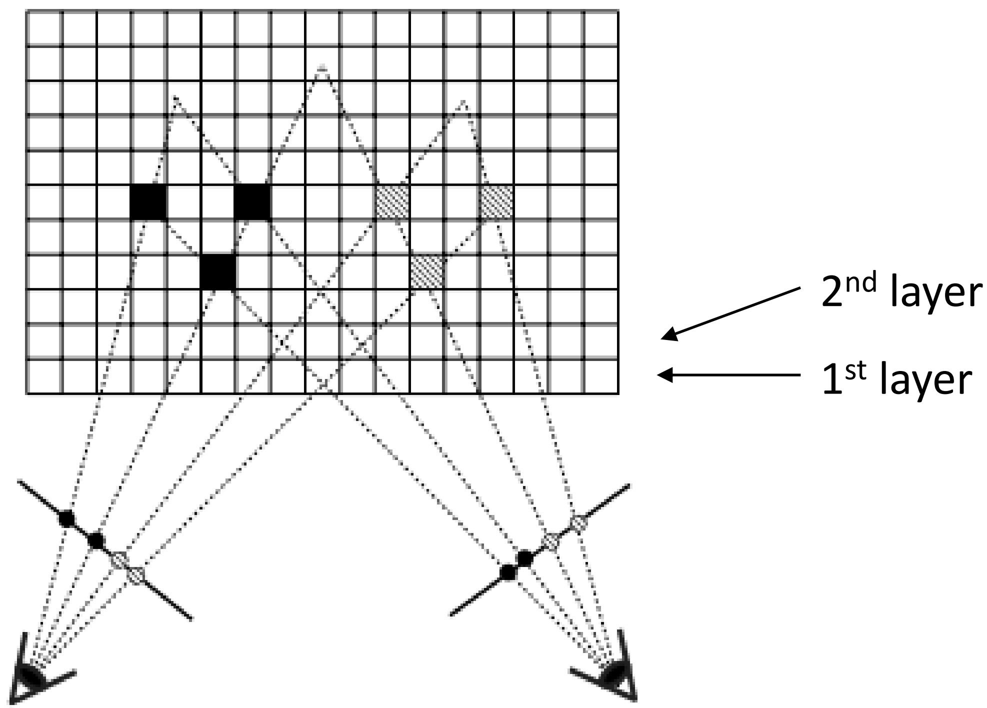
If no scene points lie within the convex hull of the camera positions, then D is the shortest distance from the hull to a voxel



The Algorithm

Initialization: all pixels ‘unmarked’, all voxels empty

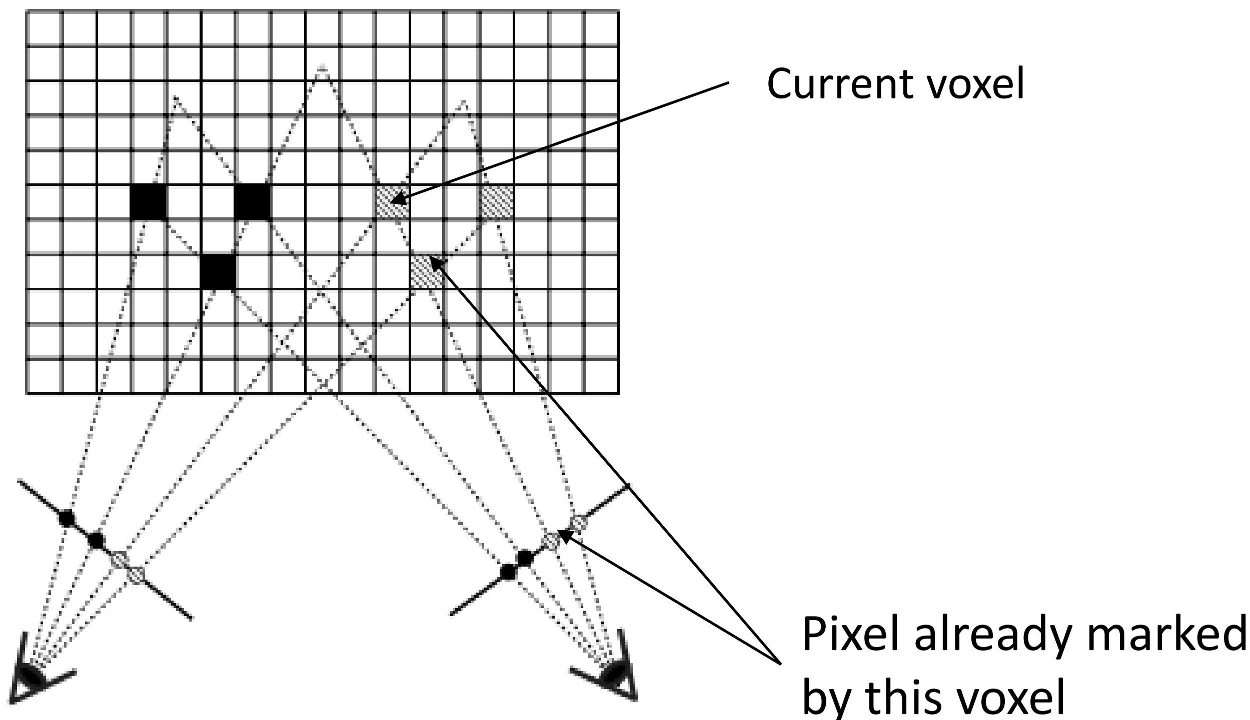
Step through the voxel space in ‘layers’ with equal D values, starting from the smallest.



