Recovering Geometric Structure

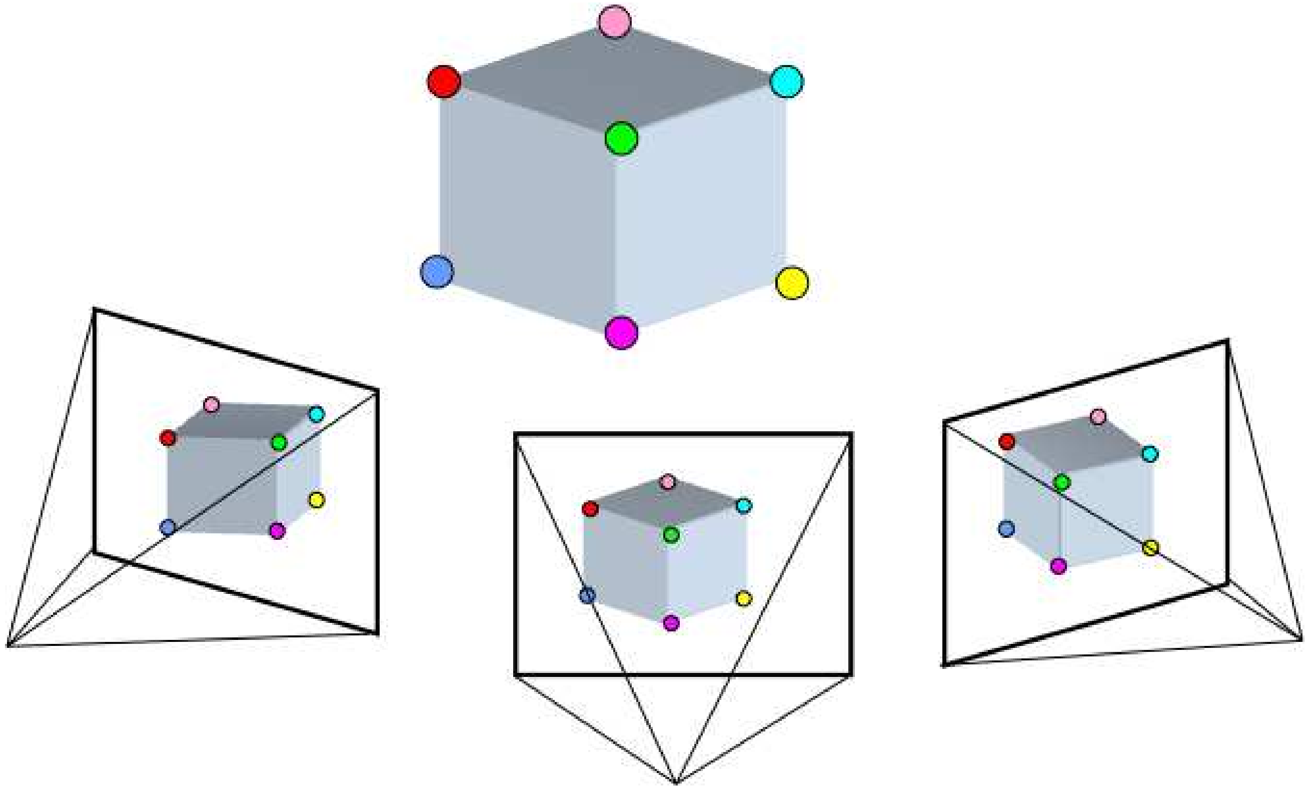
P J Narayanan

CS5765. Computer Vision. Spring 2013

CVIT, IIIT, Hyderabad

Multiple Views of Points/Objects

- Given projections of a set of 3D points in two or more cameras, get their 3D coordinates
- **Each 3D point is identified in every camera view**
- What else is known? Camera matrices K_i, R_i, t_i? Only the intrinsic parameters K_i?



Different Problems

Variations of the problem:

- **(Binocular) Stereo:** Two cameras with known intrinsic and extrinsic parameters.
- **Multiview Stereo:** Multiple known cameras
- **Structure-from-Motion:** Given m cameras and n points and projections x_{ij} of point j in camera i, recover 3D points X_j and camera matrices C_i
  - **Affine SfM:** For affine cameras
  - **Projective SfM:** For general projective cameras
- **Bundle Adjustment:** Directly recover C_i, X_j from x_{ij}

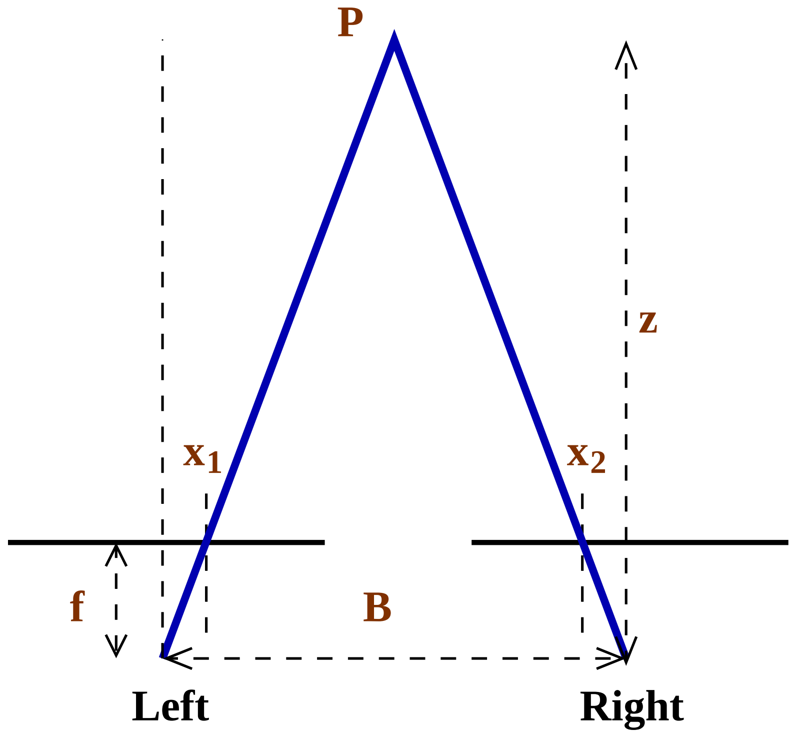
Binocular Stereo and Feature Correspondence

Geometry of Matching

- B: baseline, f: focal length, z: depth, x: image coords
- From similar triangles:

$$\frac{x_1 - x_2}{f} = \frac{B}{z}$$

- **Stereo Disparity or Parallax**: the "shift" between the left and right images. $\Delta = \frac{Bf}{z}$.

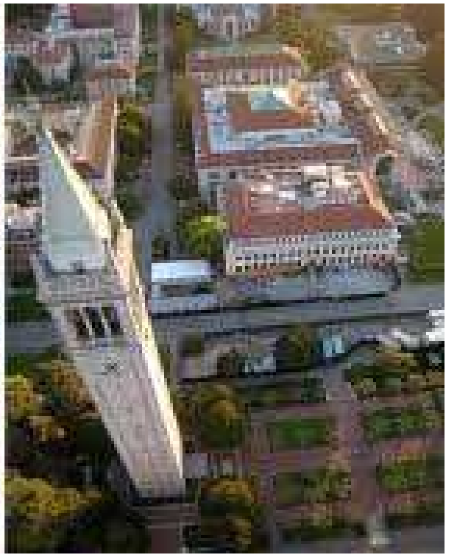


- Farther the point, smaller the disparity and vice versa
- A large baseline can give more reliable estimates of depth. But, matching may become harder
- Basic step: **Identify common points in camera views**

Identifying Common Points

- Find a world point in 2 or more views
- Appearance is the only clue to identify them
- Individual pixel colours are similar very often. Match is too noisy
- Match a (small) neighbourhood of colours from one image to a similar neighbourhood in others
- Will work if local surface is fronto-parallel and images have similar magnification
- Foreshortening can happen when viewing an oblique surface
- Many ambiguities. We need a lot of help!

Some Examples



Matching Patches

- Compare $m \times m$ patches from two views.
  Form vectors $\mathbf{v}$ and $\mathbf{v}'$ of length m^2 from them
- Matching scores between patches:
  - Sum of Absolute Difference (SAD): $||\mathbf{v} - \mathbf{v}'||_\mathbf{1}$
  - Sum of squared difference (SSD): $||\mathbf{v} - \mathbf{v}'||_\mathbf{2}$
  - Correlation: $\frac{\mathbf{v}'^\mathbf{T}\mathbf{v}}{\sqrt{\mathbf{v}^\mathbf{T}\mathbf{v}}\sqrt{\mathbf{v}'^\mathbf{T}\mathbf{v}'}}$
  - Normalized correlation: $\frac{\bar{\mathbf{v}}^\mathbf{T}\bar{\mathbf{v}}'}{\sqrt{\bar{\mathbf{v}}^\mathbf{T}\bar{\mathbf{v}}}\sqrt{\bar{\mathbf{v}}'^\mathbf{T}\bar{\mathbf{v}}'}}$. Range: $[-1, 1]$

    $\bar{\mathbf{v}}, \bar{\mathbf{v}}'$ are vectors with respective patch-mean colour subtracted.
  - Invariant to affine changes in intensity/colour.

