Sparse 3D Reconstruction

AKA Structure from Motion/SLAM

Presented by Adam Fishman for CSE 576 in Spring 2020

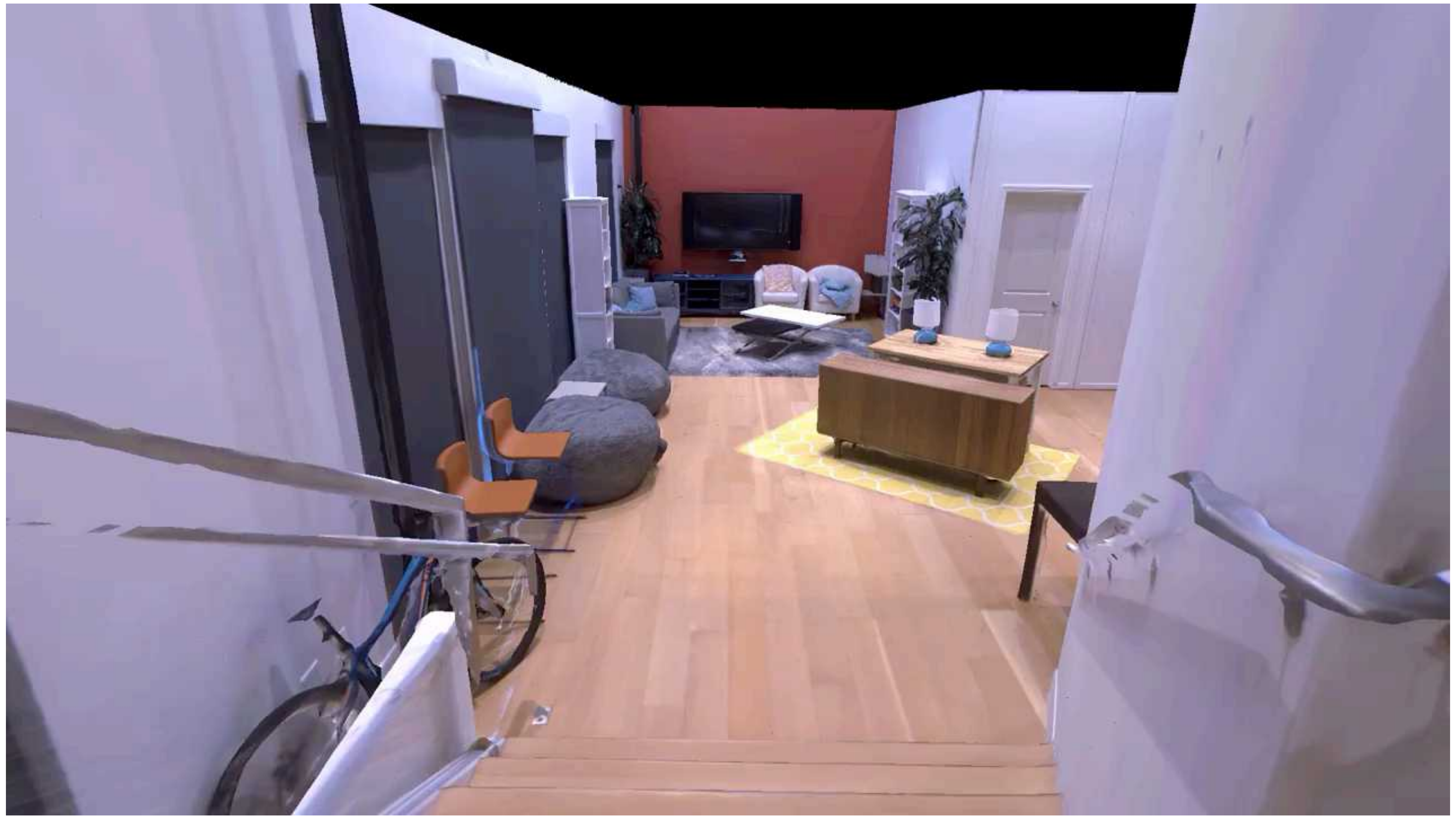
Today's Lecture

- High Level Overview
- Structure from Motion (SfM)
- How SfM relates to SLAM

Image: Facebook

Sparse vs Dense Reconstruction

Sparse[1]



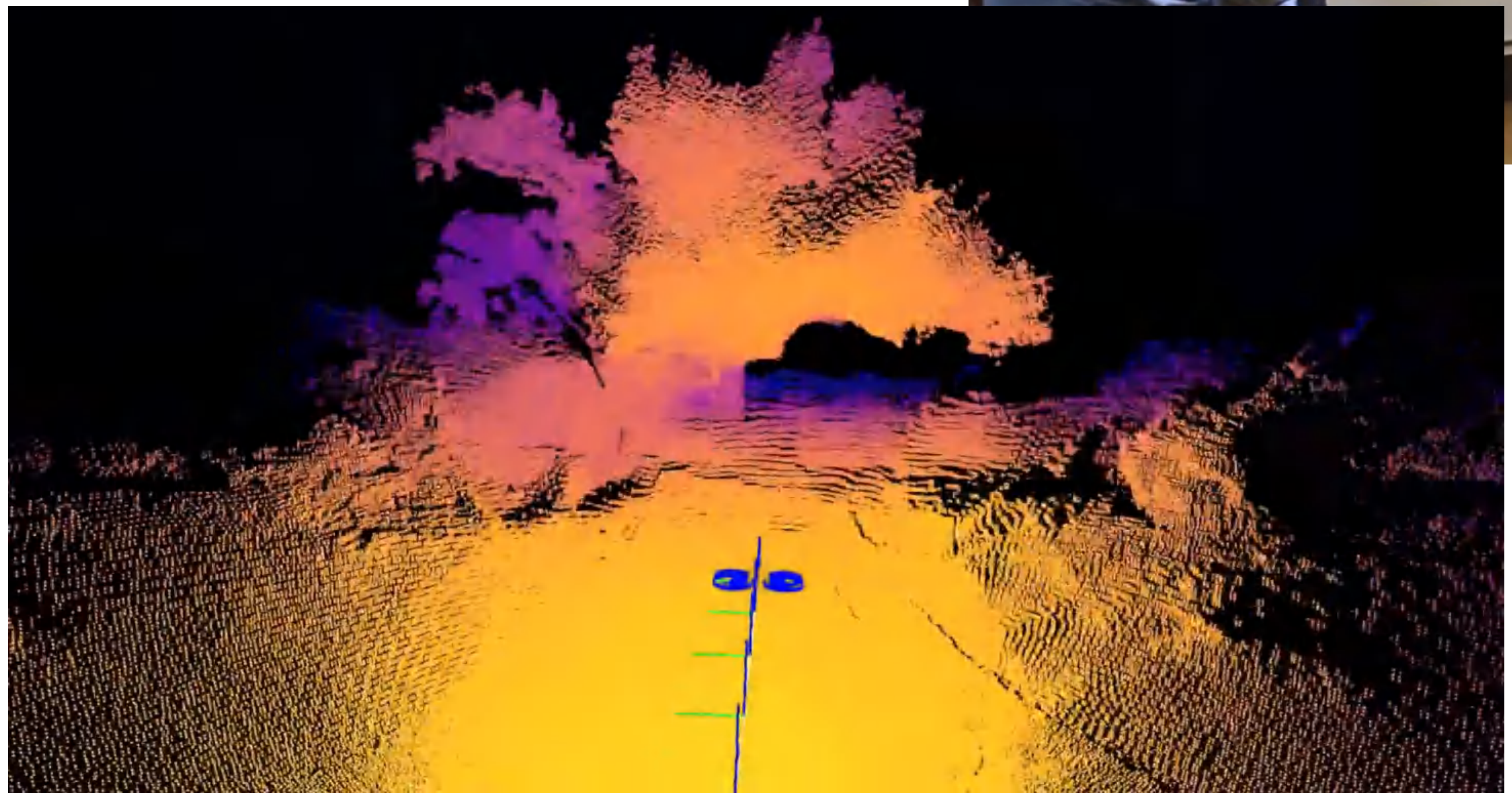
Dense[2]

1. https://www.youtube.com/watch?v=vpTEobpYoTg
2. https://www.youtube.com/watch?v=7YIGT13bdXw

Why?

3D Reconstructions Are Useful

- SLAM (Simultaneous Localization and Mapping) is essential for robot navigation
- 3D reconstructions can be used in VR, games, or movies
- Also, this stuff is just cool



Structure from Motion

- Large Scale 3D Reconstruction
- Useful for:
  - 3D modeling
  - Surveying
  - Virtual and augmented reality
  - Visual effects (“Match moving”)

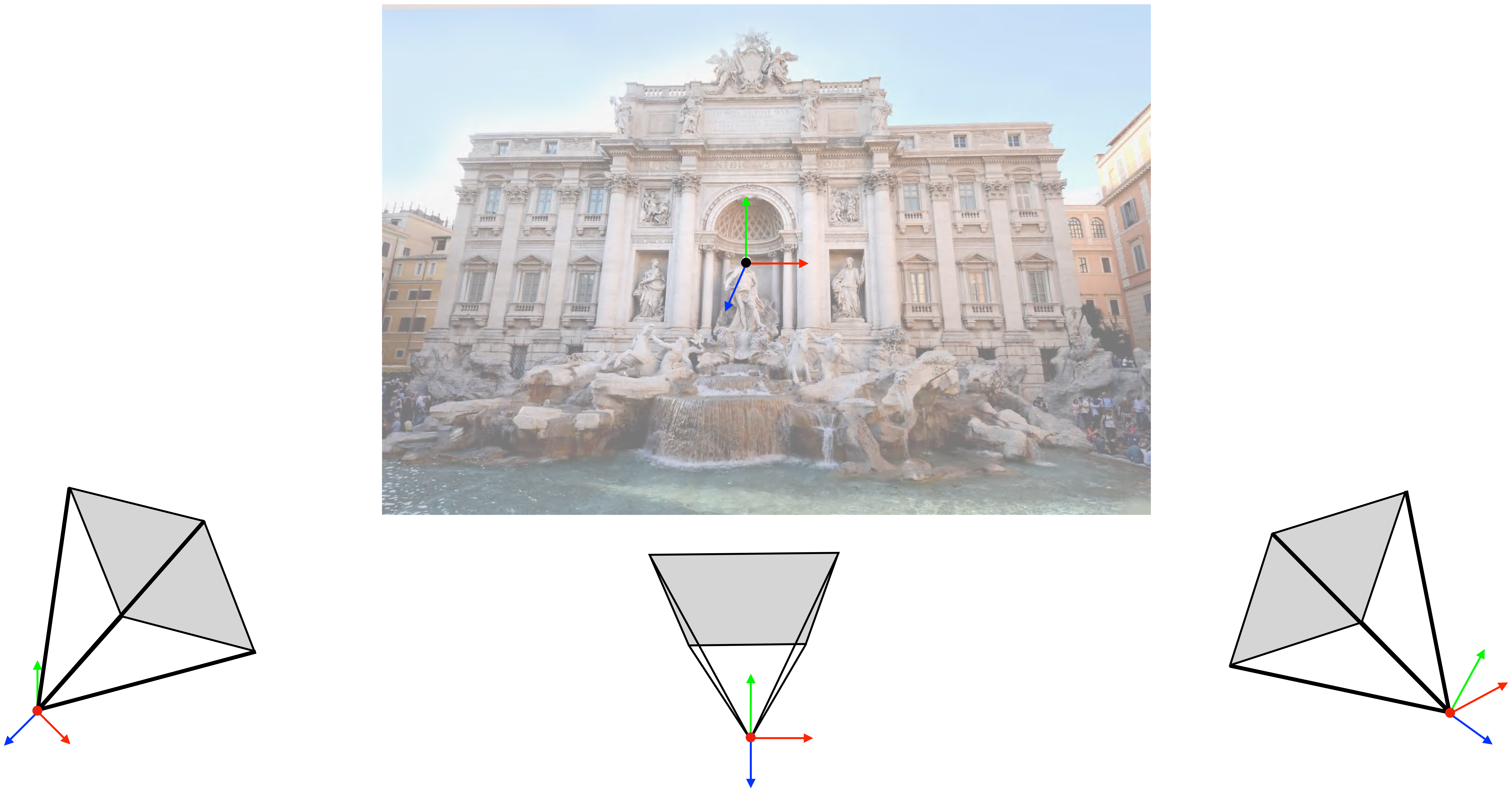
SLAM (Simultaneous Localization and Mapping)

- How a robot or XR headset keeps track of its location
- Can be done with a variety of sensor types, but we will only go into details on the camera
- A good map is necessary for localization
- Building and refining the map requires precise location
- Chicken 🔄 Egg