The Algorithm

For each voxel, find the **unmarked** pixels it occupies in every image.

Marked pixels must be occluded by voxels already traversed, because we are traversing in order of increasing D



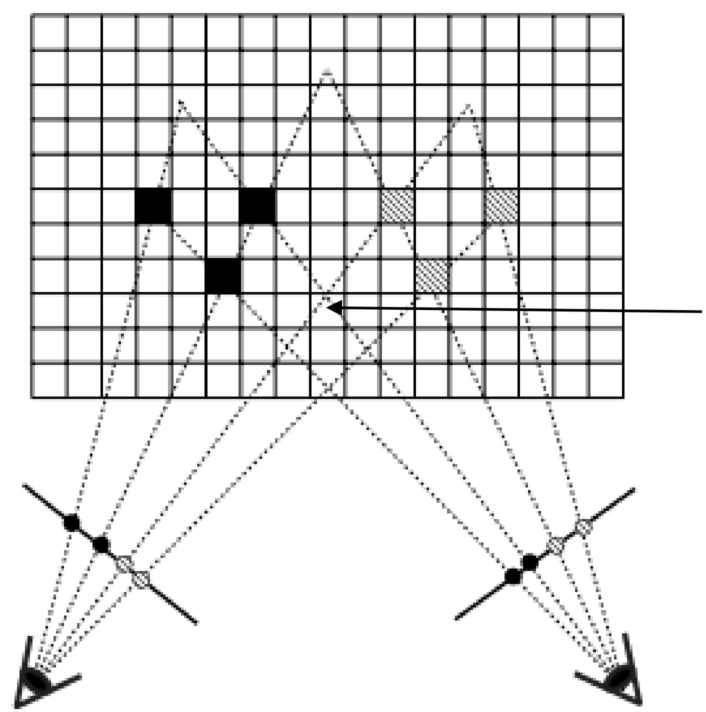
Coloring the Voxel

We compute the color consistency

$$\lambda = \frac{s^2}{s_0^2},$$

where s is std of visible pixels colors, and s_0 is the typical std of matching pixel colors.

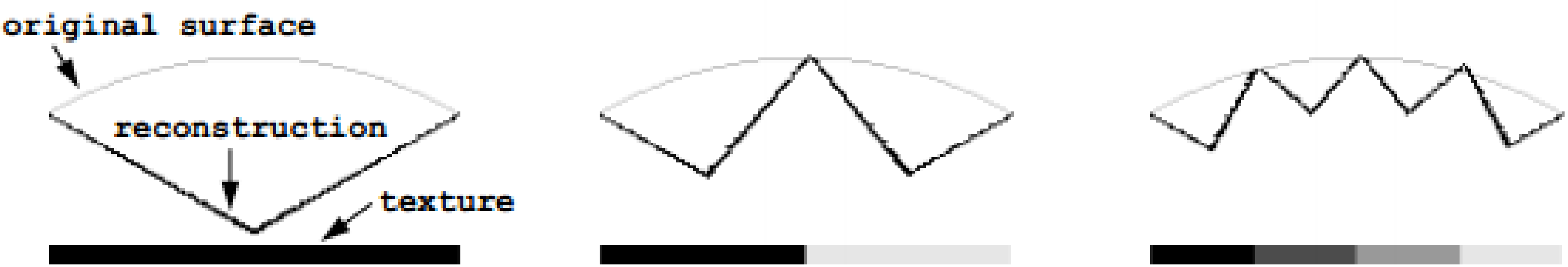
The voxel is colored if it is visible in at least 2 images and λ is less than some threshold



Failure cases

Voxel coloring is greedy – it colors the nearest voxel that matches the observation.

Need lots of texture variation for accurate results



(b) (c) (d)

True Photorealistic Reconstruction

Qian-Yi Zhou and Vladlen Koltun 2014

Build 3D reconstruction using Kinect Fusion

Refine camera poses to make texture as sharp as possible

Questions about PS3