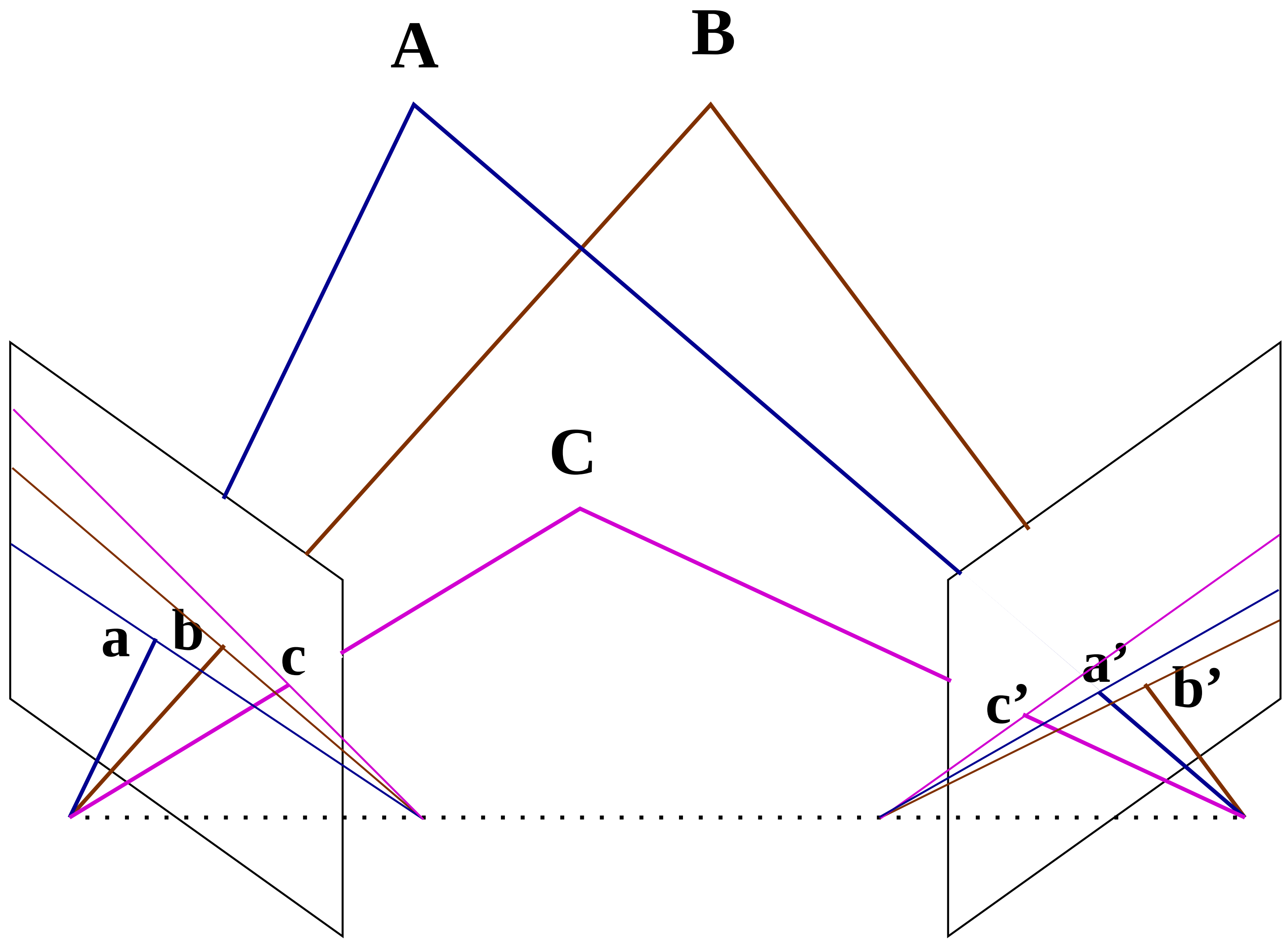
Constraints on Matching

- **Epipolar:** Match lies on the epipolar line of the pixel
- **Colour Constancy:** The appearance/colour does not change from one view to another
- **Uniqueness:** A point on left image can match with only one point on the right and vice versa
- **Ordering or Monotonicity:** If point A is to the left of B in view 1, it will to the left of B in view 2 also. (Violated if great difference in depth exists)
- **Continuity:** Disparity values vary smoothly (violated at occlusion boundares)