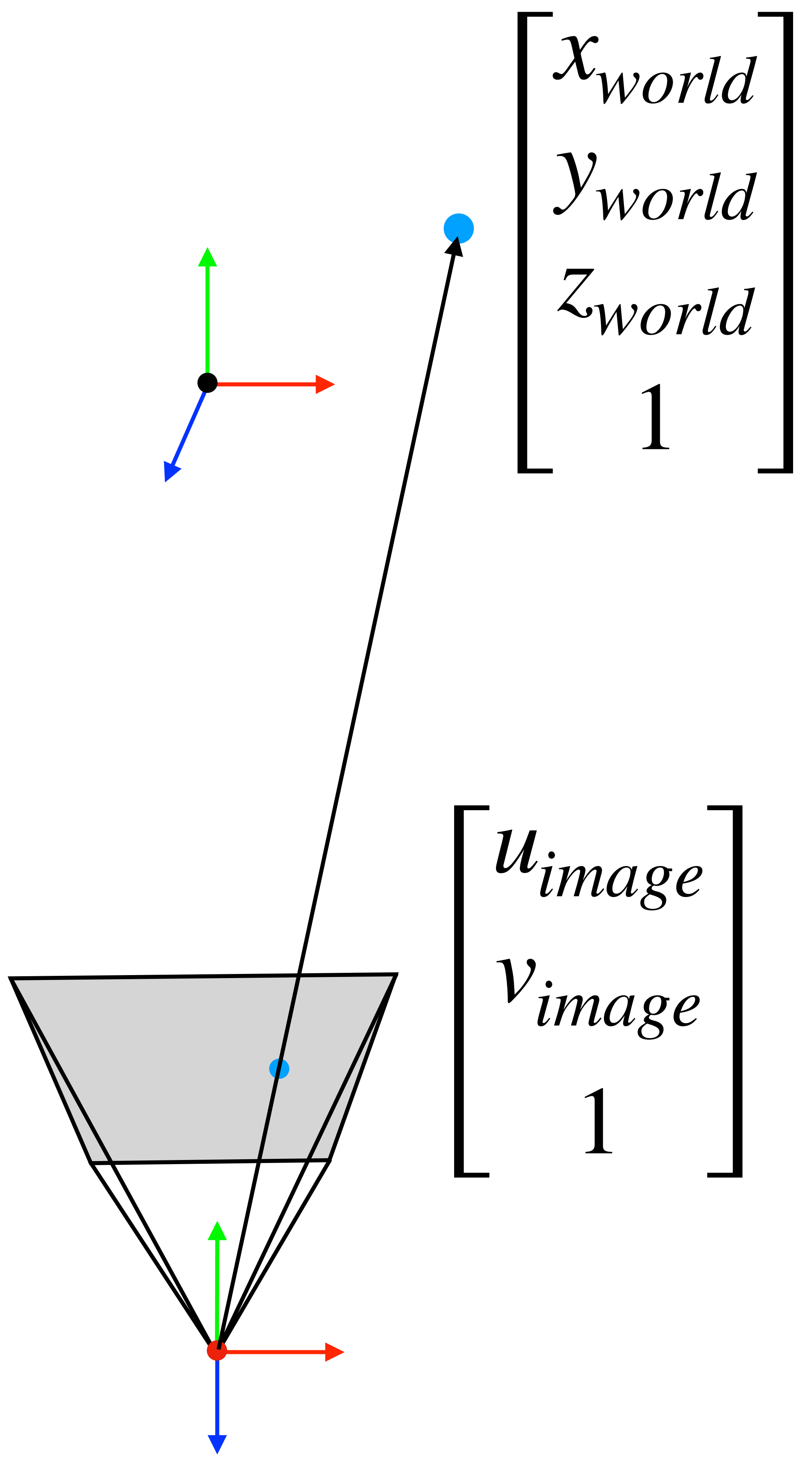
Quick Summary of Projective Geometry

Projective Geometry

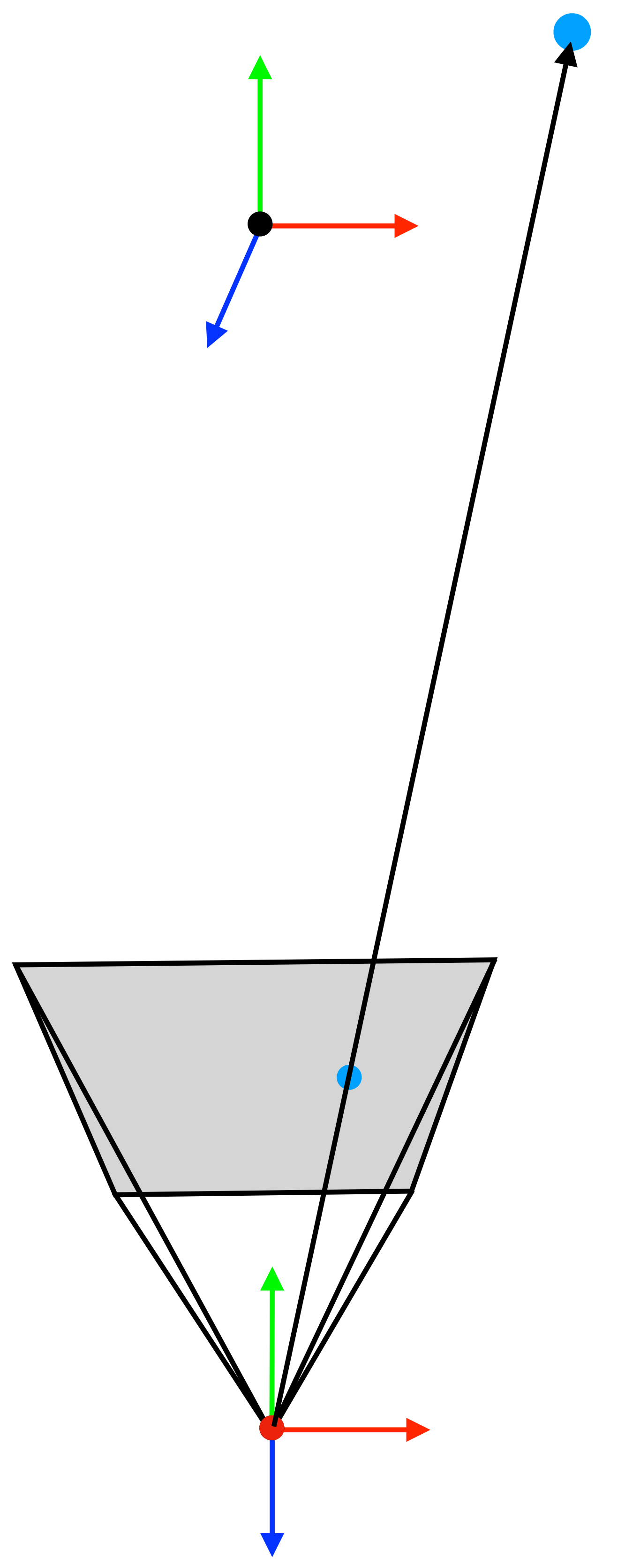
Projective Geometry



Projective Geometry



Projective Geometry



$$\begin{bmatrix} u_{image} \\ v_{image} \\ 1 \end{bmatrix} = \begin{bmatrix} K_{3x4} \end{bmatrix} \begin{bmatrix} R_{3x3} & T_{3x1} \\ 0_{3x1} & 0 \end{bmatrix} \begin{bmatrix} x_{world} \\ y_{world} \\ z_{world} \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} K_{3x4} \end{bmatrix} \triangleq \begin{bmatrix} f \cdot m_x & \gamma & u_0 & 0 \\ 0 & f \cdot m_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

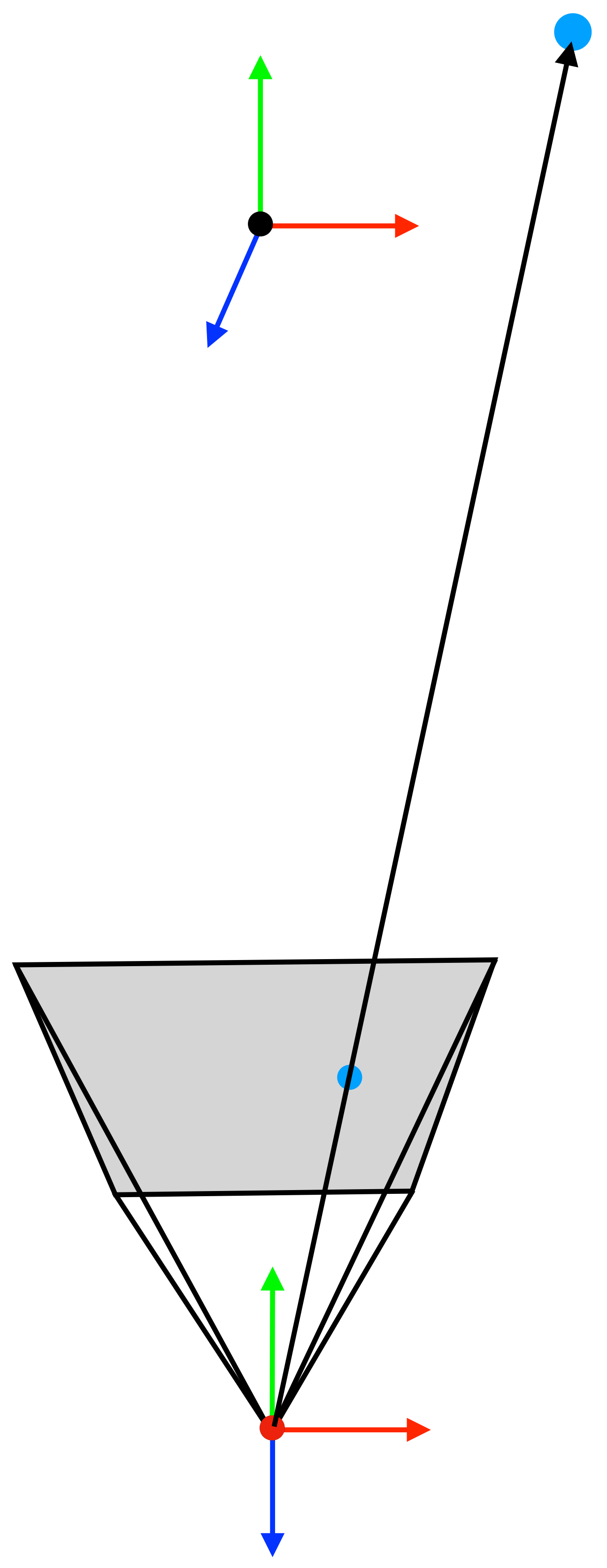
f : Focal distance

m_x, m_y : Scale factors of world units to pixels

γ : Axis skew

u_0, v_0 : Principal point, i.e. image origin

Projective Geometry



$$[K_{3x4}] \triangleq \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The calibration matrix can be simplified when better calibration data is not available

Structure from Motion

Slides largely by Noah Snavely

Reconstruction From Two views

- Solve for Fundamental matrix / Essential matrix
- Factorize into intrinsics and extrinsics (rotation and translation of camera center)

What about more than two views?

- With several views, it's possible to use analogous methods
- With many views, there is too much accumulated error between the various sensors
- We want our 3D estimate to be as accurate as possible given our many (possibly noisy) images.
- This lends itself well to an optimization-based method

Structure from motion

- Given many images, how can we
  - a) figure out where they were all taken from?
  - b) build a 3D model of the scene?

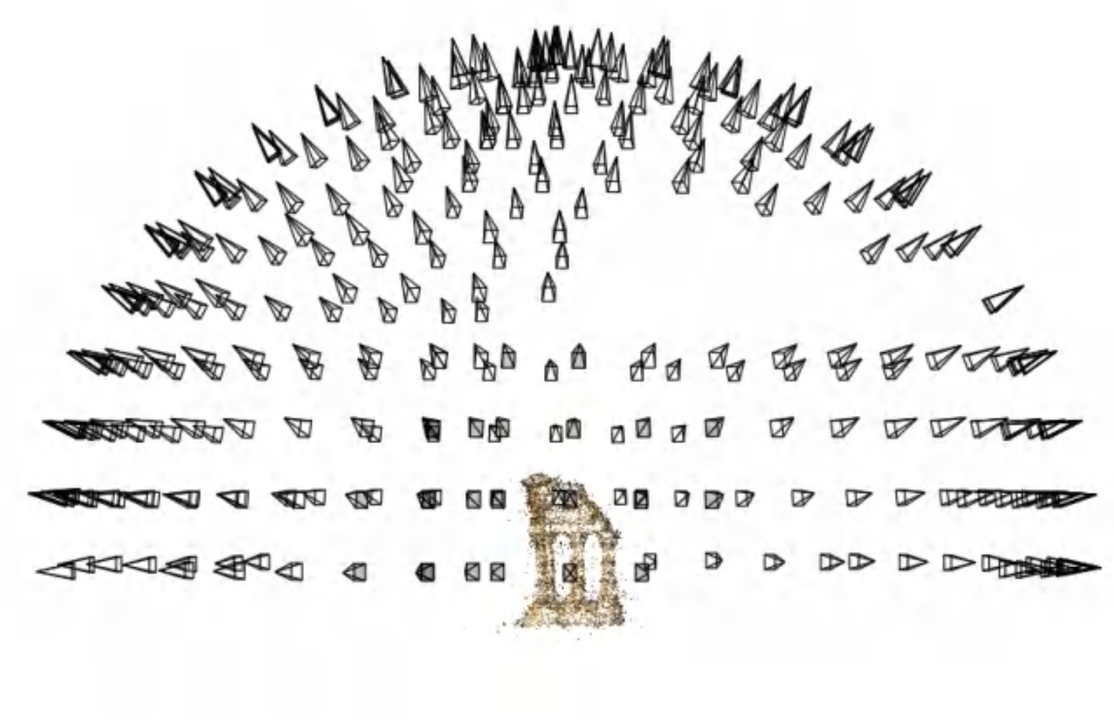
- This is (roughly) the structure from motion problem

Structure from motion

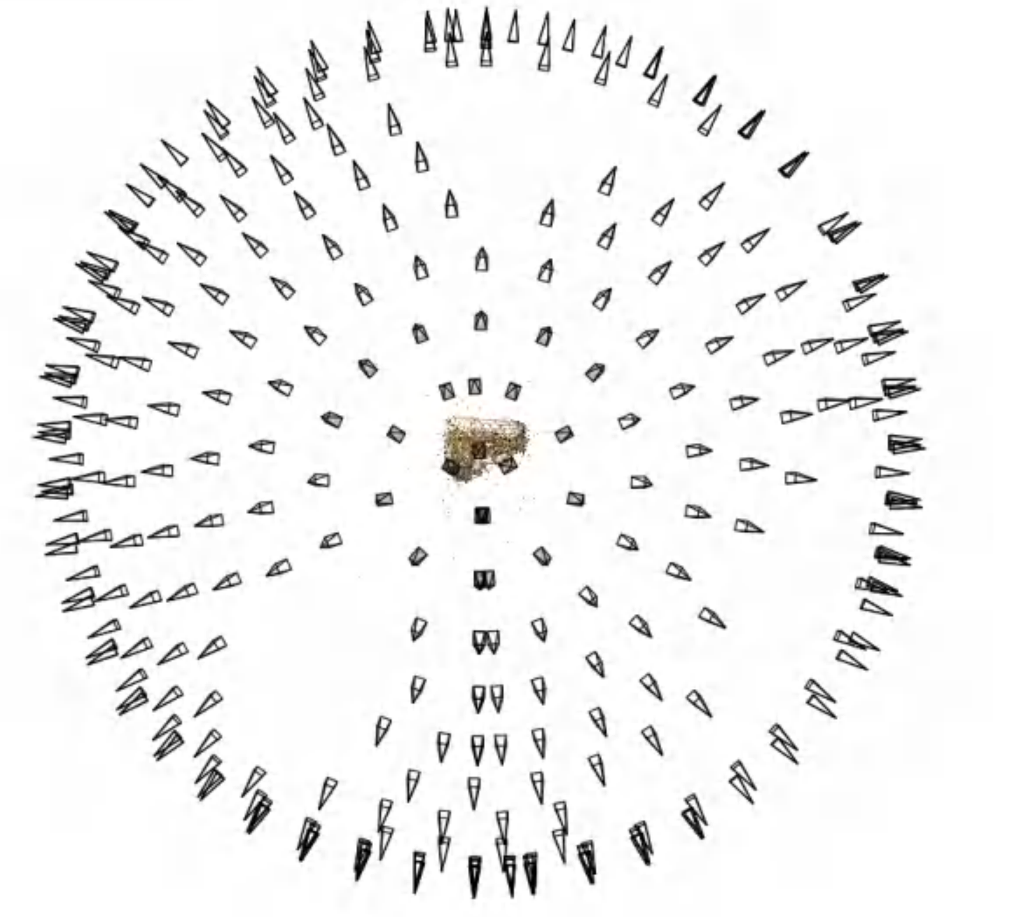
- Input: images with points in correspondence $p_{i,j} = (u_{i,j}, v_{i,j})$

- Output
  - Structure: 3D location $\mathbf{x}_i$ for each point p_i
  - Motion: camera parameters $\mathbf{R}_j$, $\mathbf{t}_j$ possibly $\mathbf{K}_j$