Sparse correspondence: only for good feature points

Dense correspondence: a match for every pixel

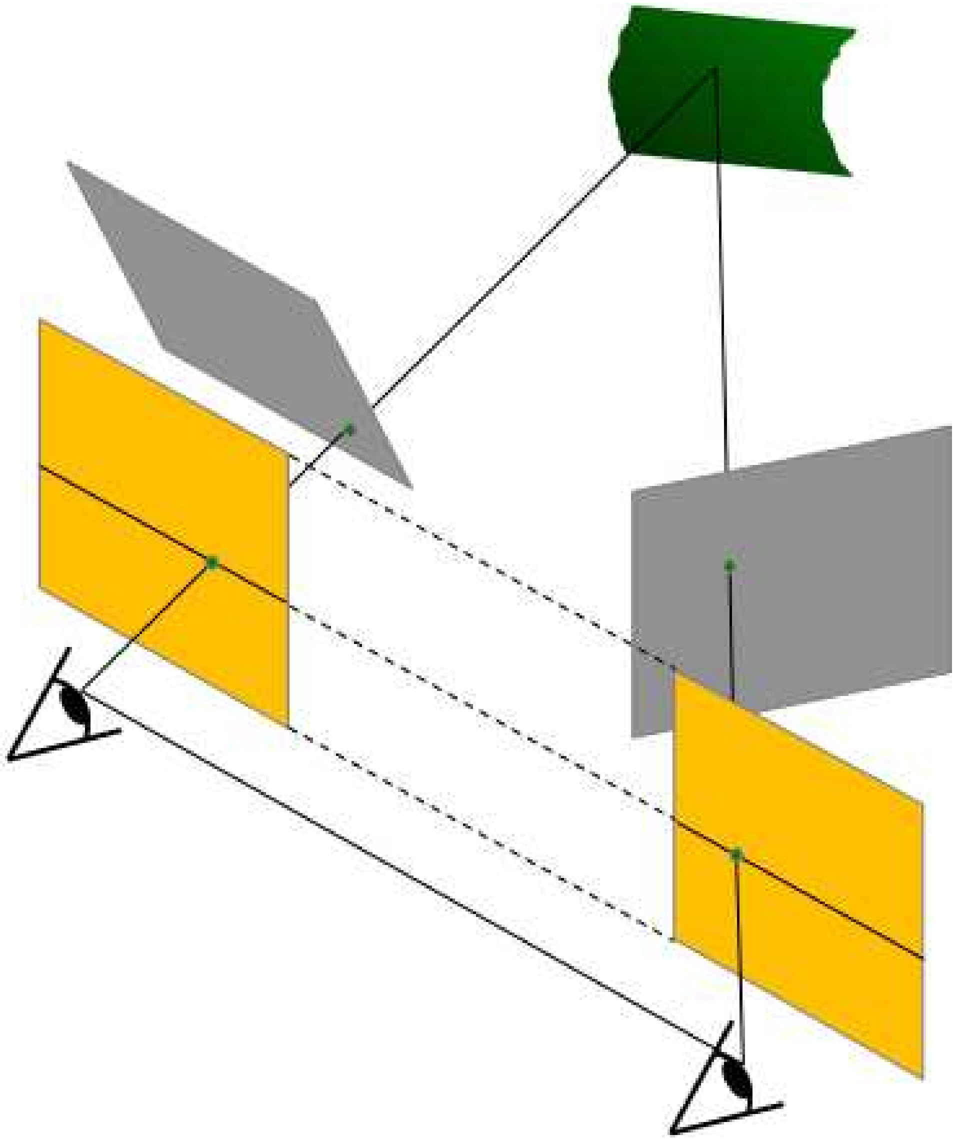
Epipolar Geometry



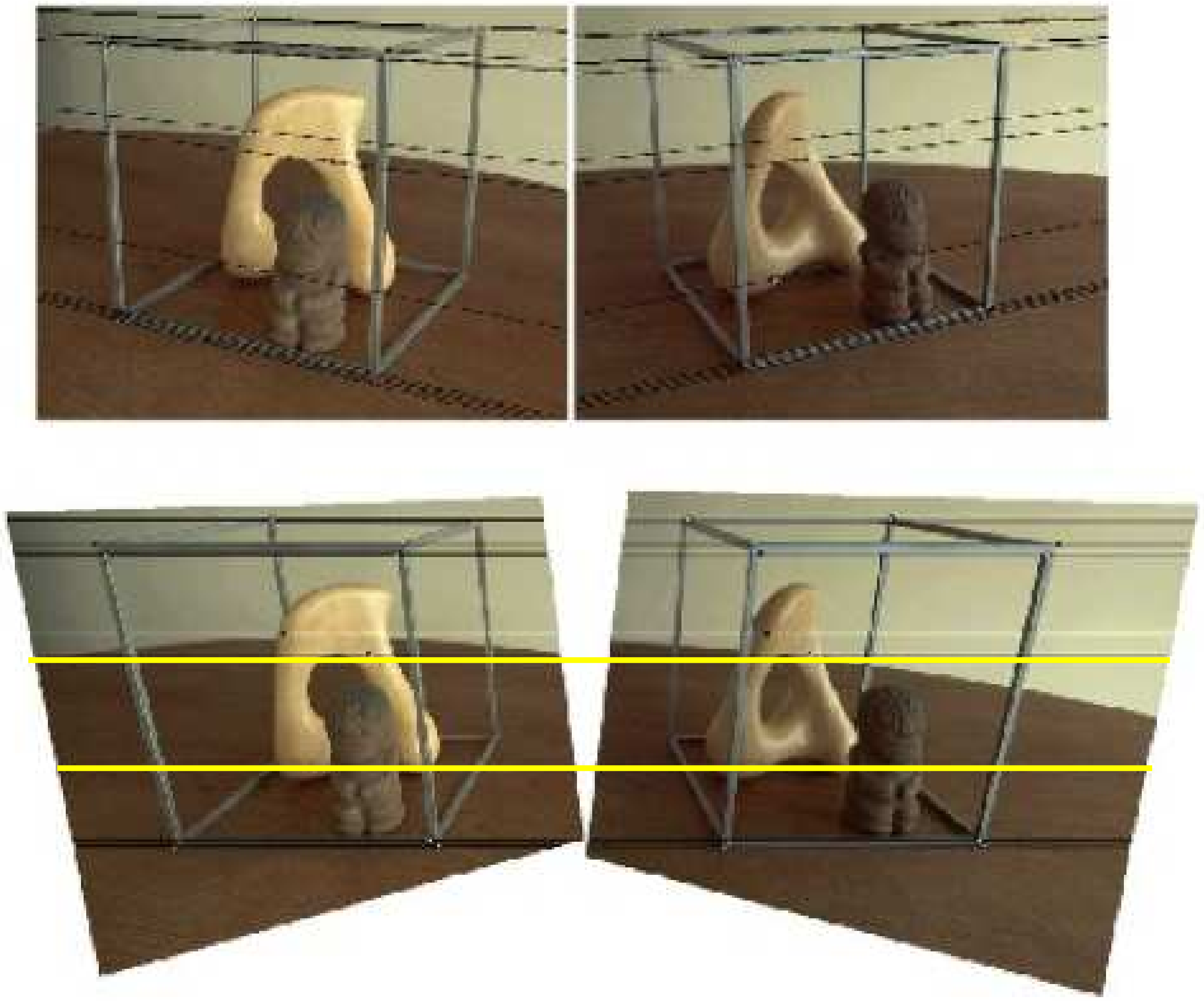
Reduced Search and Rectification

- The search is limited to a line if fundamental matrix known (i.e., **weakly calibrated**)
- Simplest if left and right cameras have same image plane and pure $\mathbf{X}$-translation between them.
- Fundamental matrix has a simple form.
  Epipolar constraint reduces to $y' = y$.
- Matches constrained to lie in the same scan line
- **Rectification:** A rotation of the camera (to make image planes parallel) and a change in $\mathbf{K}$ matrix (focal length, image center).
- Can be represented using a homography $\mathbf{H}$ to align one image plane to the other
  Or, homographies $\mathbf{H_1}, \mathbf{H_2}$ to align them to a third plane

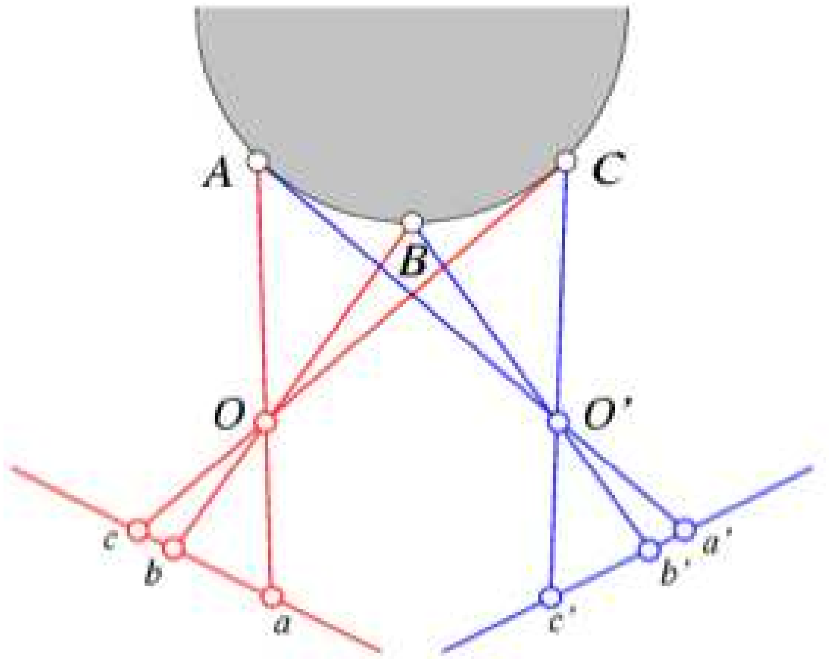
Rectification

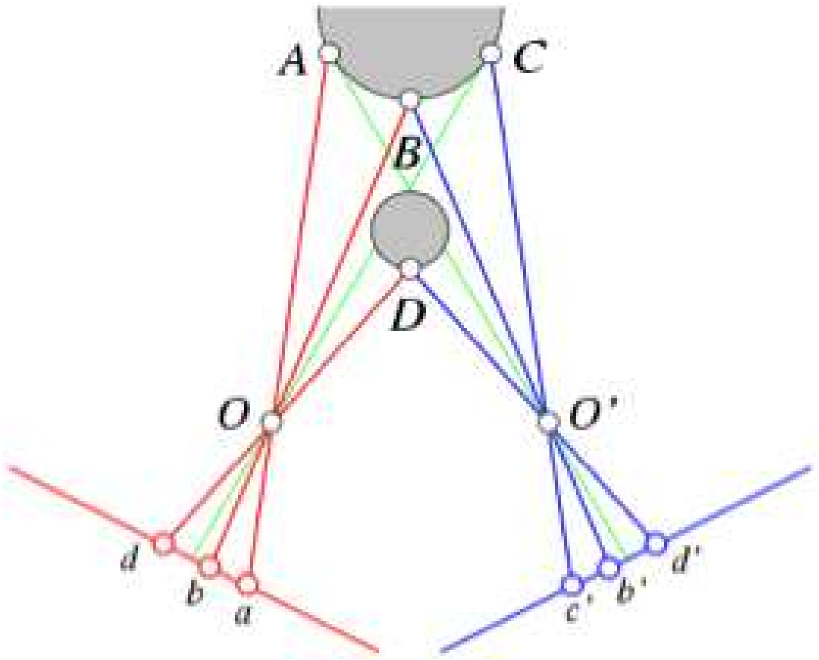


Rectification: Example



Ordering Constraint





Order of matches from left and right is ordinarily preserved
Violation may mean something else.

Various Situations

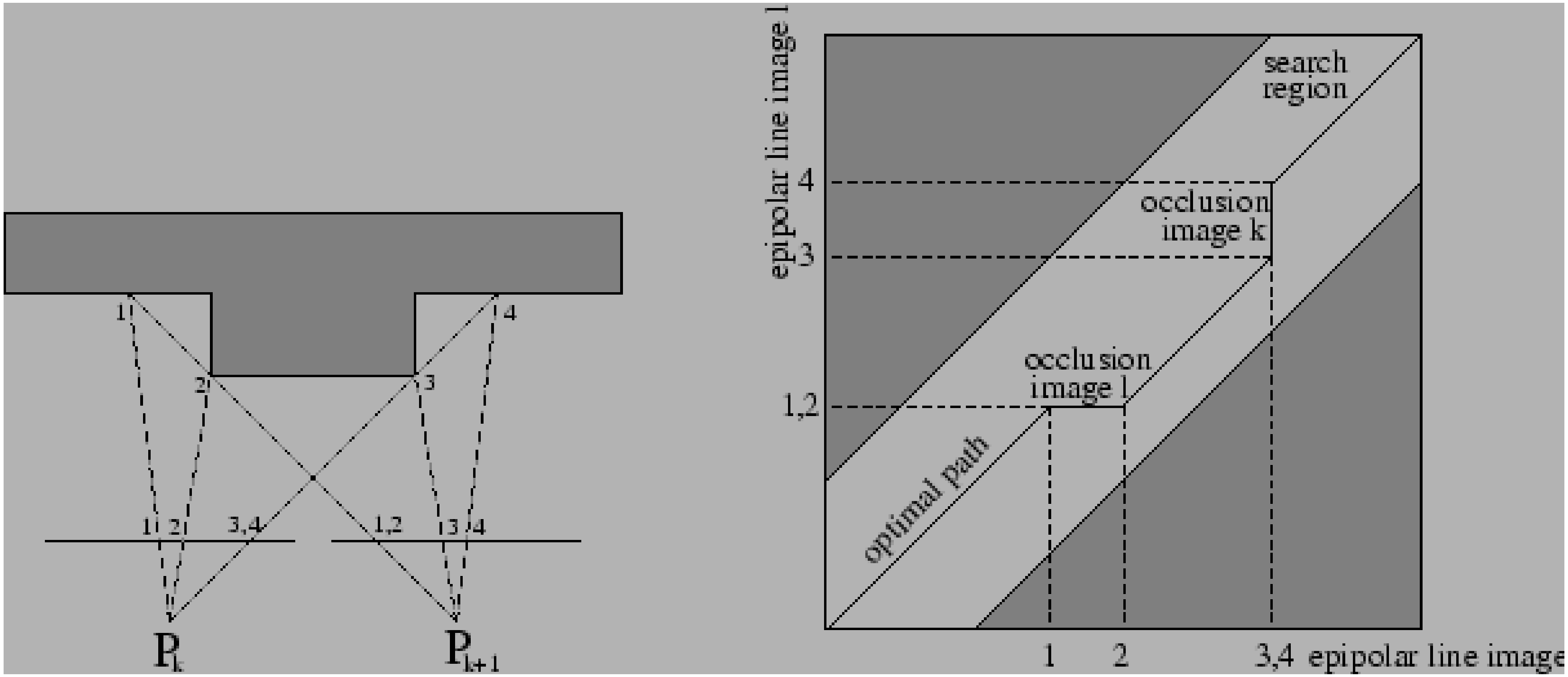


Image courtesy Marc Pollefeys

Shows the *epipolar line image* or *disparity space image* with different scenarios

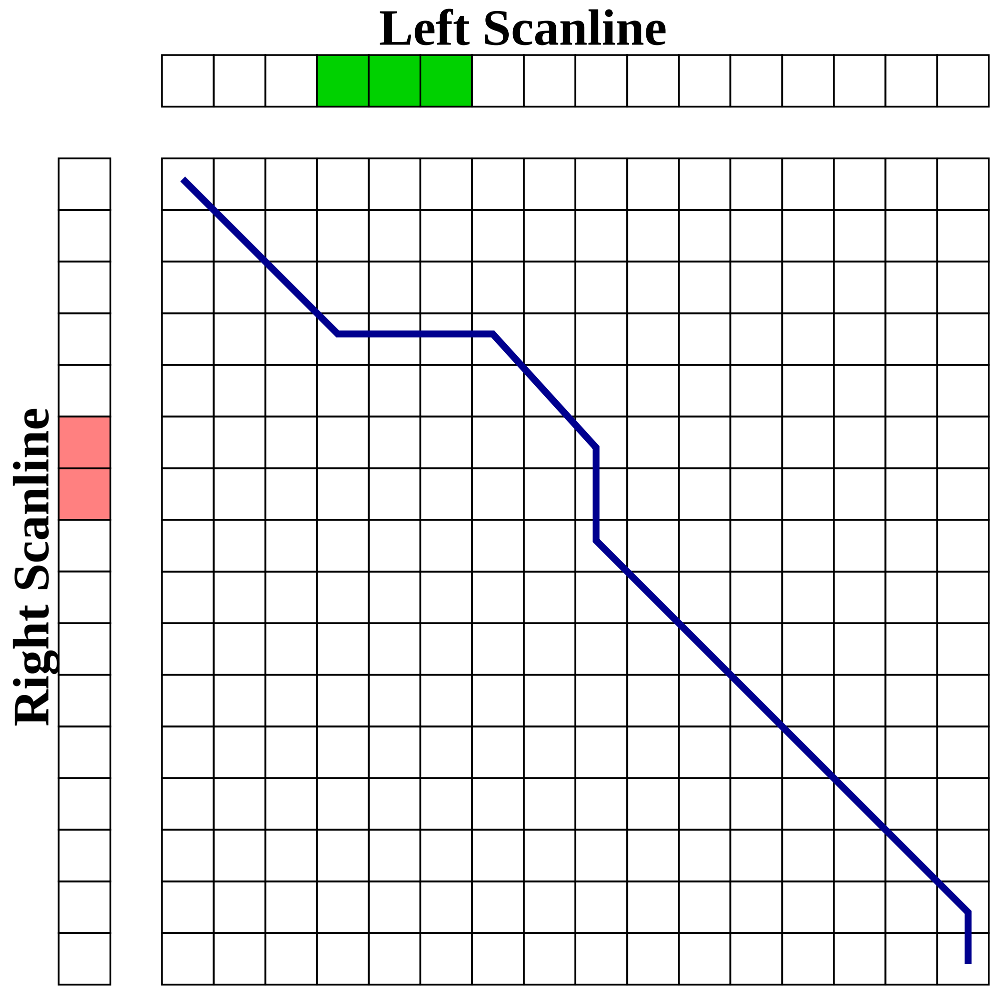
Scan-Line to Scan-Line Matching

- Disparity Line Image pits pixels of one row of the left image against the pixels of the matching epipolar line in the other.
- Several matching scenarios:
  - If left pixel $(i-1)$ matches with right pixel $(j-1)$, next pixel i can match pixel j, if the match is good
  - Otherwise, it may continue the match with $(j-1)$ with an occlusion cost (due to left occlusion)
  - Or, $(i-1)$ can match with j with another occlusion cost (due to right occlusion)
- Sub-pixel definitions may be needed when zoom is different

Dynamic Programming Solution

- Cost of matching: $C(i-1, j-1) + c(i, j)$ if pixels match, $C(i-1, j) + C_o$ if left occlusion, and $C(i, j-1) + C_o$ if right occlusion, where C_o is a high occlusion cost
- Select the minimum from those three and declare match or occlusion accordingly
- Can be setup nicely as a dynamic programming solution working in the i, j space, starting with leftmost pixel match
- Cost of matching: $O(N^2)$ where N is the number of pixels in each scanline.

Dynamic Programming Path



- Initialize first row and col to $i * C_o$
- Do for $i \in [0, N-1]$ and $j \in [0, N-1]$:

  Set $C(i,j)$ to min of $C(i-1, j-1) + c(i,j), C(i-1, j) + C_o, C(i, j-1) + C_o$
- Mark each as **M/L/R**
- Reconstruct from (N, N), by folllowing the **M** pixels and their connections.

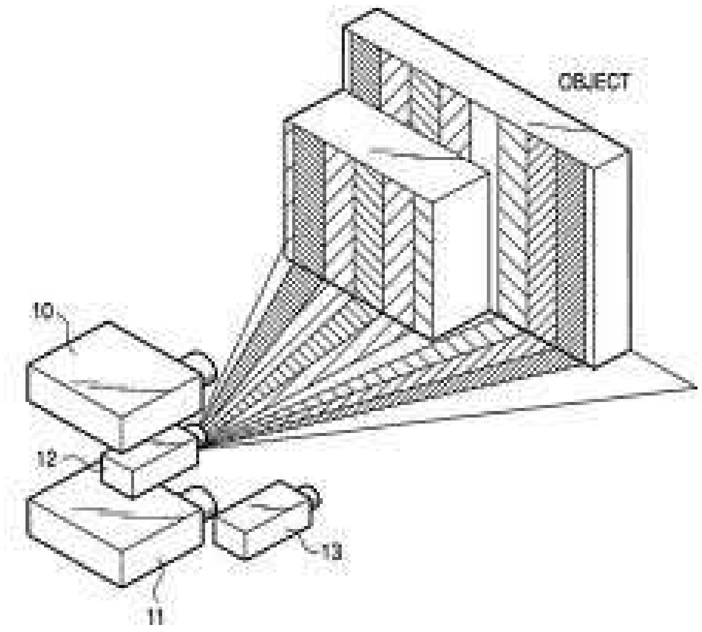
Globally Optimal Solution

- Provides a *globally optimal* match as opposed to the local matching done by search
- Provides dense correspondence: a match for every pixel
- Works well enough. And is a prototype for many global stereo matching approaches that followed
- Difficulty: assigning a cost for occlusions.
- Difficulty: maintaining consistency across scan lines

We will see another global solution using graphcuts later

Structured Lighting

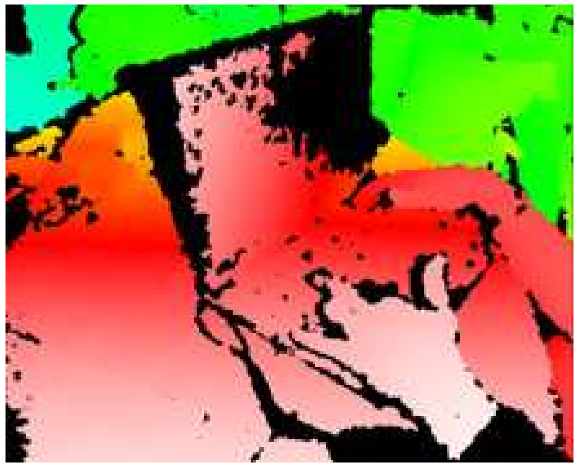
- Finding correspondences is hard by itself
- Can we help it by projecting patterns onto the world? **Structured Lights!**
- Lightstrip range finders, etc.
- Combination of sinusoids sometimes to get dense matches
- *Active vision*, as it changes the appearance
- The light projected need not be in the visible spectrum