- Objective function: minimize *reprojection error*



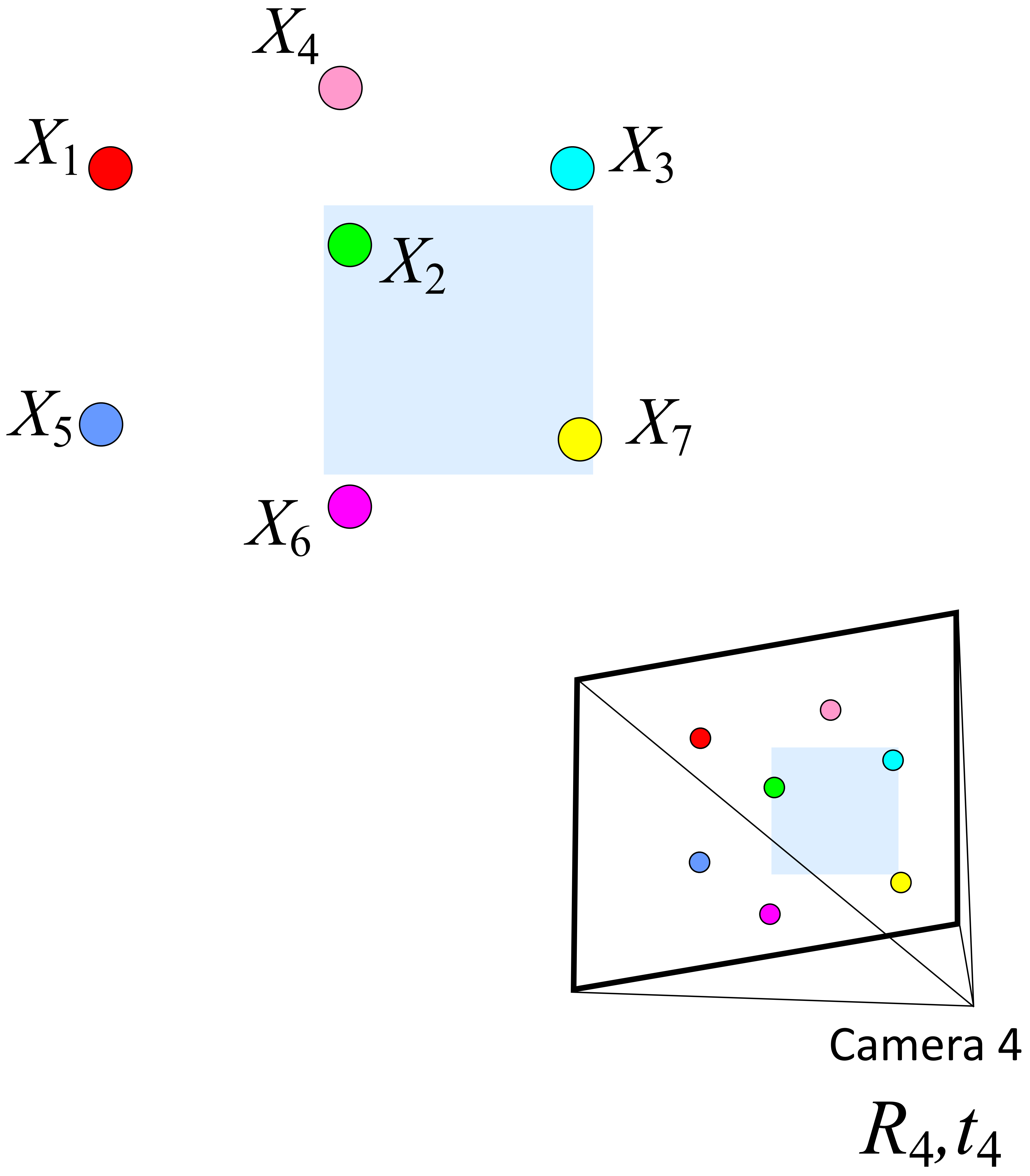
Reconstruction (side)



(top)

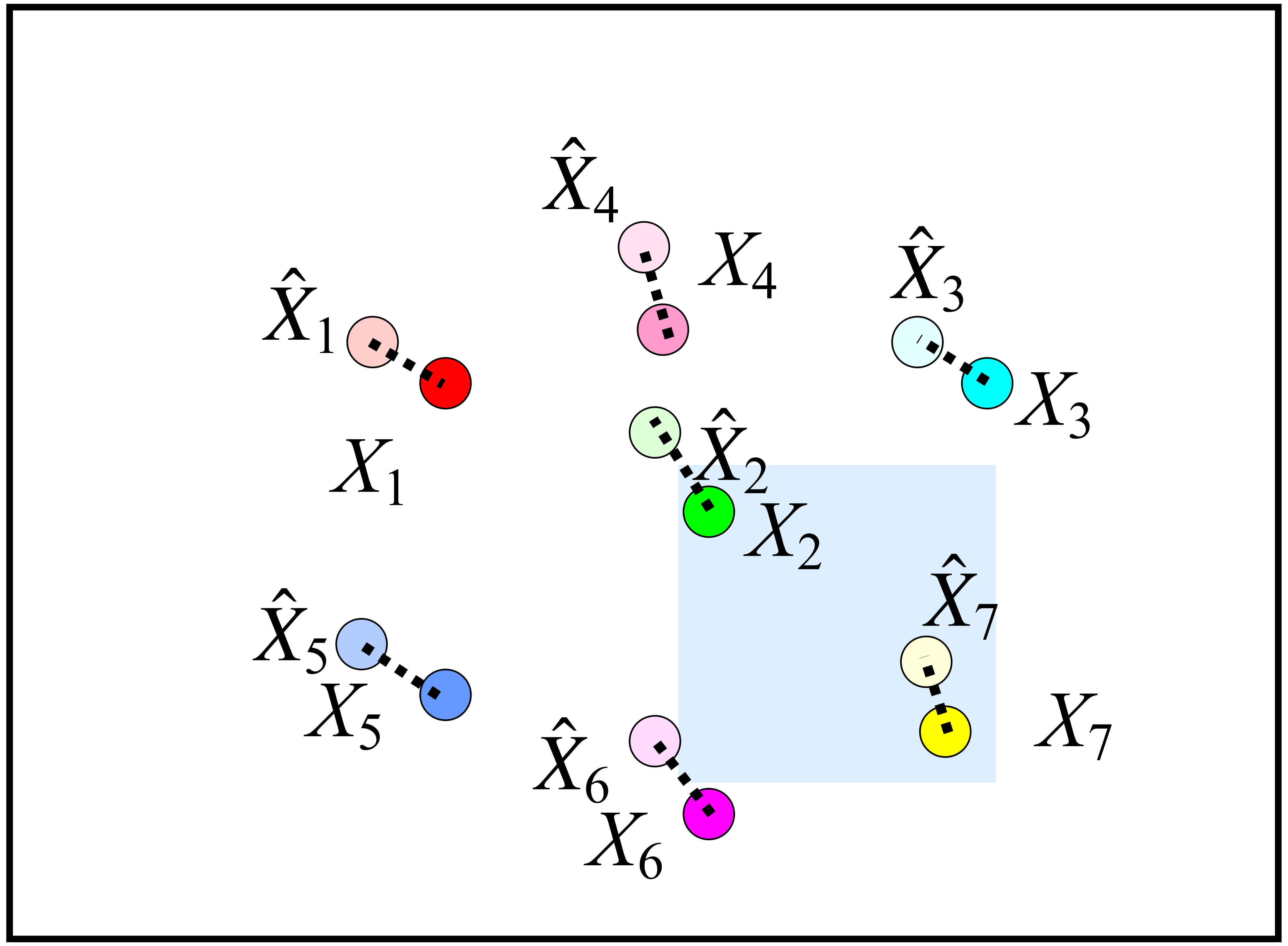
Reprojection Error

- Take estimated 3D point positions and camera pose
- Project 3D points onto image using camera projection model
- Calculate Euclidean error in image space

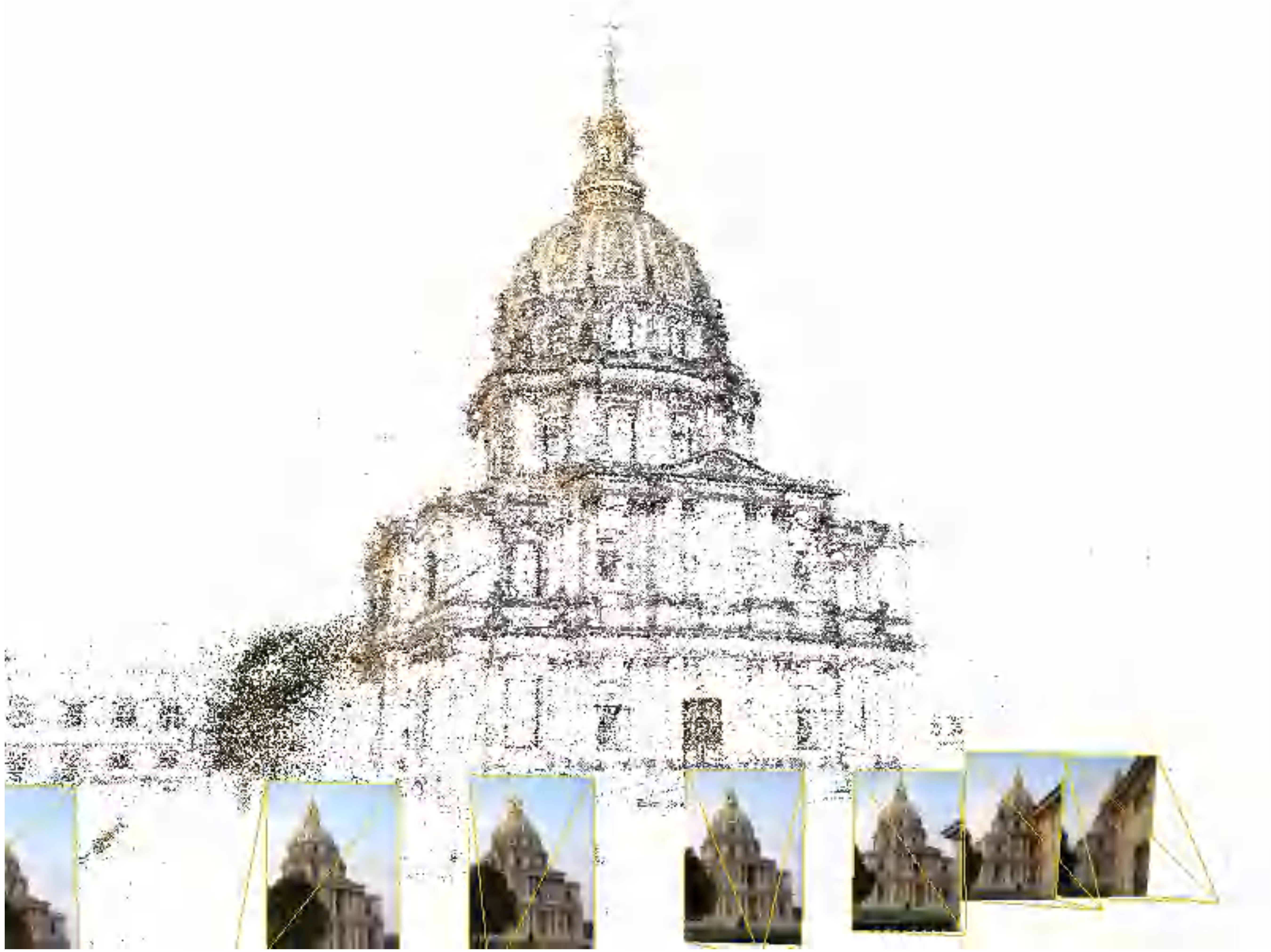


Reprojection Error

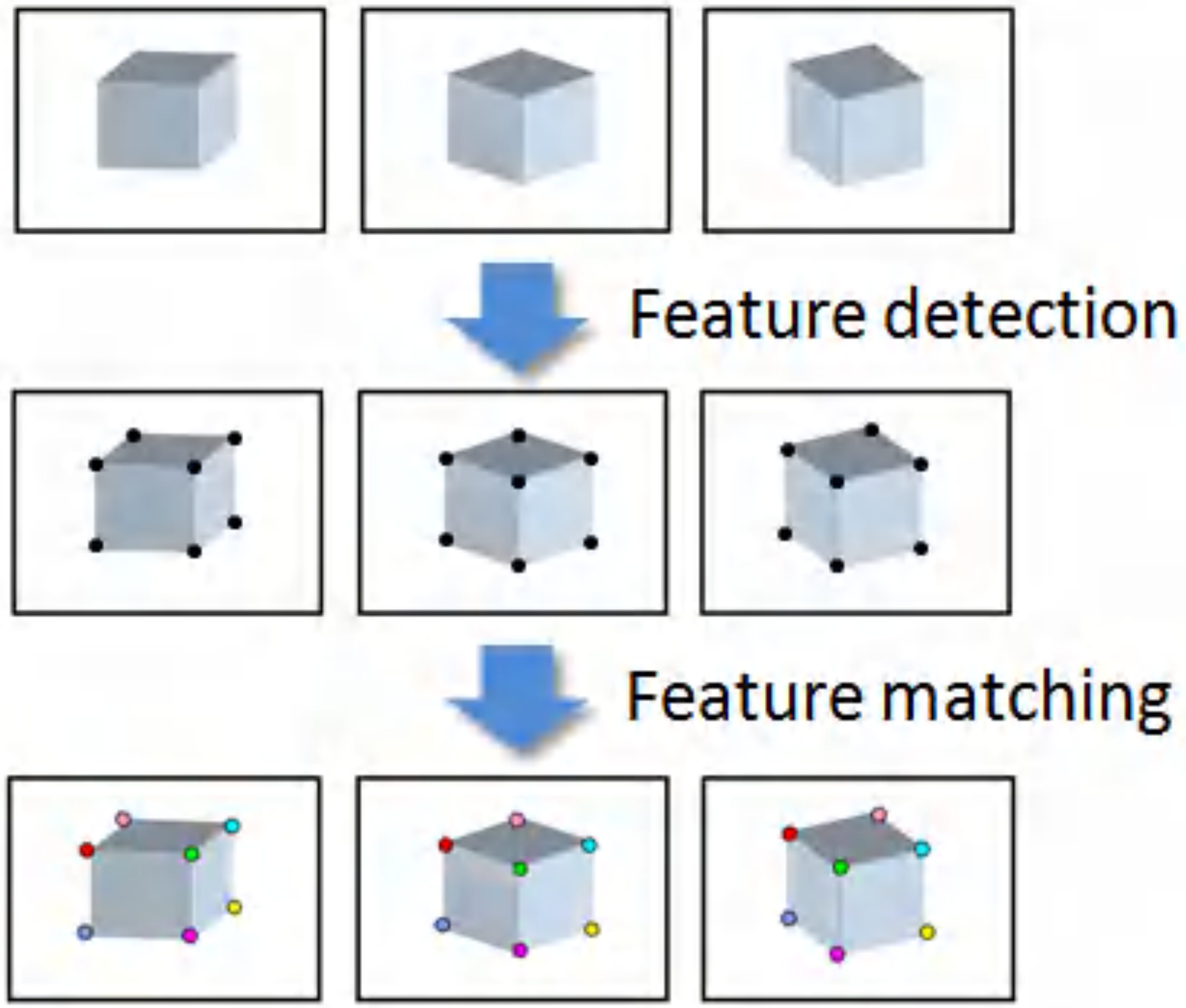
- Take estimated 3D point positions and camera pose
- Project 3D points onto image using camera projection model
- Calculate Euclidean error in image space



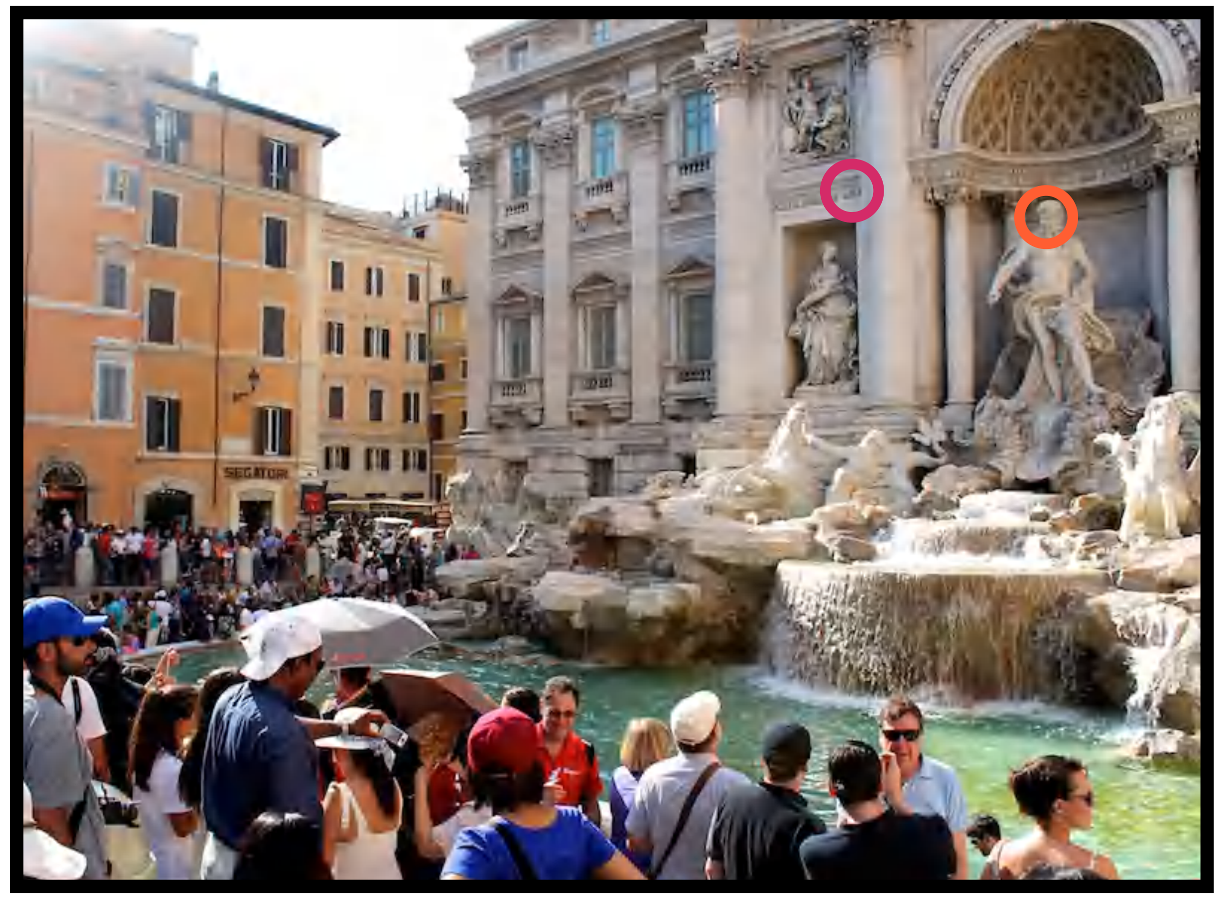
Also doable from video

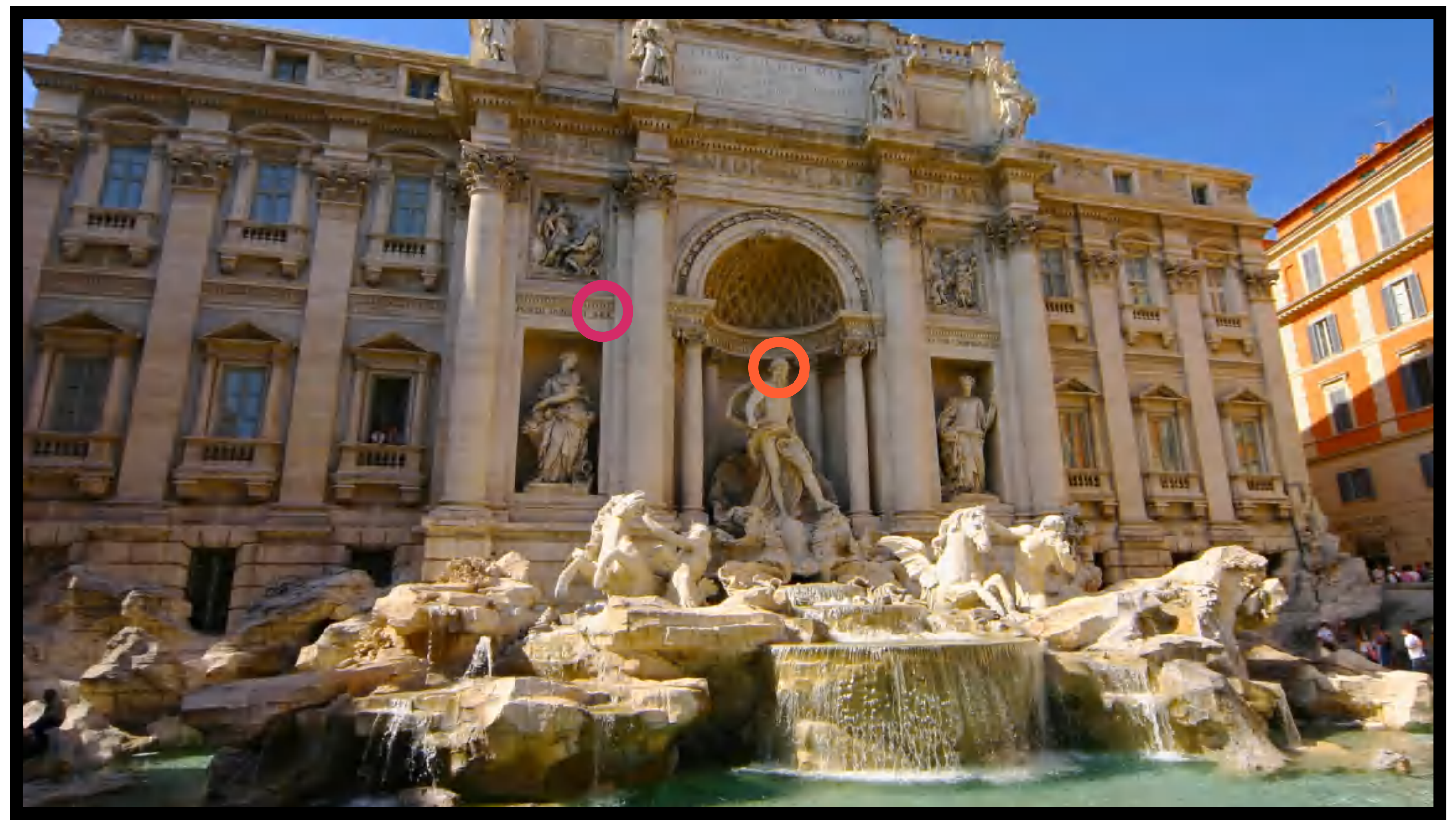


Input



3D Matching





Camera calibration and triangulation

- Suppose we know 3D points
  - And have matches between these points and an image
  - How can we compute the camera parameters?

- Suppose we have known camera parameters, each of which observes a point
  - How can we compute the 3D location of that point?

Structure from motion

- SfM solves both of these problems *at once*
- A kind of chicken-and-egg problem
  - (but solvable)

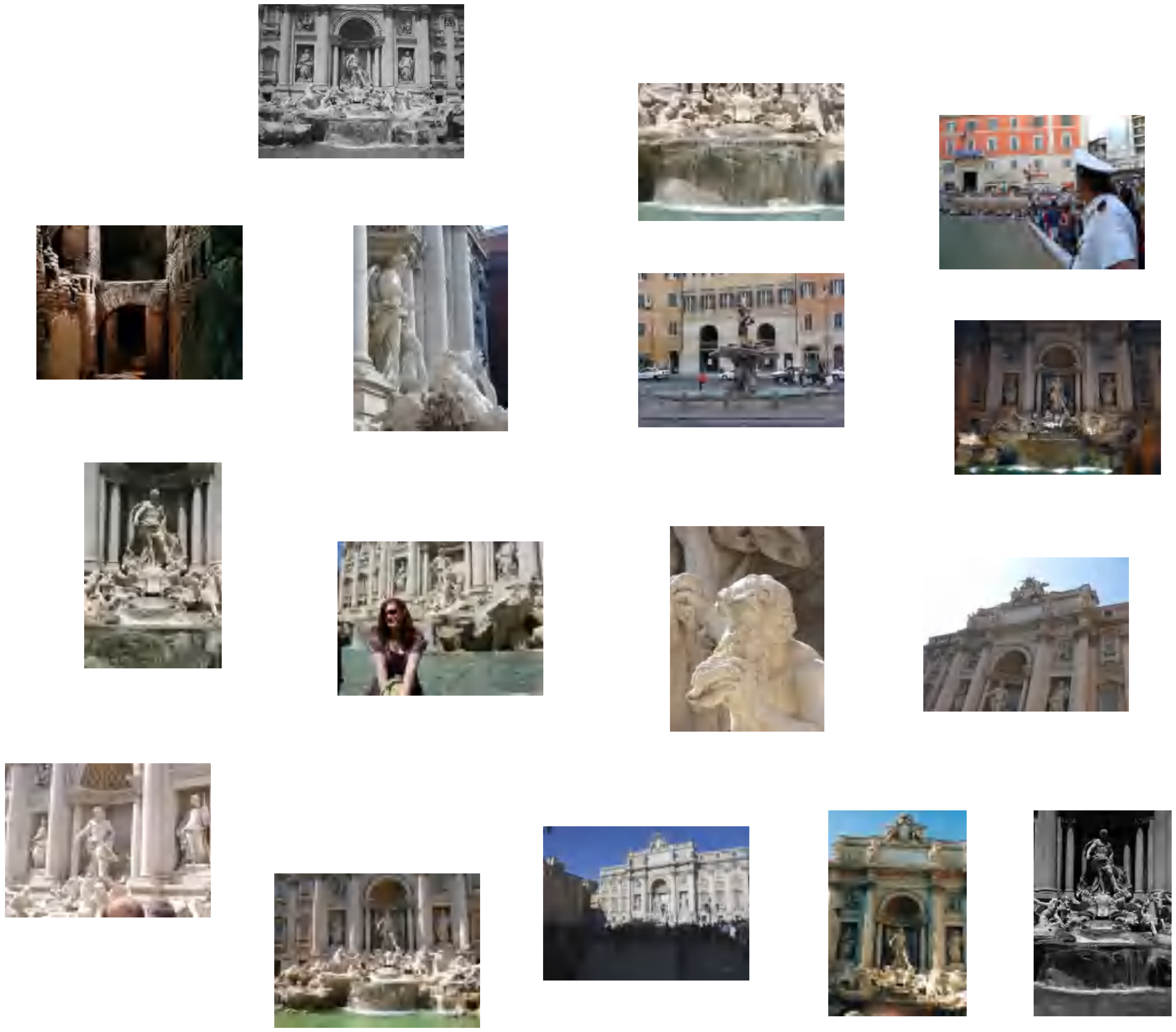
Photo Tourism

First step: how to get correspondence?

- Feature detection and matching

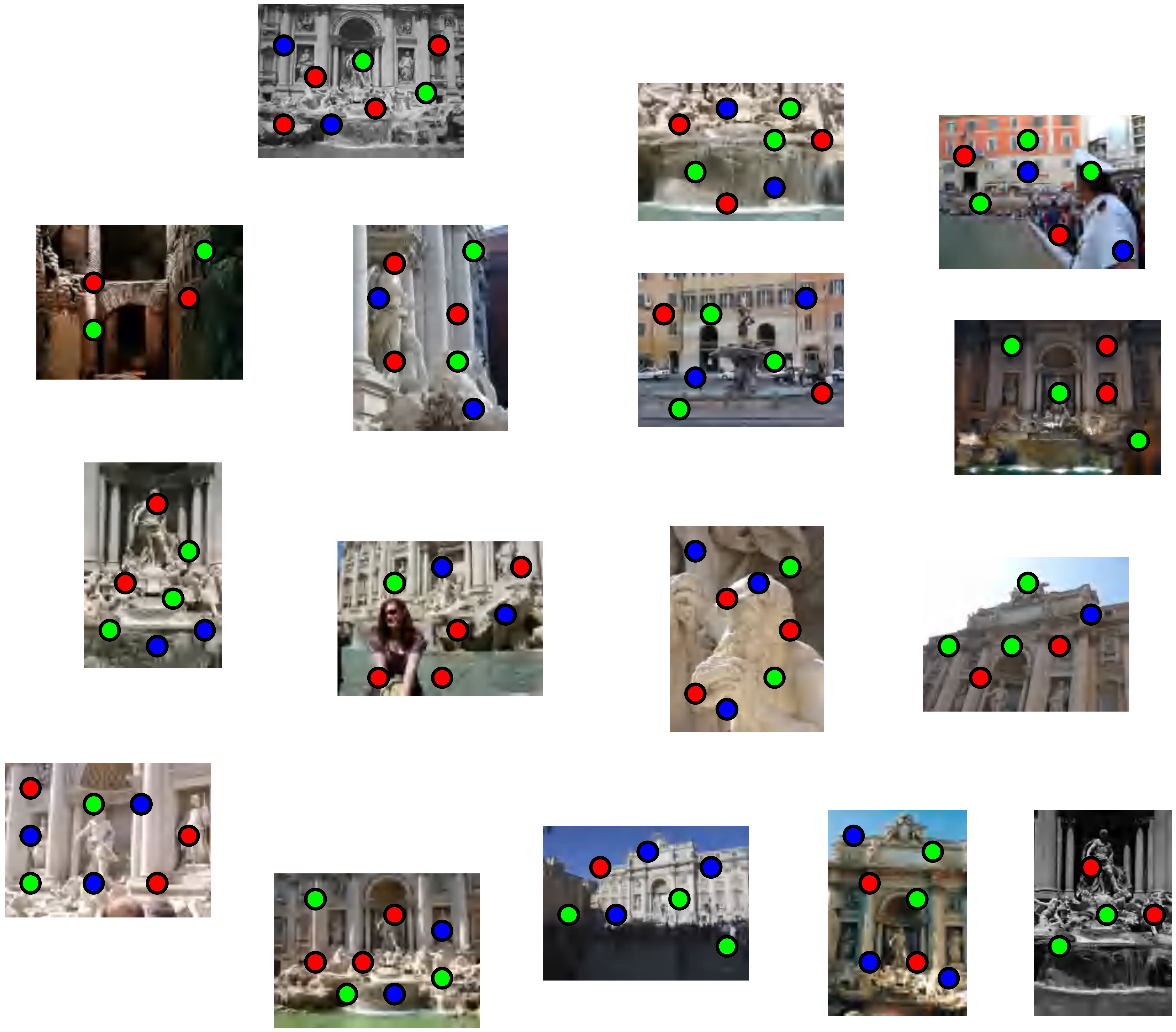
Feature Detection

- Detect features using SIFT [Lowe, IJCV 2004]



Feature Detection

- Detect features using SIFT [Lowe, IJCV 2004]