Xbox Kinect

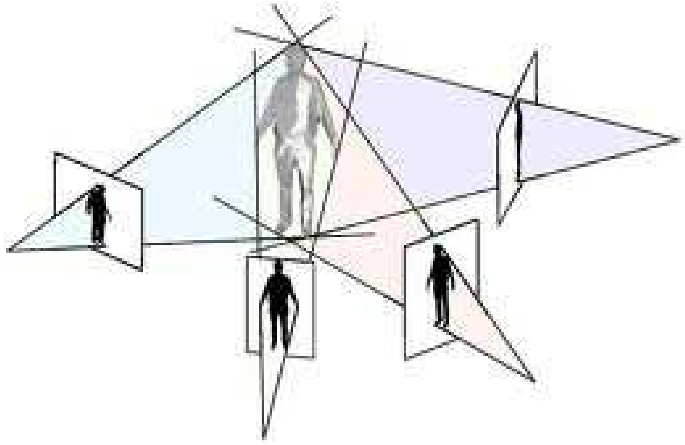
IR-based range sensor for Xbox

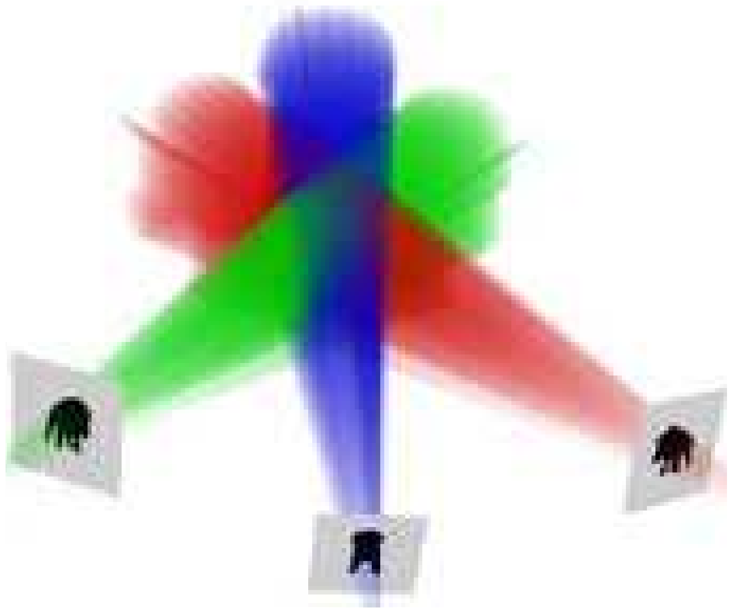
- Aligned depth and RGB images at 640×480
- Original goal: Interact with games in full 3D
- Computer vision happy with real-time depth and image
  - Games, HCI, etc
  - Action recognition
  - Image based modelling of dynamic scenes
- Fastest selling electronic appliance ever!!
- Other products that use PrimeSense sensor



Visual Hull

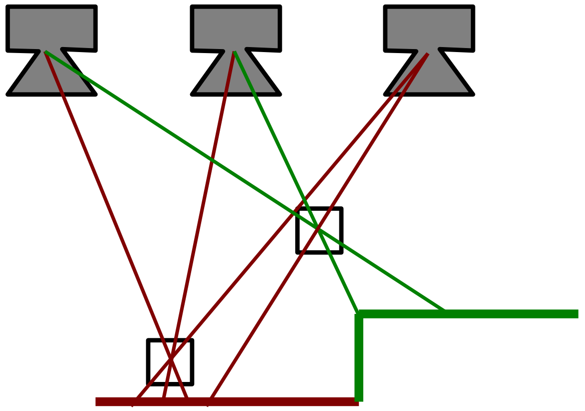
- Object silhouette represents a generalized cone with the camera centre as the apex
- Intersect these cones for multiple views in the 3D space
- **Visual Hull**, like convex hull
- Cannot get fine details like concavities
- Gives a very good, approximate shape, without scene modification!

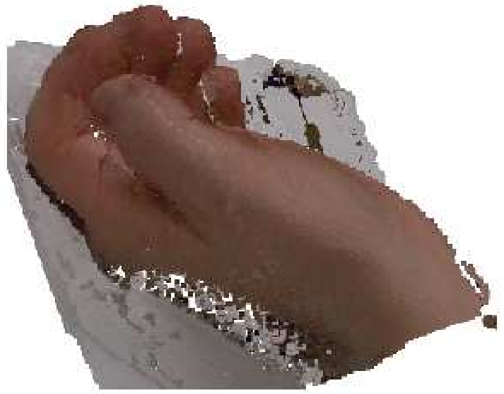




Space Carving

- Reason directly in a volumetric **voxel** space
- If a voxel is filled, it projects to similar colours in all cameras
- If a voxel is empty, its projections will have different appearances
- *Colour consistency:* filled or empty?
- Assume all filled initially; carve out empty ones by going over the images, guessing visibility, etc.





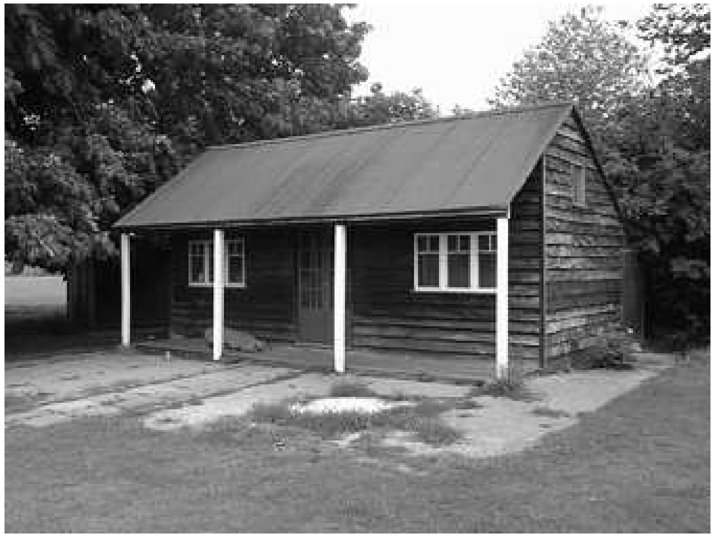
Stratified 3D Reconstruction

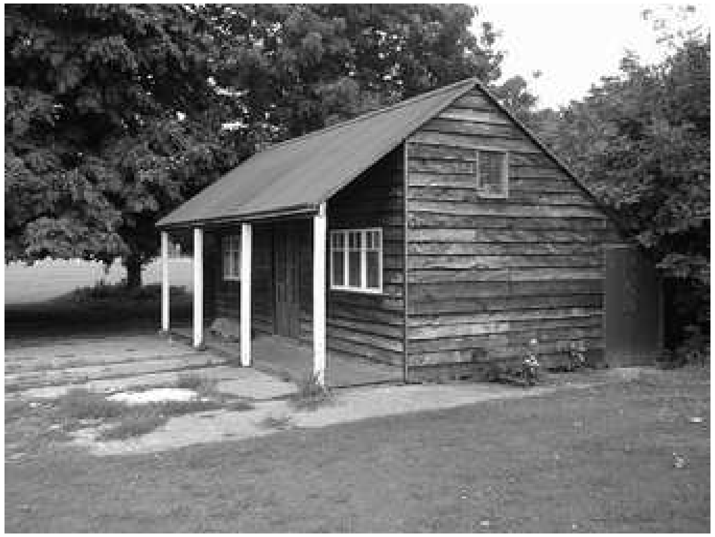
Structure from Two Views

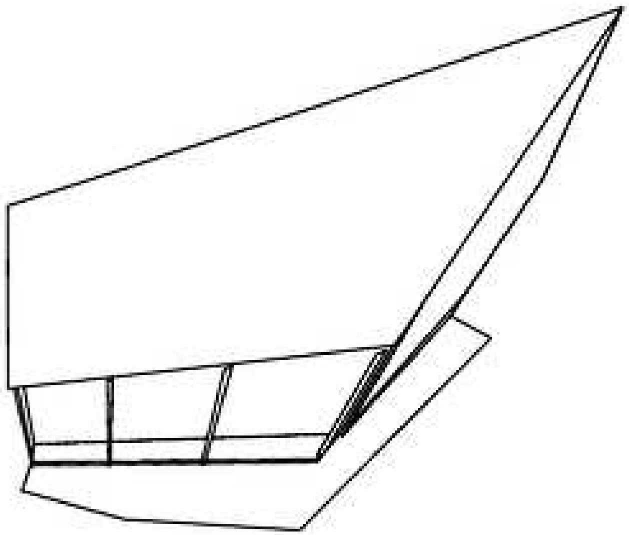
- Estimate fundamental matrix from point matches
- Decompose $\mathbf{F}$ into canonical cameras: $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$ and $\mathbf{P}' = [\mathbf{M}|\mathbf{m}]$. Epipole $\mathbf{e}'$ is the left null-space of $\mathbf{F}$. Second camera centre is: $\mathbf{C}' = –\mathbf{M}^{-1}\mathbf{e}'$
- Rays for matching $(\mathbf{x}, \mathbf{x}')$ are: $[0\ 0\ 0\ 1]^{\mathbf{T}} + \lambda[\mathbf{x}^{\mathbf{T}}\ \ \mathbf{0}]^{\mathbf{T}}$ and $\mathbf{C}' + \gamma[(\mathbf{P}'^{+}\mathbf{x}')^{\mathbf{T}}\ \ 0]^{\mathbf{T}}$. They meet at a 3D point $\mathbf{X}$ as they lie on a plane
- $\mathbf{X}$ can be found by equating the two equations to solve for λ, γ (3 equations in 2 unknowns).