Feature Matching

- Detect features using SIFT [Lowe, IJCV 2004]
- Match features between each pair of images
- Refine matching using RANSAC to estimate fundamental matrix between each pair

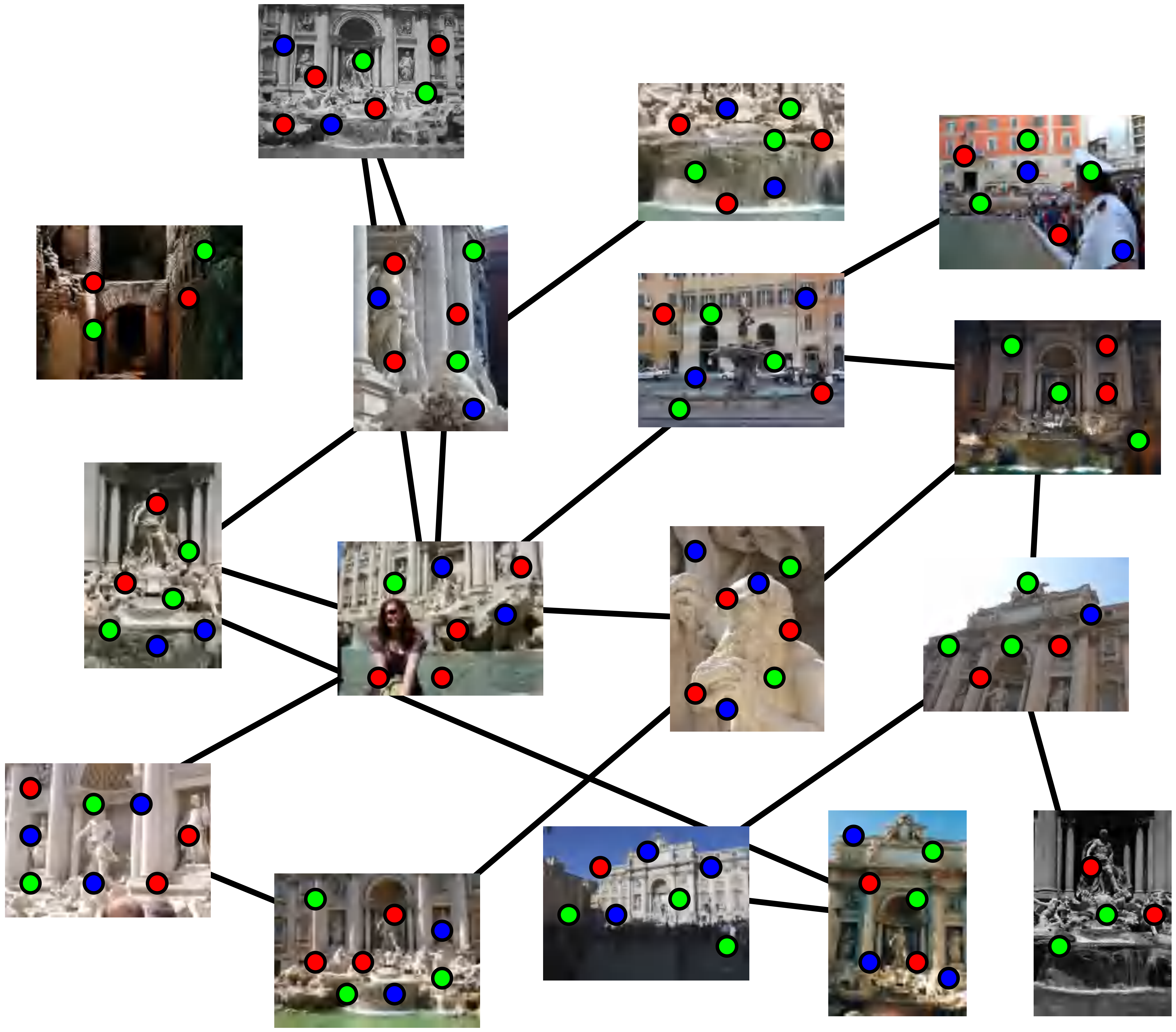
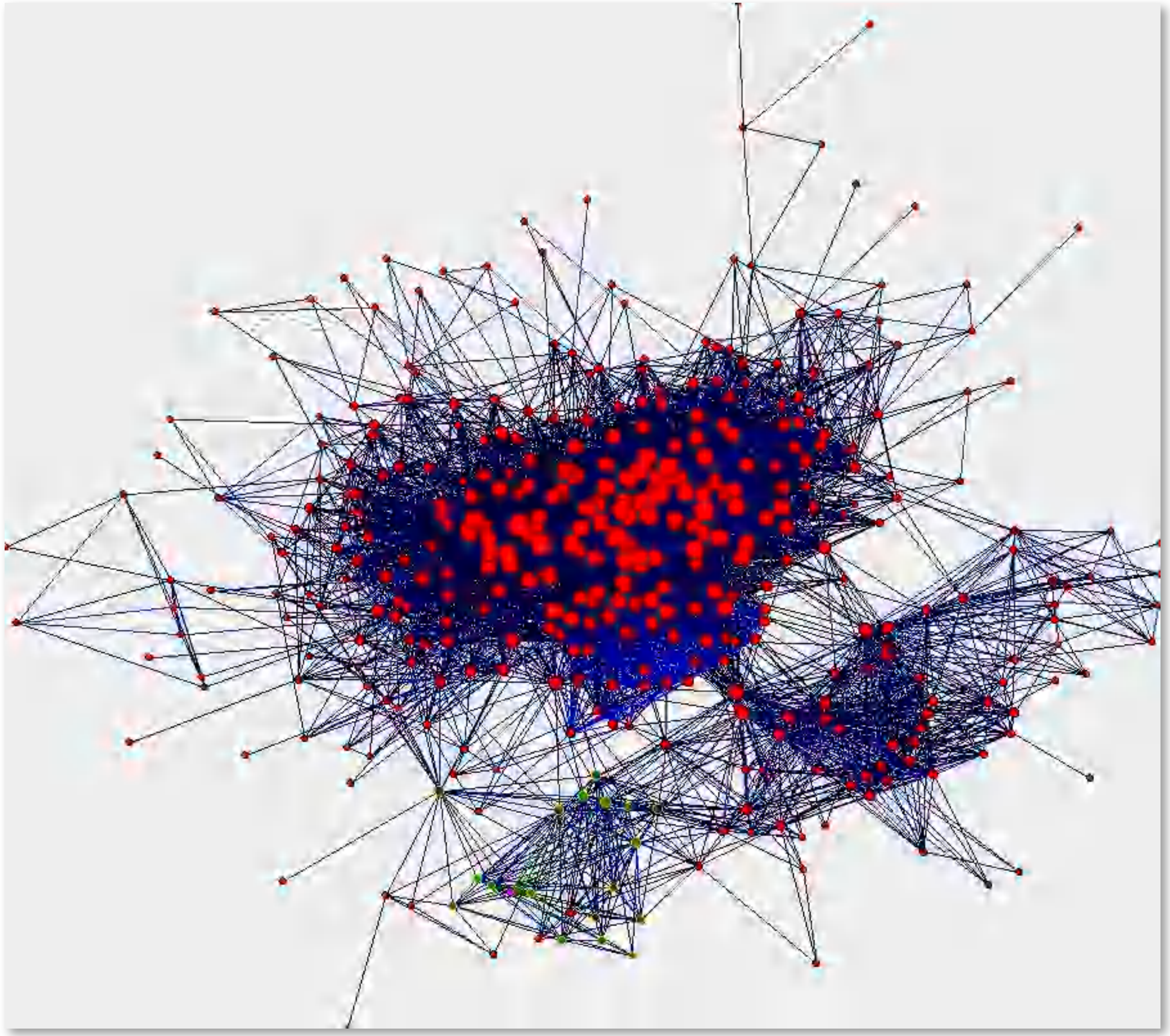


Image connectivity graph

- Graph of connectivity based on matched features



(graph layout produced using the Graphviz toolkit: http://www.graphviz.org/)

Correspondence estimation

- Link up pairwise matches to form connected components of matches across several images

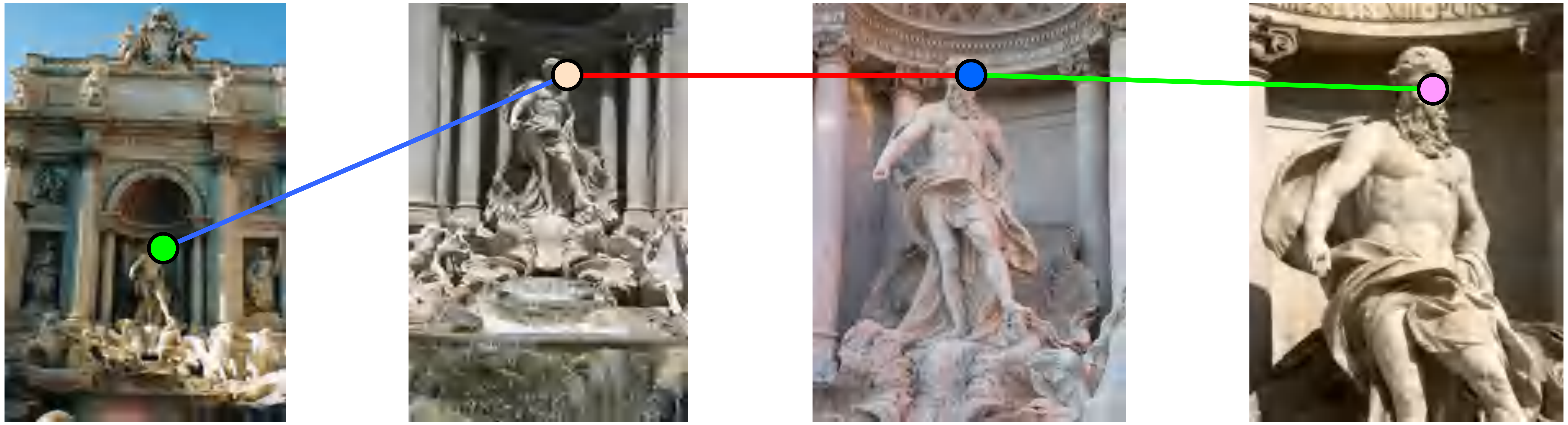
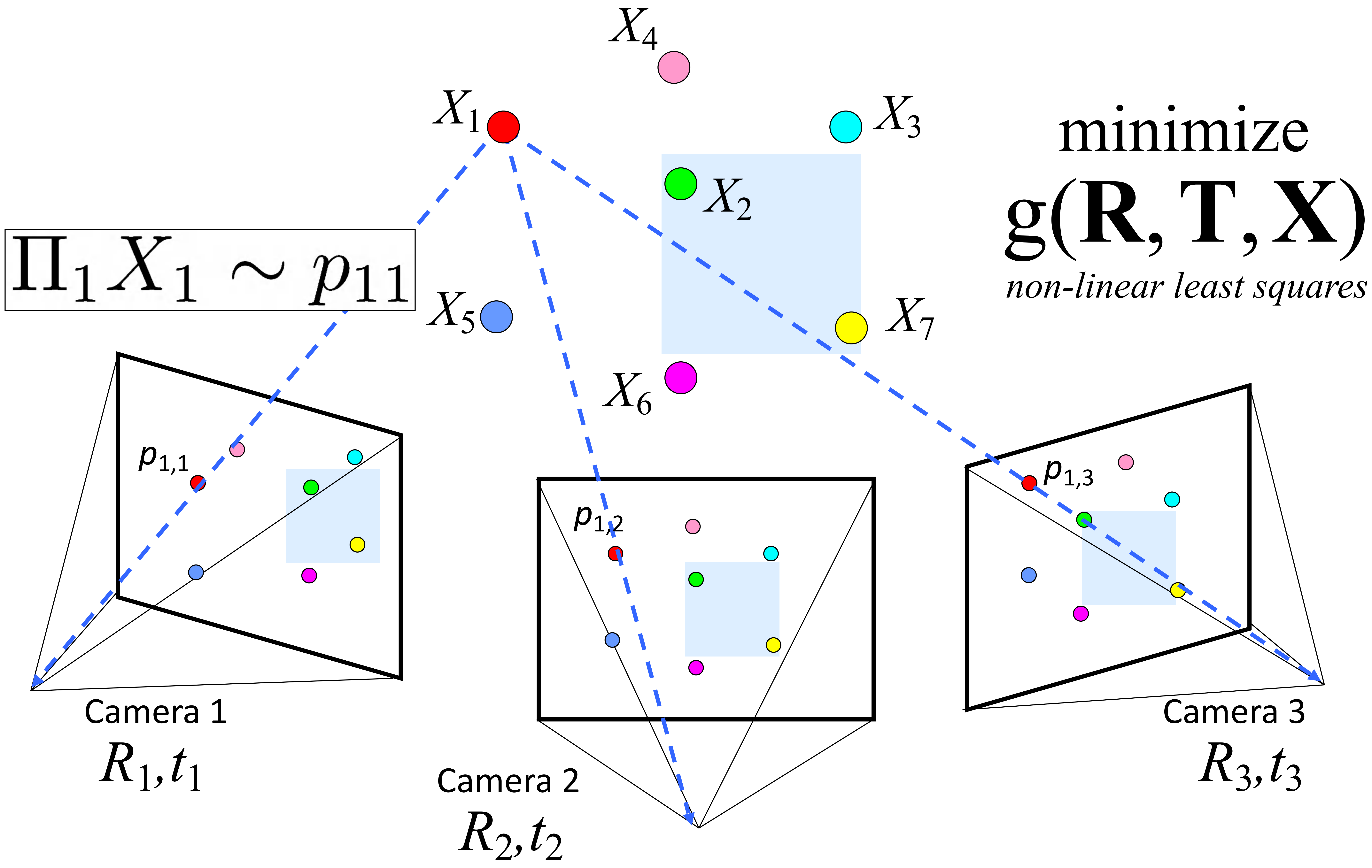


Image 1 Image 2 Image 3 Image 4

Structure from motion



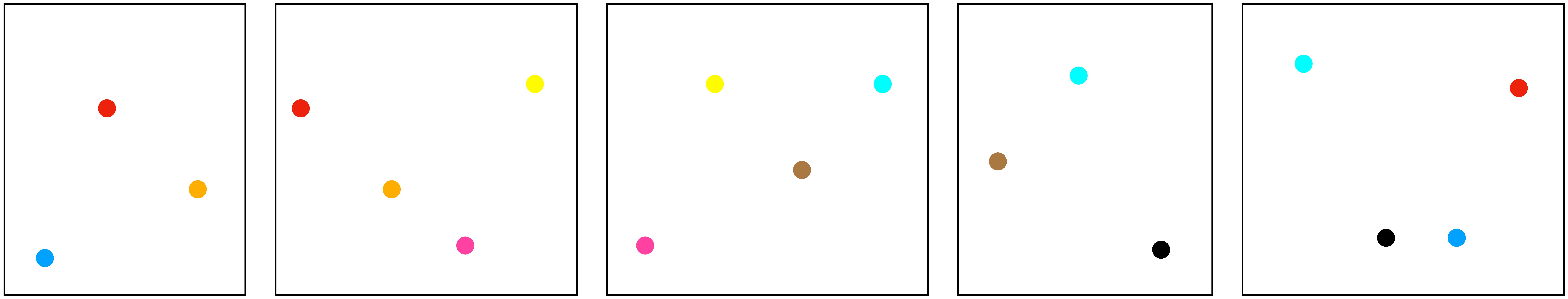
Problem size

- What are the variables? 3D points, camera parameters
- How many variables per camera? Depends on calibration, but 7ish (T, R, f)
- How many variables per point? 3

- Trevi Fountain collection

  466 input photos
  \+ > 100,000 3D points
  = very large optimization problem

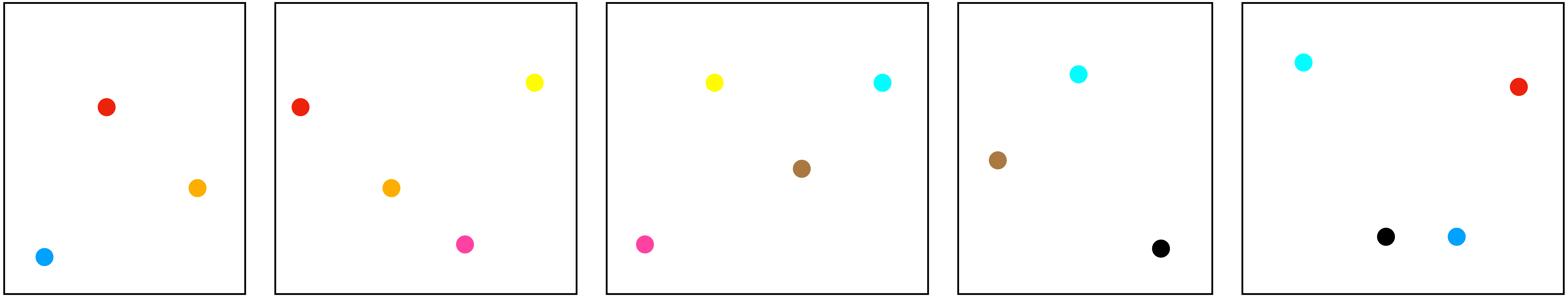
Simple "Bundle Adjustment" (Translation Only)



- Let's say we want to find the translation t_j from the global frame to the j^{th} image
- For each point $\mathbf{p}_{i,j} = (u_{i,j}, v_{i,j})$ defined in the coordinates of the j^{th} image, we want to match $\mathbf{p}_{i,j}$ and $\mathbf{p}_{i,j+1}$, which would mean we want $\mathbf{p}_{i,j+1} - \mathbf{p}_{i,j} = t_{j+1} - t_j$

Slide adapted from Sameer Agarwal

Simple “Bundle Adjustment” (Translation Only)



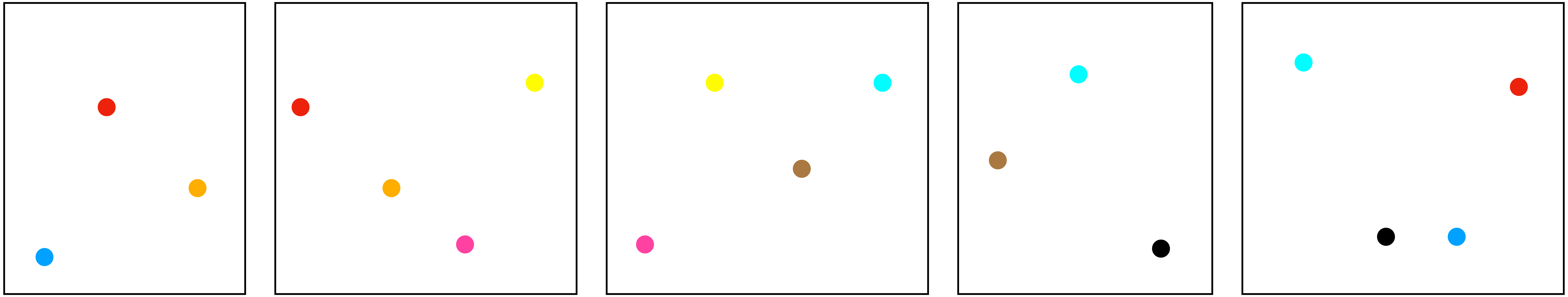
- Leads to an optimization function: $\sum_i \sum_j \delta_{i,j} ||(\mathbf{p}_{i,j+1} - \mathbf{p}_{i,j}) - (t_{j+1} - t_j)||^2$ where

  $\delta_{i,j}$ is 1 when point $\mathbf{p}_i$ appears in image j

- This is a linear least squares problem and can be solved using standard techniques.

Slide adapted from Sameer Agarwal

Simple “Bundle Adjustment” (Translation Only)



- There will be ambiguity in the solution unless we fix the location of one of the frames. This is because our constraints are defined in terms of distance, so the solution will be unique up to a similarity transform.

- This is called *Gauge Ambiguity*

Slide adapted from Sameer Agarwal

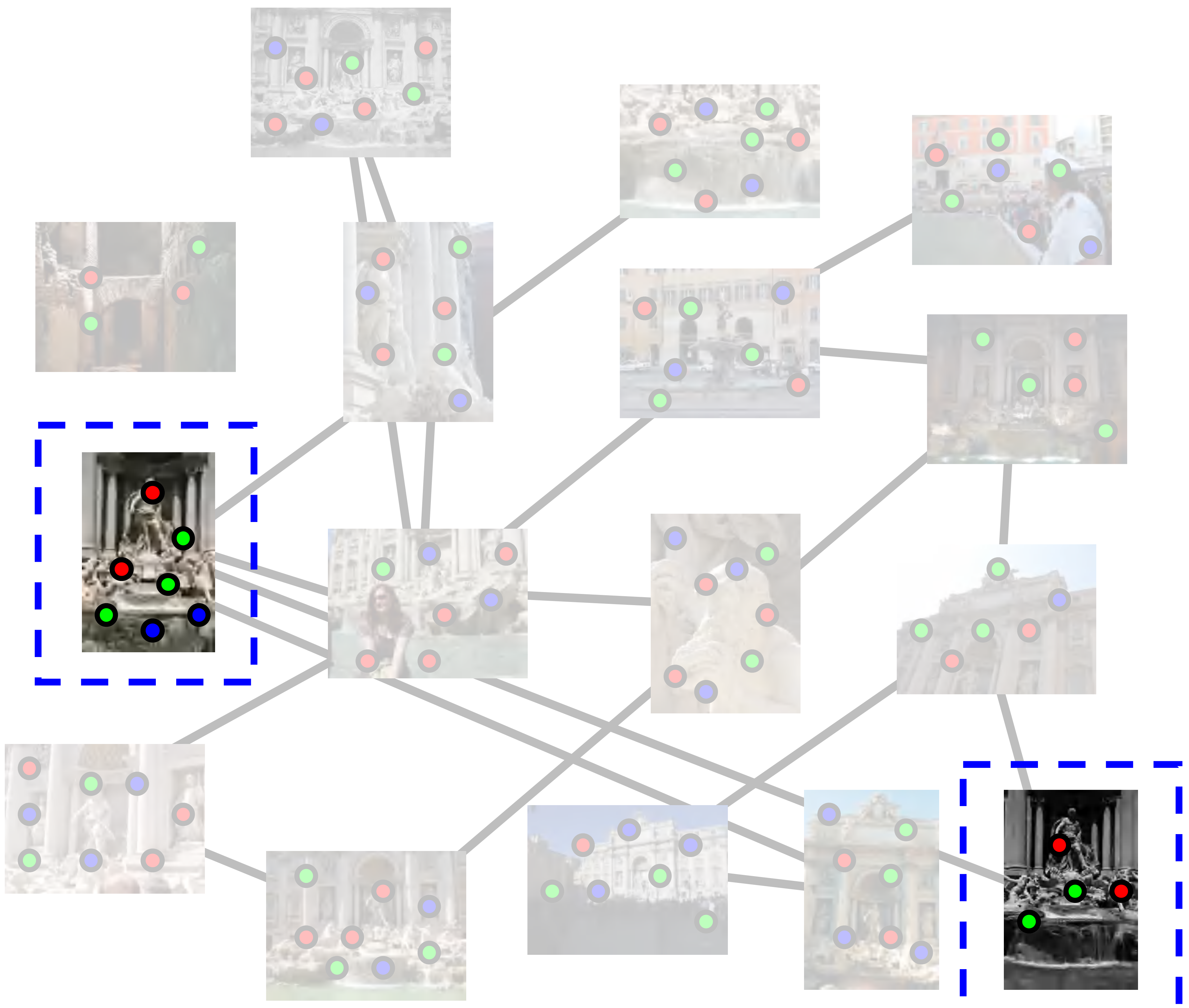
Structure from motion

- Minimize sum of squared reprojection errors:

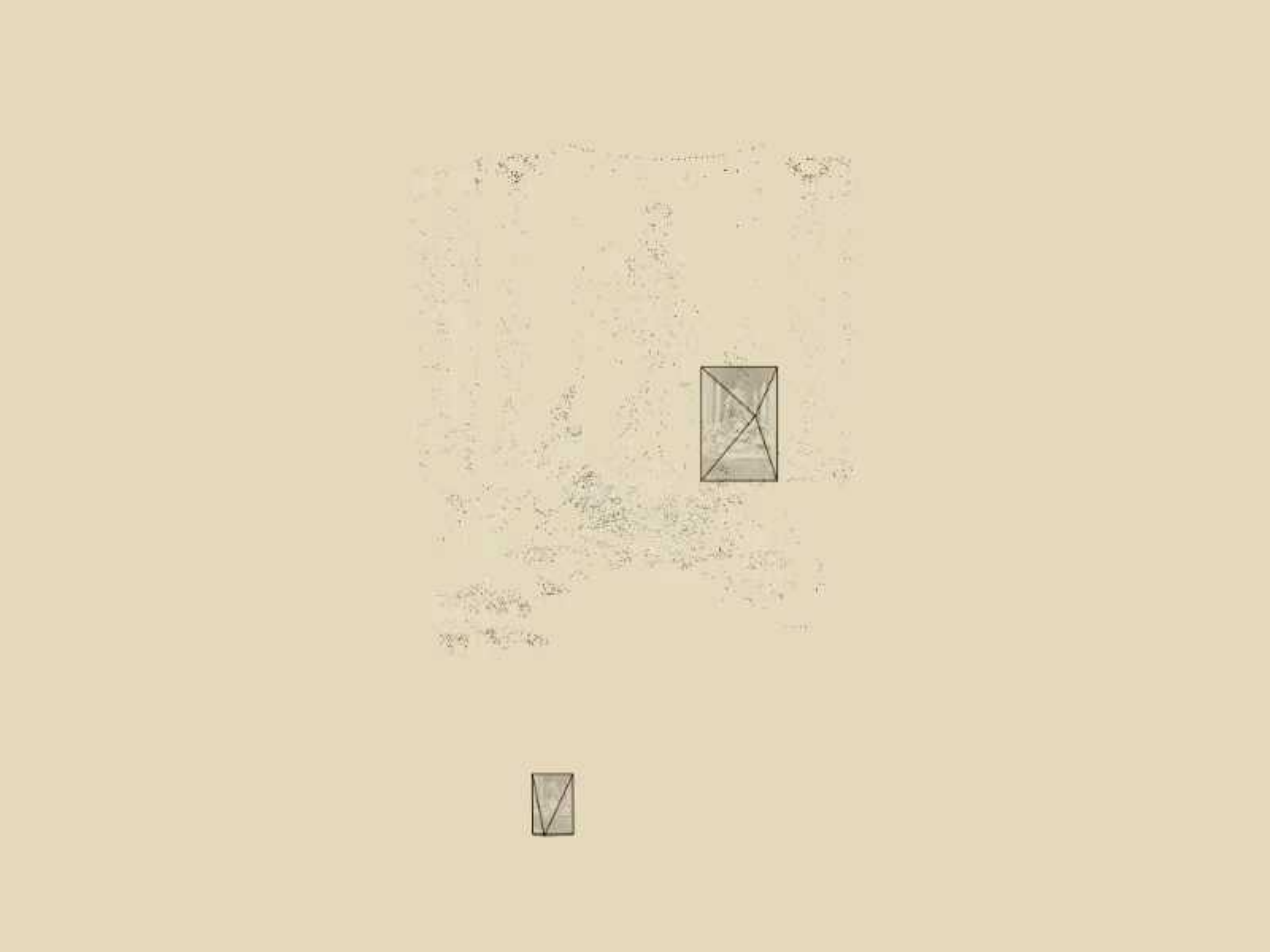
$$g(\mathbf{X}, \mathbf{R}, \mathbf{T}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \underbrace{w_{ij}}_{\substack{\textit{indicator variable}: \\ \text{is point } i \text{ visible in image } j\text{ ?}}} \cdot \left\| \underbrace{\mathbf{P}(\mathbf{x}_i, \mathbf{R}_j, \mathbf{t}_j)}_{\substack{\textit{predicted} \\ \text{image location}}} - \underbrace{\begin{bmatrix} u_{i,j} \\ v_{i,j} \end{bmatrix}}_{\substack{\textit{observed} \\ \text{image location}}} \right\|^2$$

- Minimizing this function is called *bundle adjustment*
  - Optimized using non-linear least squares, e.g. Levenberg-Marquardt