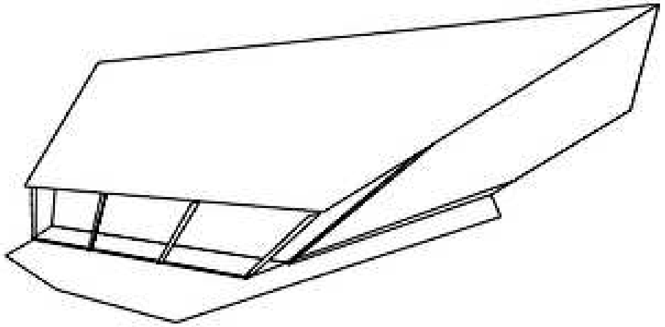
The structure recovered is *projective*. Points $\mathbf{X}$ are within an unknown, general, 4×4 projective transformation of the true Euclidean structure

Projective Structure of House



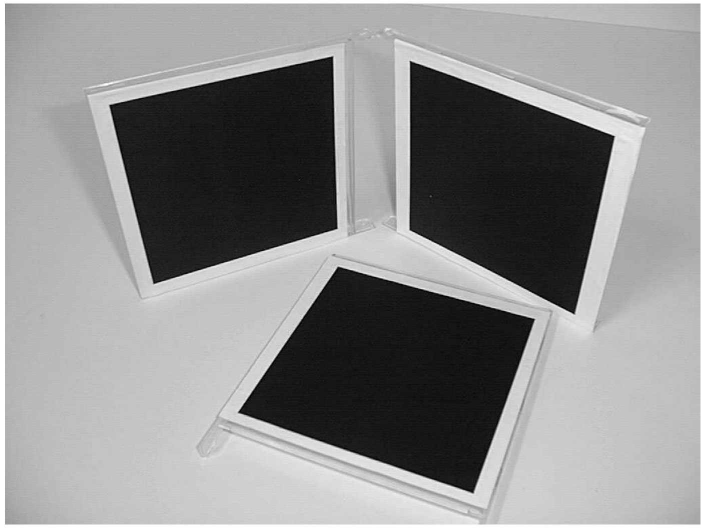






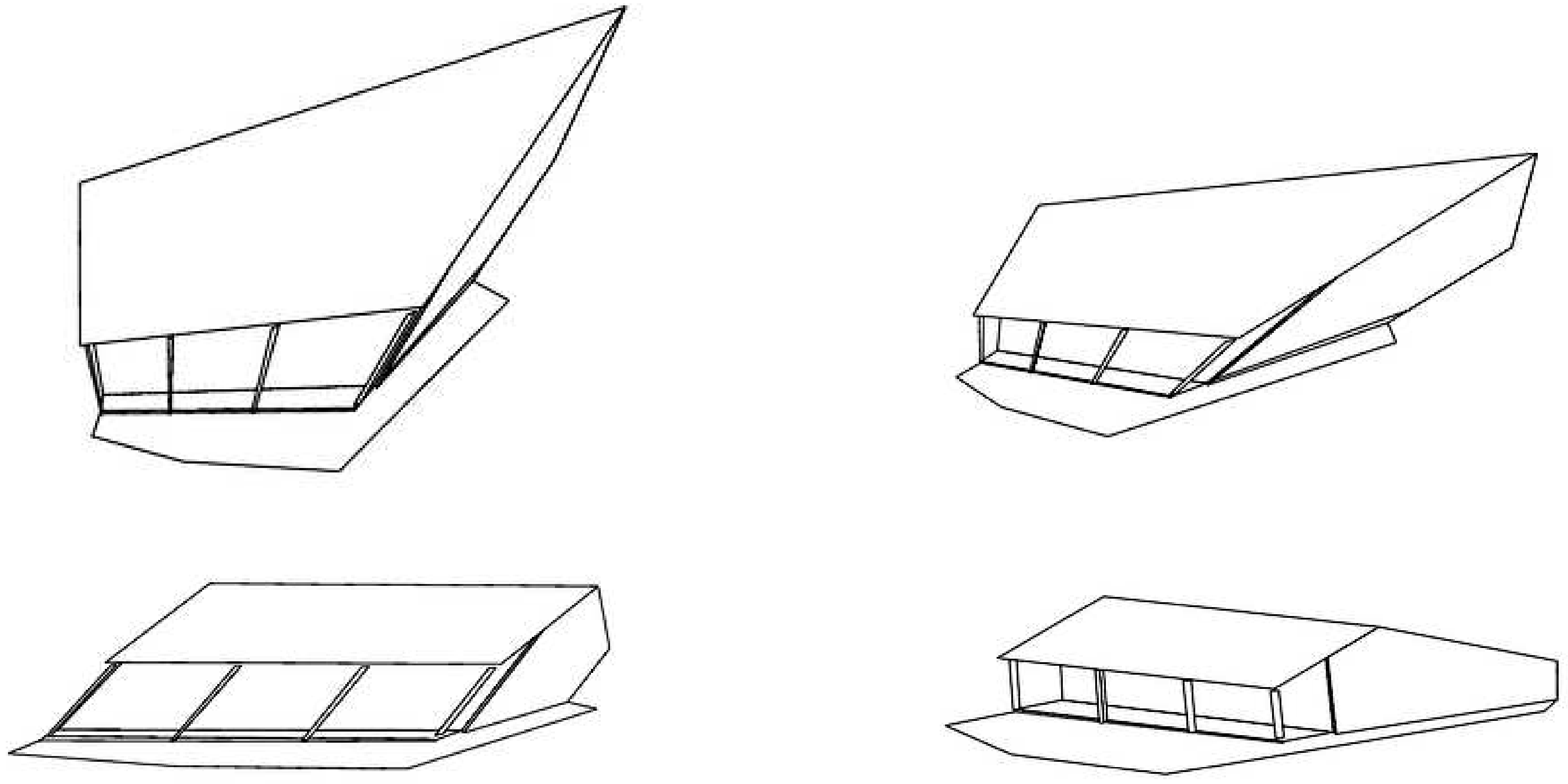
Projective to Affine Structure

- The ambiguity is affine if the plane at infinity or $\mathbf{H}_\infty$ is known.
- Plane at infinity can be recovered in the image using 3 parallel line pairs in different directions. Each gives a point on the plane at infinity.



- π_∞ can be obtained in many ways: from distance ratios, perpendicular lines, etc.
- If $\mathbf{H}_\infty$ is known, $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$ and $\mathbf{P}' = [\mathbf{H}_\infty|\mathbf{e}']$ are affine cameras. Triangulation using these will give structure upto an affine ambiguity