Incremental structure from motion



Incremental structure from motion

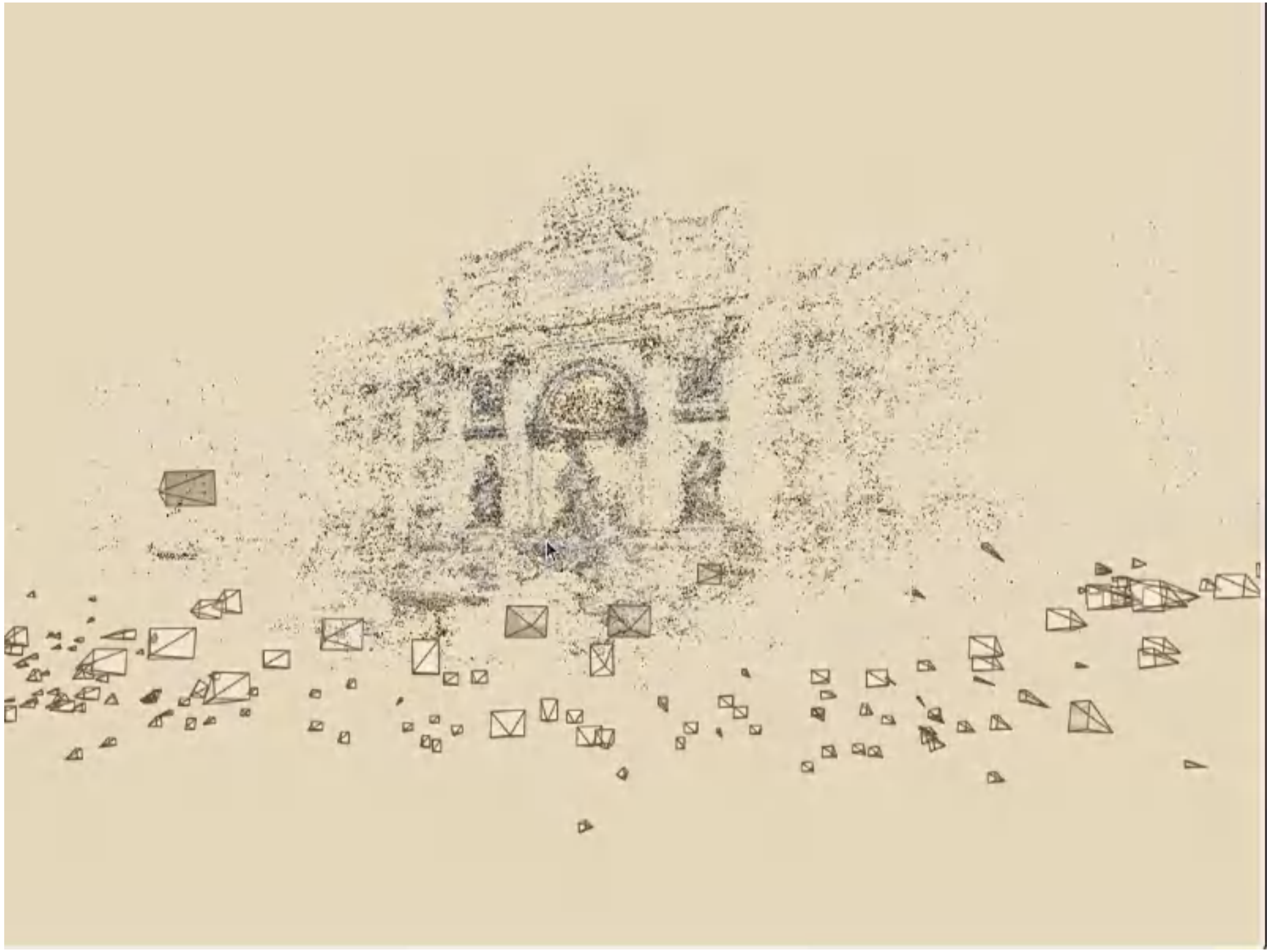


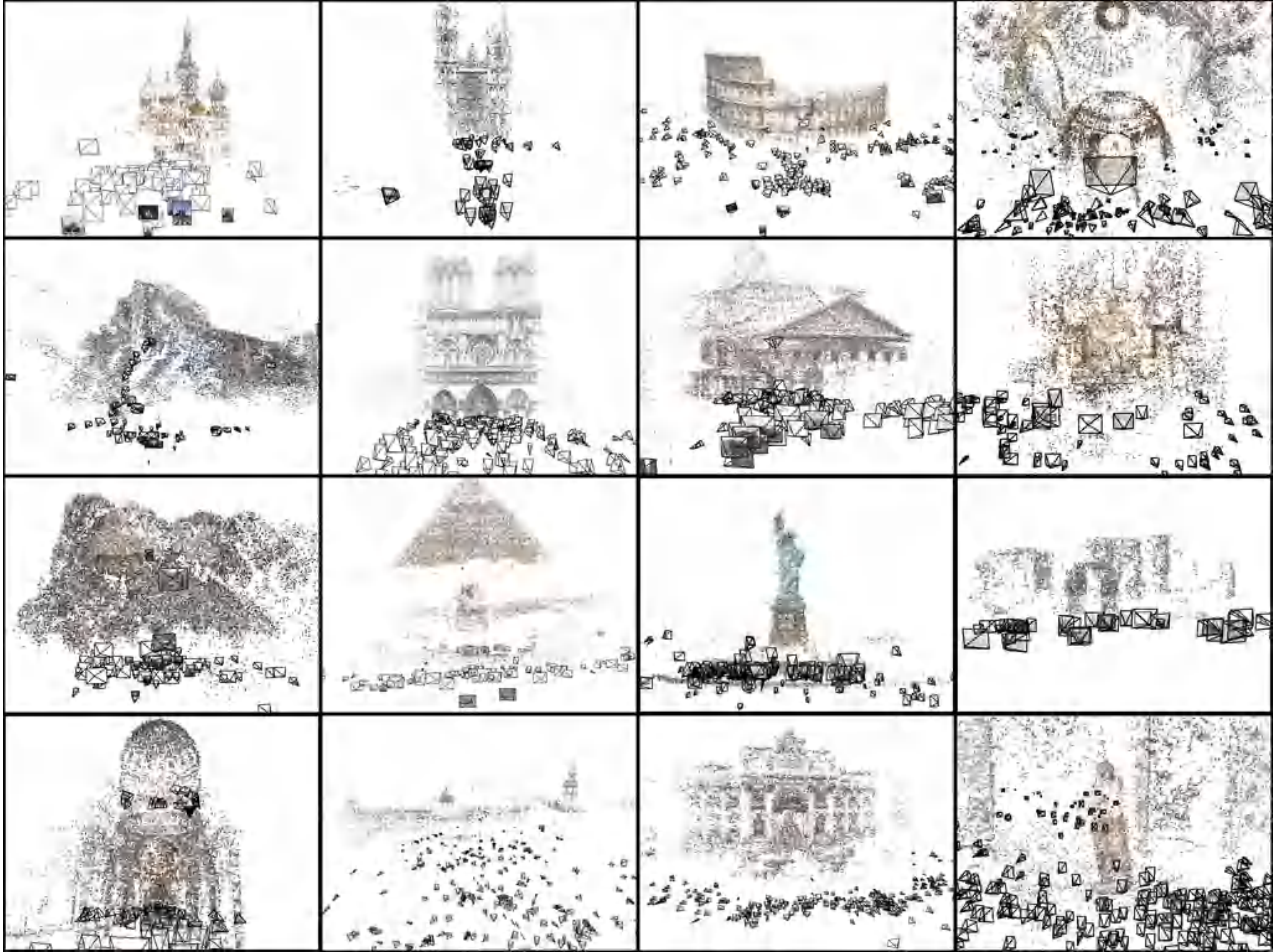
Incremental structure from motion

Incremental structure from motion

Time-lapse reconstruction of Dubrovnik, Croatia, viewed from above

Photo Explorer (Part of Noah Snavely's PhD work)

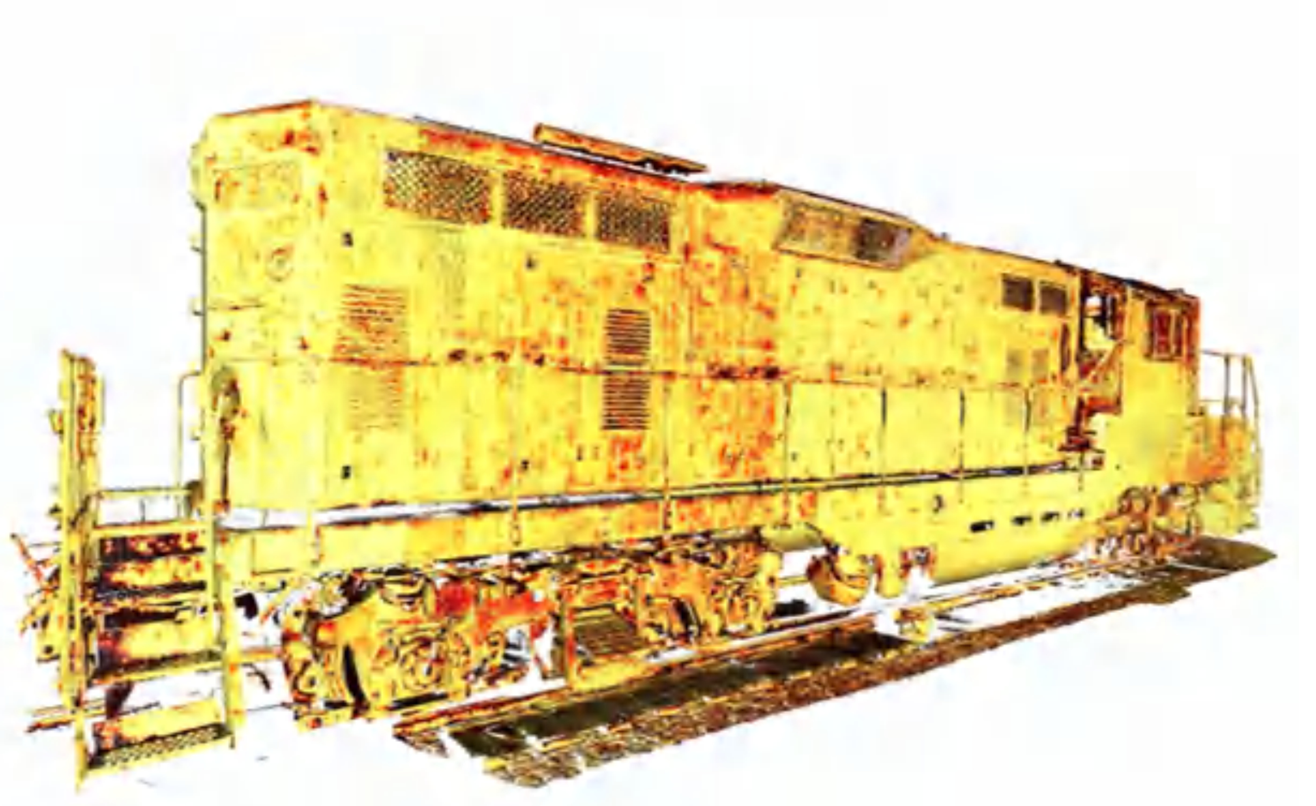




SfM vs MVS (Multi-view Stereo)

- MVS takes camera positions and images, produces dense point cloud (~one point per pixel) or full surface
- More similar to depth from stereo
- Often initialized with model from SfM

Image



Ground Truth (Laser)

MVS Reconstruction

How to try SfM Today: COLMAP

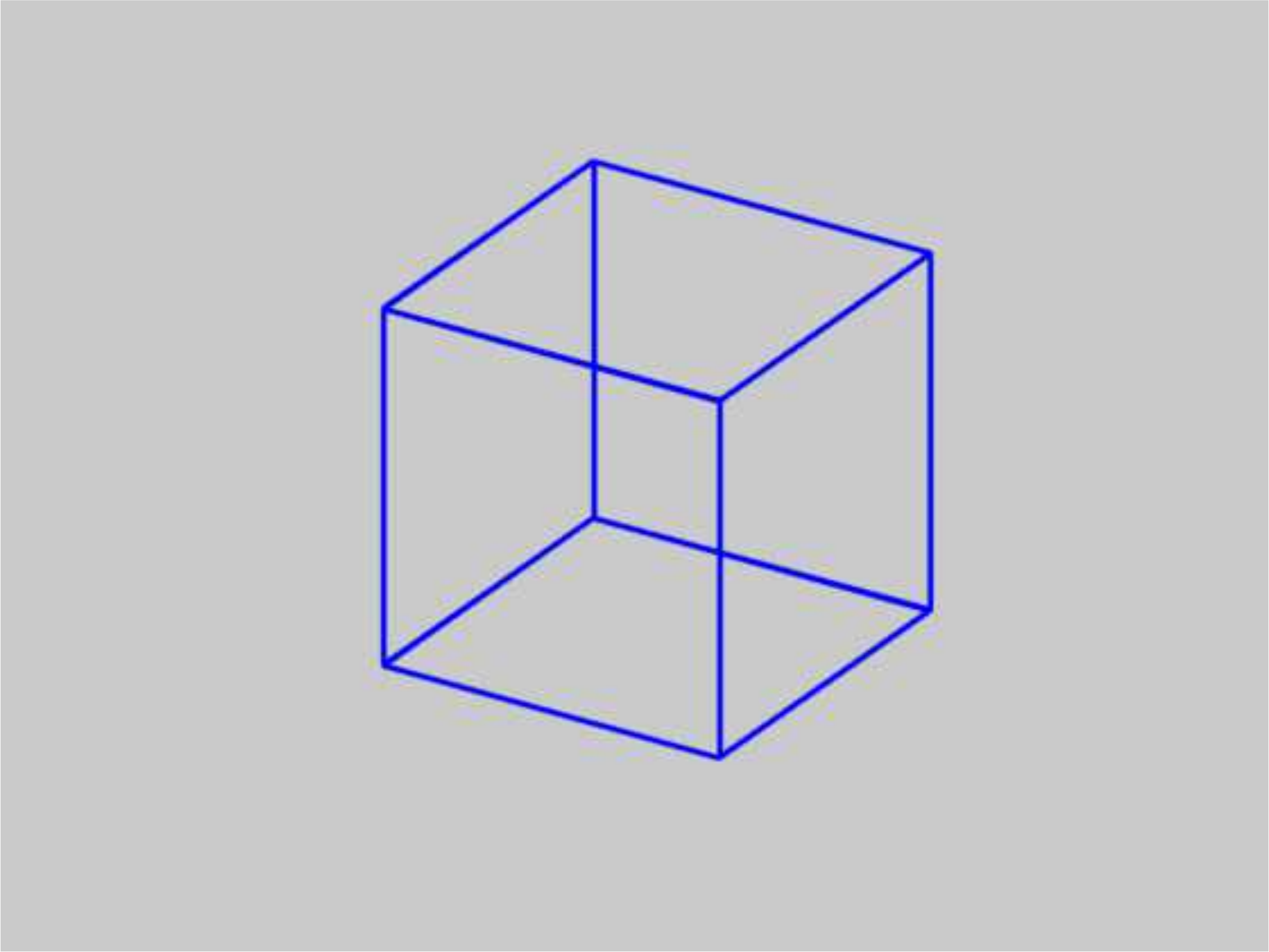
- COLMAP is an open source SfM (and MVS) implementation
- Written in C++
- Available on Github!

https://github.com/colmap/colmap

Questions?

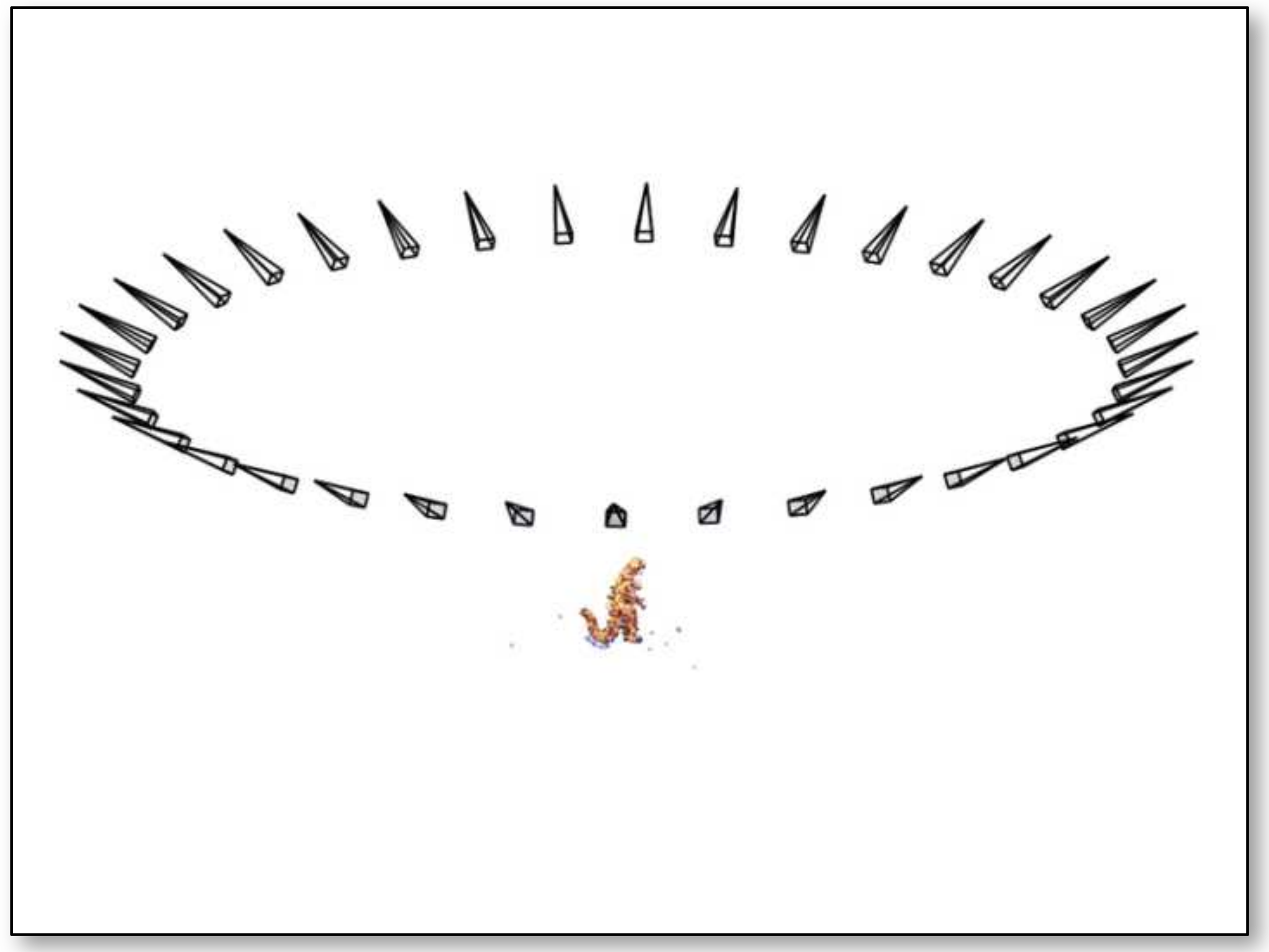
Is SfM always uniquely solvable?