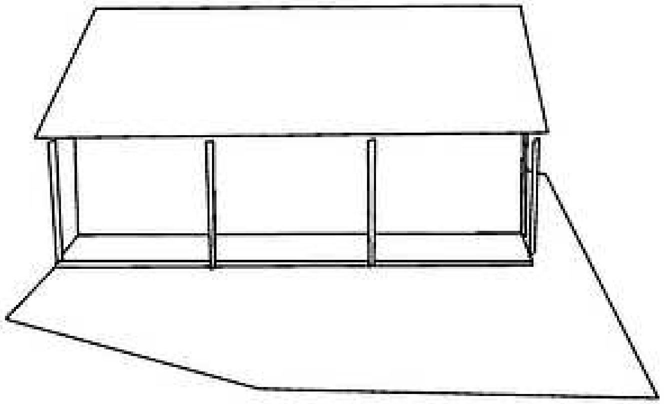
To Affine Structure of House

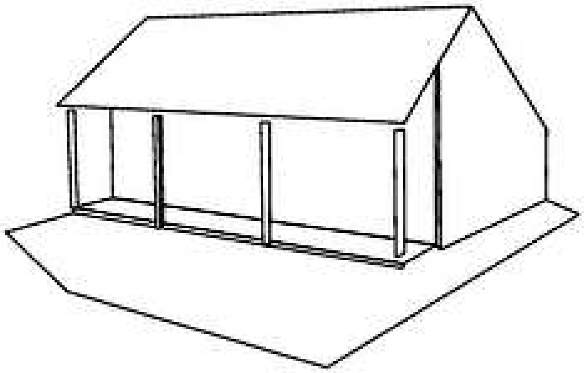


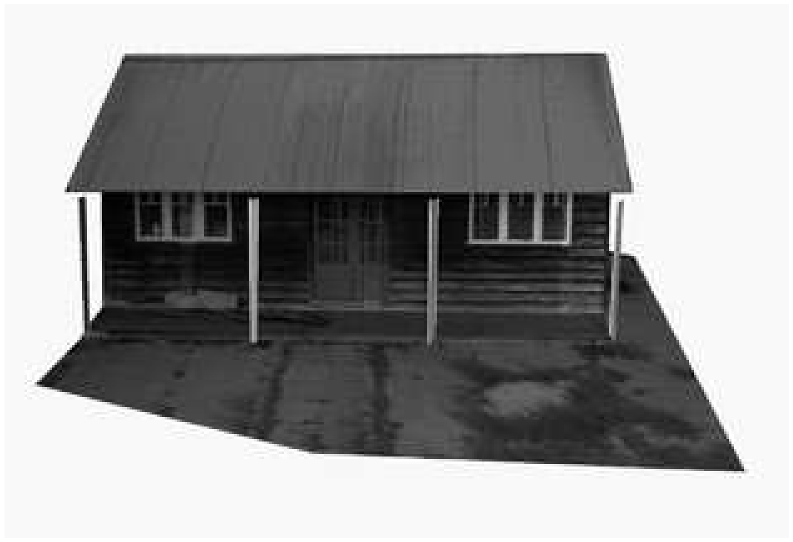
Affine to Metric Structure

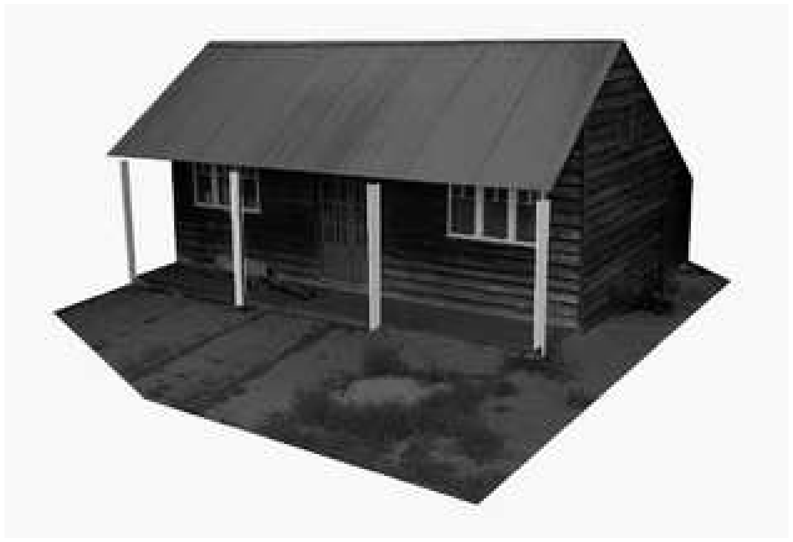
- Metric ambiguity if image of the absolute conic ω is known.
- Map ω to $\mathbf{I}$ on π_∞ using $\mathbf{H}$. The same $\mathbf{H}$ will transform affine structure to metric structure.
- IAC can be recovered from parallel lines, perpendicular lines/planes in the image, or a combination of those.
- IAC needs to be known only in one view. It can be transferred to the other using $\mathbf{H}_\infty$
- From IAC, get $\mathbf{K}$ using $\omega^{-1} = \mathbf{K}\mathbf{K}^\mathbf{T}$. Essential matrix $\mathbf{E} = \mathbf{K}'^\mathbf{T}\mathbf{F}\mathbf{K}$. Decomposed it into metric cameras with 4-way ambiguity. Resolve by identifying a point in front of the camera. Triangulation will now result in metric structure of the scene.

To Metric Structure of House







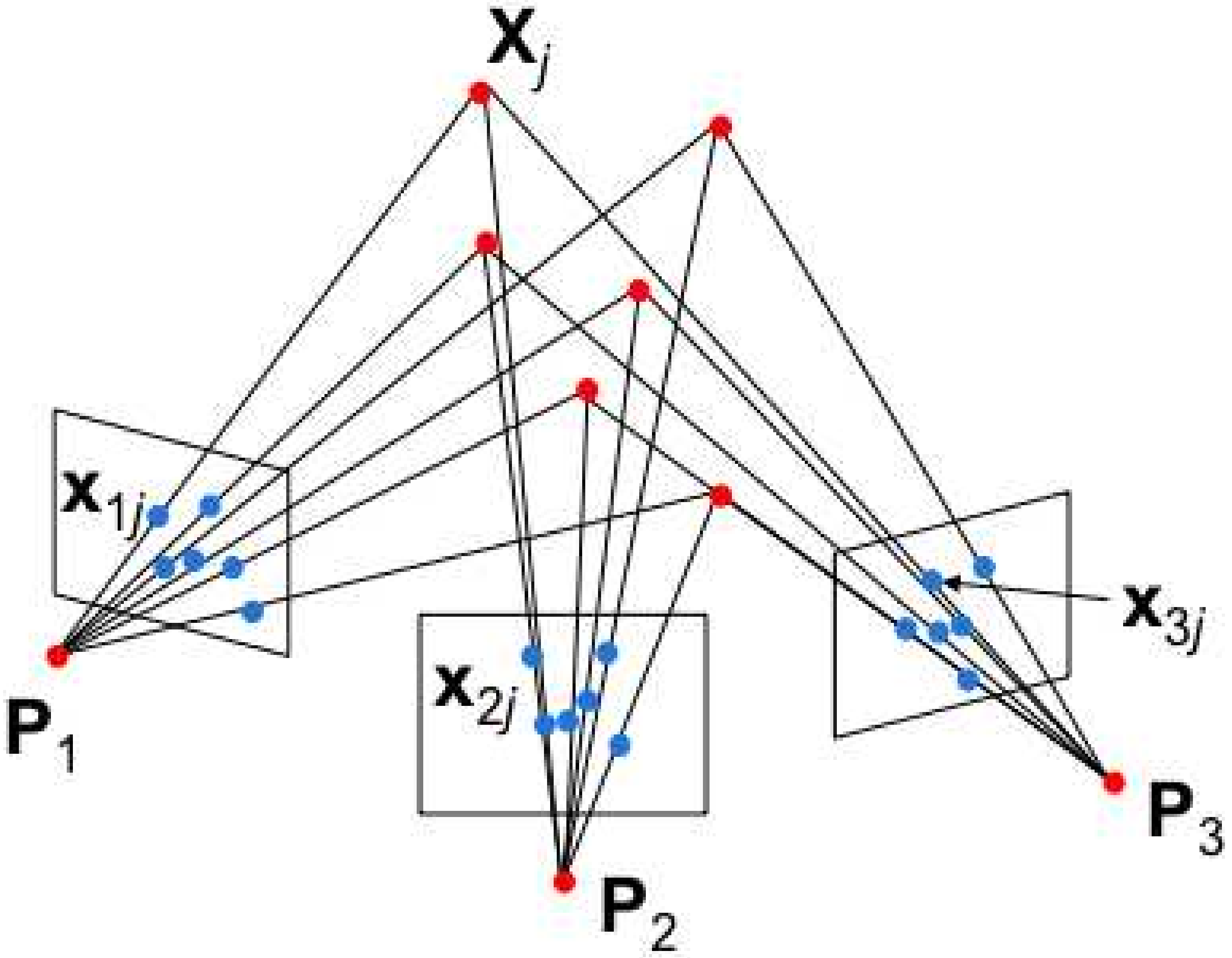


Direct Metric Structure

- Projective structure can be converted to metric using ground truth points whose metric structure is known.
- Assume recovered projective $\mathbf{X_i}$ has metric coordinates $X_{\mathbf{E}i}$. A homography exists such that: $X_{\mathbf{E}i} = \mathbf{H}\mathbf{X}_i$
- The above gives 3 equations. With 5 or more points, the homography $\mathbf{H}$ can be recovered.
- Apply the $\mathbf{H}$ to all recovered points (and camera) to get metric structure.

Structure from Motion Using Bundle Adjustment

Points in Multiple Views



Structure From Motion

- m cameras and n points, with correspondences
- Unknown: m matrices P_i and n coordinates, X_j
- We have: $x_{ij} = P_i X_j, \quad 1 \leq i \leq m, \; 1 \leq j \leq n$
- $2mn$ equations in total (2 for each match)
- Can be solved if $2mn > 11m + 3n$
- However, under projective, $\mathbf{PX} = (\mathbf{PQ})(\mathbf{Q}^{-1}\mathbf{X})$.
  A projective ambiguity will remain
- Projective structure if $2mn > 11m + 3n - 15$
- Affine structure if $2mn > 11m + 3n - 12$
- Metric structure if $2mn > 11m + 3n - 7$
- Affine/Metric structure only by enforcing affine/metric constraints

Bundle Adjustment

Given m views of n 3D points, with unknown $\mathbf{P_i}$ and $\mathbf{X_j}$. Ideally, $x_{ij} = P^i X_j$

Minimize the re-projection error over all camearas/views:

$$\min_{P_i, X_j} \sum_{i=1}^{m} \sum_{j=1}^{n} dist(x_{ij}, P_i X_j)^2$$

A non-linear optimization problem. Can be solved using the Levenberg-Marquardt procedure directly.

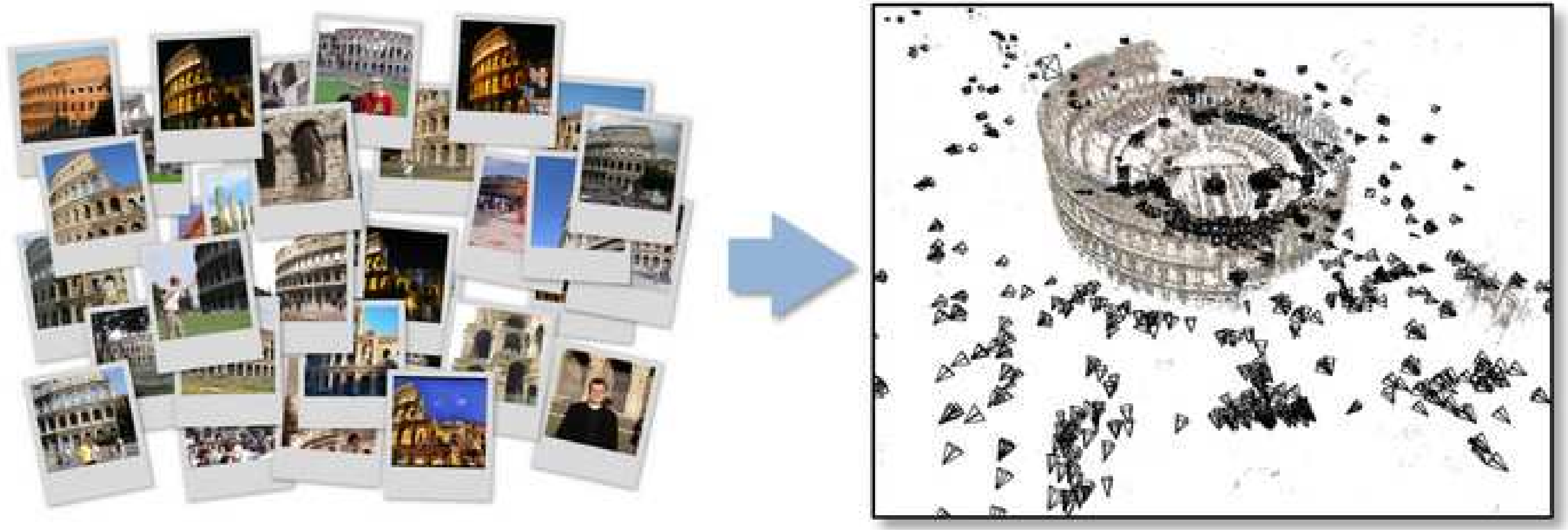
Called **bundle adjustment**. Known to photogrammetry community for a long time

Needs good initialization as the complex non-linear optimization problem can get stuck in local minima

SfM from Community Photo-Collections

PhotoTourism or PhotoSynth

- An automatic process, starting with independent images of a scene/monument/object. The images could be from a video sequence.
- Of particular interest has been SfM from Community Photo Collections (CPC), which are images that can be downloaded from flickr/picasa by giving a keyword like “Taj Mahal”.



SfM: Steps

1. Download images for the place of interest!!
2. Extract descriptors from interest points on all images
3. Match points in pairs of images using Approximate Nearest Neighbours
4. Refine matches using Geometric Verification: Epipolar relationship, etc.
5. Form tracks of points across images. Transitively connect matches to get long matching “tracks”
6. Build image connectivity graph based on common points
7. Perform incremental SfM using the image connectivity graph and bundle adjustment

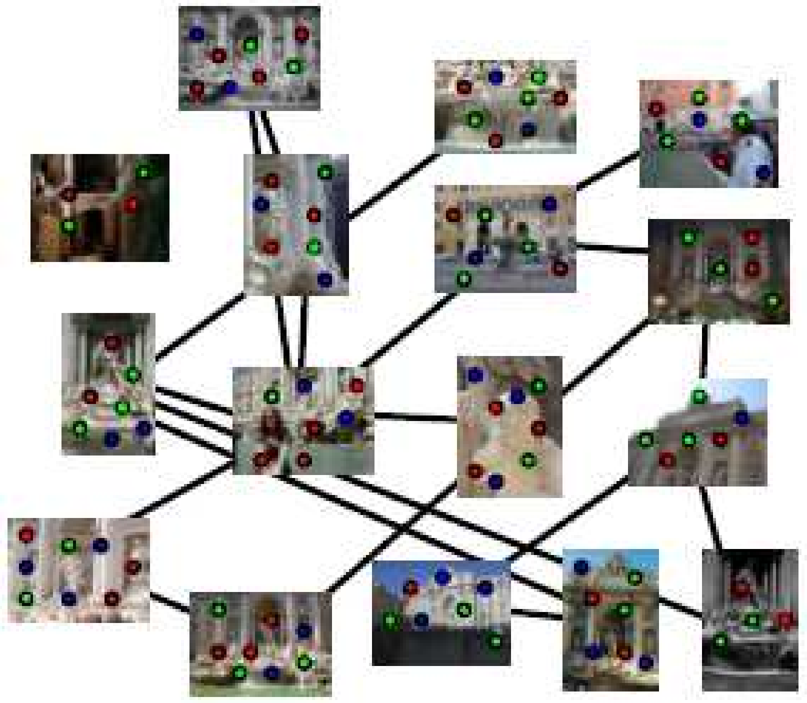
Matching Points Across Images

- Extract interest points $\mathbf{p}_{ij}$ in each image I_i and descriptors s_{ij} for it. SIFT is popular. A few thousand in a typical image
- Match interest points in image pairs. An approximate nearest neighbour approach is used, with a ratio test
- Point $\mathbf{p}_{ij}$ matches point $\mathbf{p}_{kl}$ iff:
  (a) dist(s_{ij}, s_{kl}) is minimum over all points in I_k and
  (b) dist$(s_{ij}, s_{kl}) < r \times$ dist(s_{ij}, s_{km}) where p_{km} is the second closest point in I_k. r is typically 0.6
- Discard all points involved in case of multiple matches

Geometric Verification and Tracks

- Find fundamental matrix between pairs of cameras using RANSAC
- Refine matches by eliminating those not satisfying epipolar relation
- Propagate matches using transitivity to generate **tracks** which represent the same world point in multiple images
- Form image connectivity graph. Two images have an edge if they share a point
- Densely connected regions represent a parts of the scene visible to a large number of views. Sparse, leaf regions denote low sampling of parts of the scene

Features and Match Graph



Form **tracks** of points by transitively connecting matches.
They represent the same 3D point in multiple images.

Track Statistics

For a typical large data set with approx 3000 images:

- 1.5 million tracks
- 75-80% with of length 2
- 98% of length less than 10
- A few tracks of length more than 100

Remember: Only 2D feature matching and verification has been done so far, but we seem to have come far!!

Structure from Motion

- $11m + 3n$ parameters for m cameras and n points. $2mn$ equations mapping each point in each camera
- Recover camera and structure. Minimize reprojection error across all of them using a non-linear minimization step. This needs good initialization
- Not possible to do them all together. So, start with one pair of cameras and incrementally add more cameras
- Adjust points and cameras to reduce global reprojection error after new cameras are added

Modern digital cameras store a lot of metadata in the images as EXIF tags, including the focal length!

Assume: Only focal length is the unknown intrinsic parameter!

Incremental SfM

- Find a strong starting pair of cameras. These should have a large number of points in common and a large baseline
- Find a pair with a large number of matches. Compute a planar homography from the matches. The pair is good if the error from the homography is *high*!
- Select the pair with the lowest percentage of inliers to homography using RANSAC
- Estimate the essential matrix for the camera pair
- Reconstruct cameras and common points using the essential matrix
- Perform **bundle adjustment** to minimize reprojection error

Adding Views

- While there are more connected cameras
  - Pick an image that sees most number of 3D points so far
  - Estimate pose of the camera using DLT and known 3D points. Perform a local bundle adjustment to correct new camera pose
  - Triangulate new points (if any) and add to the collection
  - Perform a global bundle adjustment on all cameras and points, using a non-linear optimization step
- Can remove outlier tracks altogether
- Can add a small group of camera views together, instead of one at a time

Bundle Adjustment

- Find $\mathbf{P}, \mathbf{X}$ that minimizes (with visibility indicator w_{ij})

$$g(\mathbf{P}, \mathbf{X}) = \sum_i^m \sum_j^n w_{ij} \, ||\mathbf{p}_{ij} - \mathbf{P}_i \mathbf{X}_j||^2$$

- Write it as $g(\mathbf{P}, \mathbf{X}) = ||\mathbf{A} - \mathcal{P}(\mathbf{P}, \mathbf{X})||^{\mathbf{2}}$ where $\mathcal{P}$ is the non-linear camera projection function
- Linearize $\mathcal{P}(\mathbf{P}, \mathbf{X}) = \mathcal{P}(\mathbf{P_0}, \mathbf{X_0}) + \mathbf{J} \begin{bmatrix} \Delta\mathbf{P} \\ \Delta\mathbf{X} \end{bmatrix}$
- Iterative solution using Levenberg-Marquardt method
- A sparse problem as the indicator w_{ij} of point j being visible in camera/image i is sparse.

Practical Aspects

- Heavy computations. Several days to reconstruct 500 images. About half of that time is for the bundle adjustment step
- Several optimizations have been worked on recently. Typical papers:
  “Building Rome in a Day”, ICCV 2009 (U of W)
  “Building Rome on a Cloudless Day”, ECCV 2010 (UNC)
- Combinatorics of pairwise matching is also huge. Use image search approaches to reduce the potential numbers

Building Rome in a Day

Agarwal, Simon, Seitz, Szeliski. ICCV 2009

- Over a million images of the city of Rome
- Pair-wise matching can take 15 years at 2 pairs/sec
- Find 40 most similar words using ...
- Query expansion to increase graph density
- Full bundle adjustment may run till end of time (nearly!)
- Use skeletal graphs to capture overall structure; perform bundle adjustment in local clusters
- Reconstructed Rome in **24 hours** on a 1000-node cluster!!

A recent local experiment for a 400-image Hampi dataset:

Extracting SIFT: **54 minutes**, Image matching: **17.2 hrs**
Bundle adjustment: **12.6 hrs**!!

Videos and Links

Videos from the Rome dataset: Colosseum and St. Peters

Check out **PhotoSynth** from Microsoft

For a digital narrative on an Indian temple, see: See **http://virtualindia.msresearch.in/DH/demo.html**

For more similar, see **http://www.digitalnarratives.net/**

3D Structure Recovery in Practice

Studios to Record Events

Virtualized Reality (CMU, 1995)

- A 51-camera recording setup
- Off-line digitization
- Multi-baseline stereo
- Merge depth maps to get structure

Free-Viewpoint Video (ETH, MPI)

- Multicamera setup, all digital
- Visual hull for quick structure

123D from Autodesk

- Submit your photographs
- Get a 3D model!!

Structure Recovery: Conclusions

- A problem that has been solved somewhat well
- Many challenges remain, but many have been tackled
- Next generation movies: Watch it from a viewpoint of your choice, decided at view time!!

Thank You!

Thanks to the Computer Vision community worldwide
for many of the figures!!