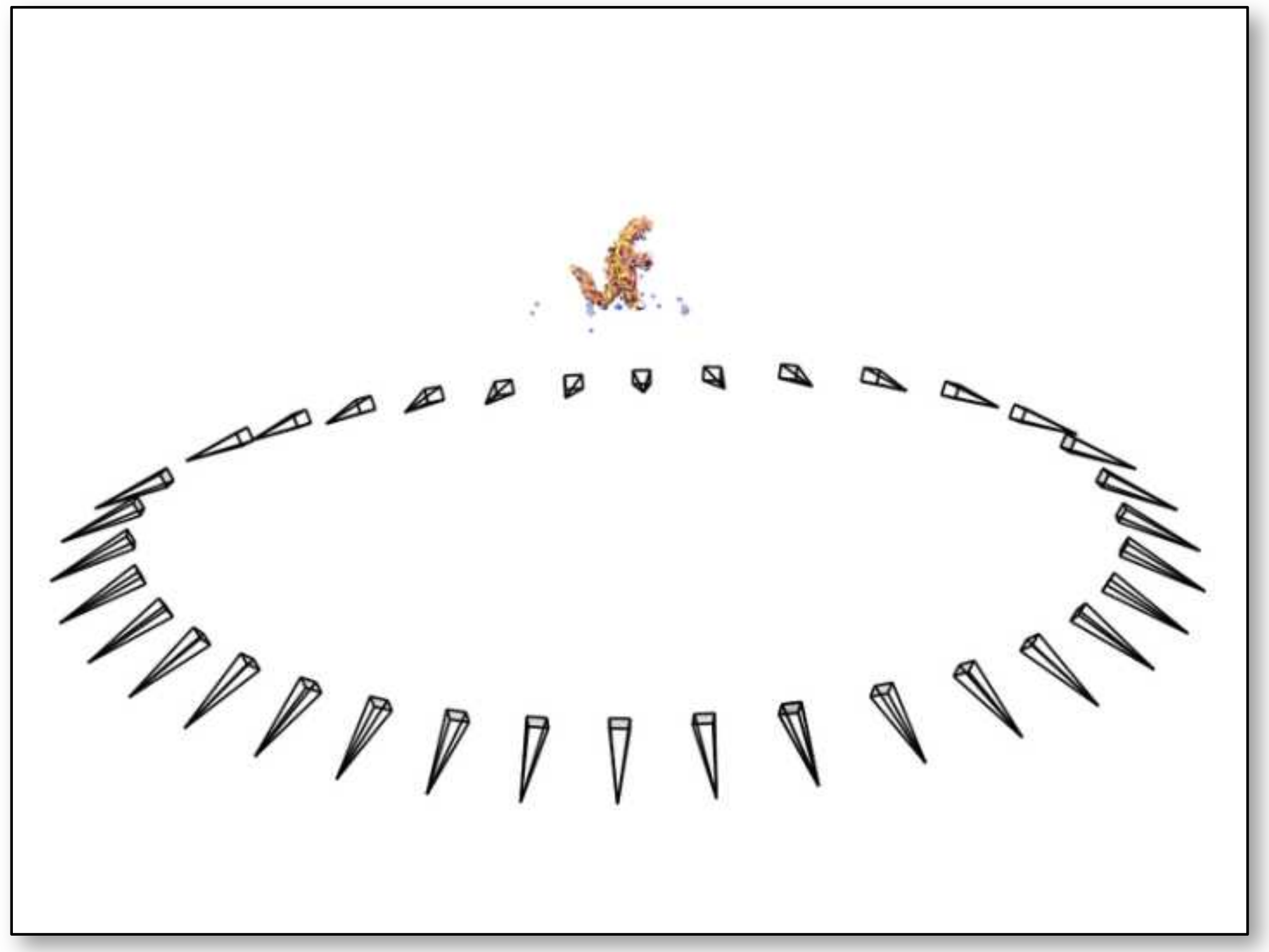
- No...



SfM – Failure cases

- Necker reversal





How these tools are used in Robotics

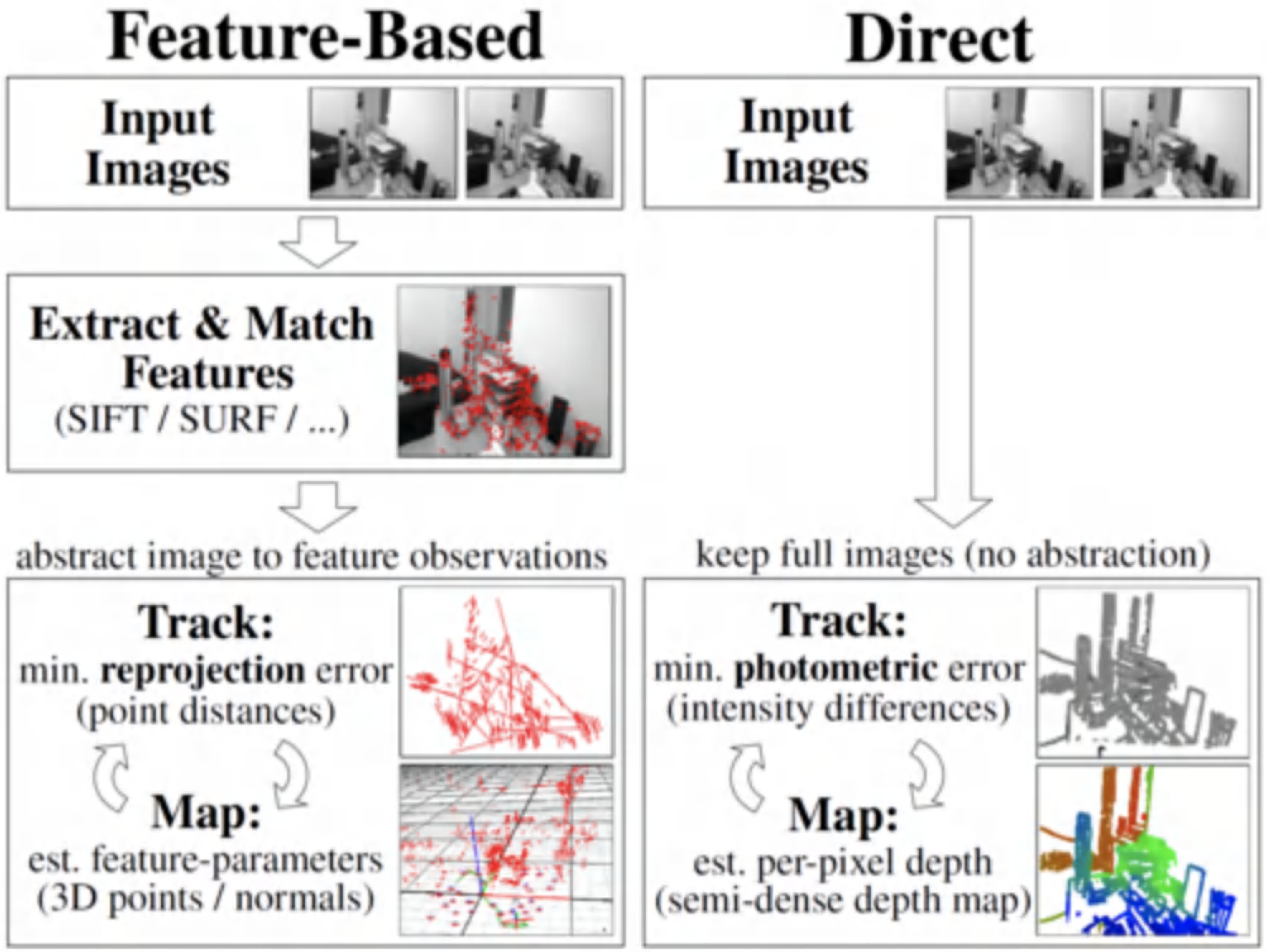
- In robotics, tools from SfM can be used for the “mapping” backend for SLAM
- Full bundle Adjustment is expensive—can the problem be reduced?
- With a map generated by SfM, how can a robot quickly localize on the map?

Basic SLAM (Similar to PTAM[1])

1. https://www.robots.ox.ac.uk/~gk/PTAM/

Keypoint Detection for SLAM

- SIFT is high quality but computationally expensive
- Other simpler and faster key point algorithms (FAST, BRISK, ORB, etc.) create binary representations of corners
- Some SLAM systems also use direct image matching which uses a photometric difference between warped images (or patches) for matching



https://vision.in.tum.de/research/vslam/lsdslam

Efficient Bundle Adjustment

- Not all frames are necessary—only need sufficient views of every map point
- Not all map points are necessary—points in dense regions can be eliminated
- Map quality can be maintained fairly well with a bundle adjustment of a local region, perhaps determined by connectedness

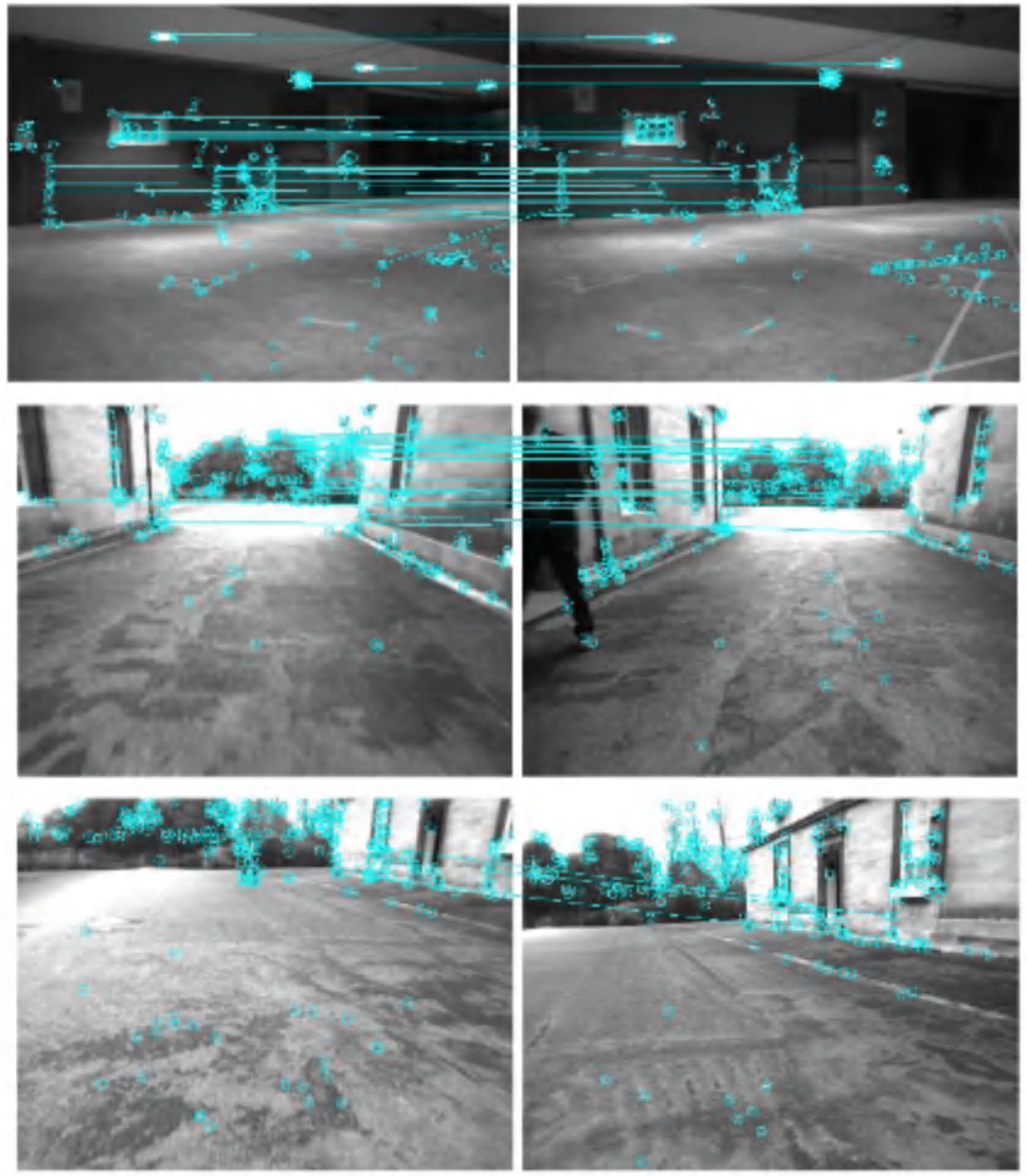
Image from: Skeletal graphs for efficient structure from motion

What if the robot is lost?

- Also called the “kidnapped robot problem”
- Will happen when robot moves too fast, no light, etc
- The robot must be able to quickly relocalize itself on existing map using only visual queues

Image-Based Localization

- With a quality map, it is often possible to recover location
- SLAM system maintains an additional queryable data structure that corresponds to the map or keyframes
- One such structure is a bag-of-words that represents the keyframes in the map based on their binary features—robot queries for closest frame then attempts to acquire pose
- Another method is to directly hash 3D points and query all possible matches, commonly used with a locality-sensitive hashing (LSH) table
- Both techniques typically use RANSAC and PnP for estimation



http://doriangalvez.com/papers/GalvezTRO12.pdf

Applications – Hyperlapse

https://www.youtube.com/watch?v=SOpwHaQnRSY

https://www.youtube.com/watch?v=sA4Za3Hv6ng

Applications – Photosynth

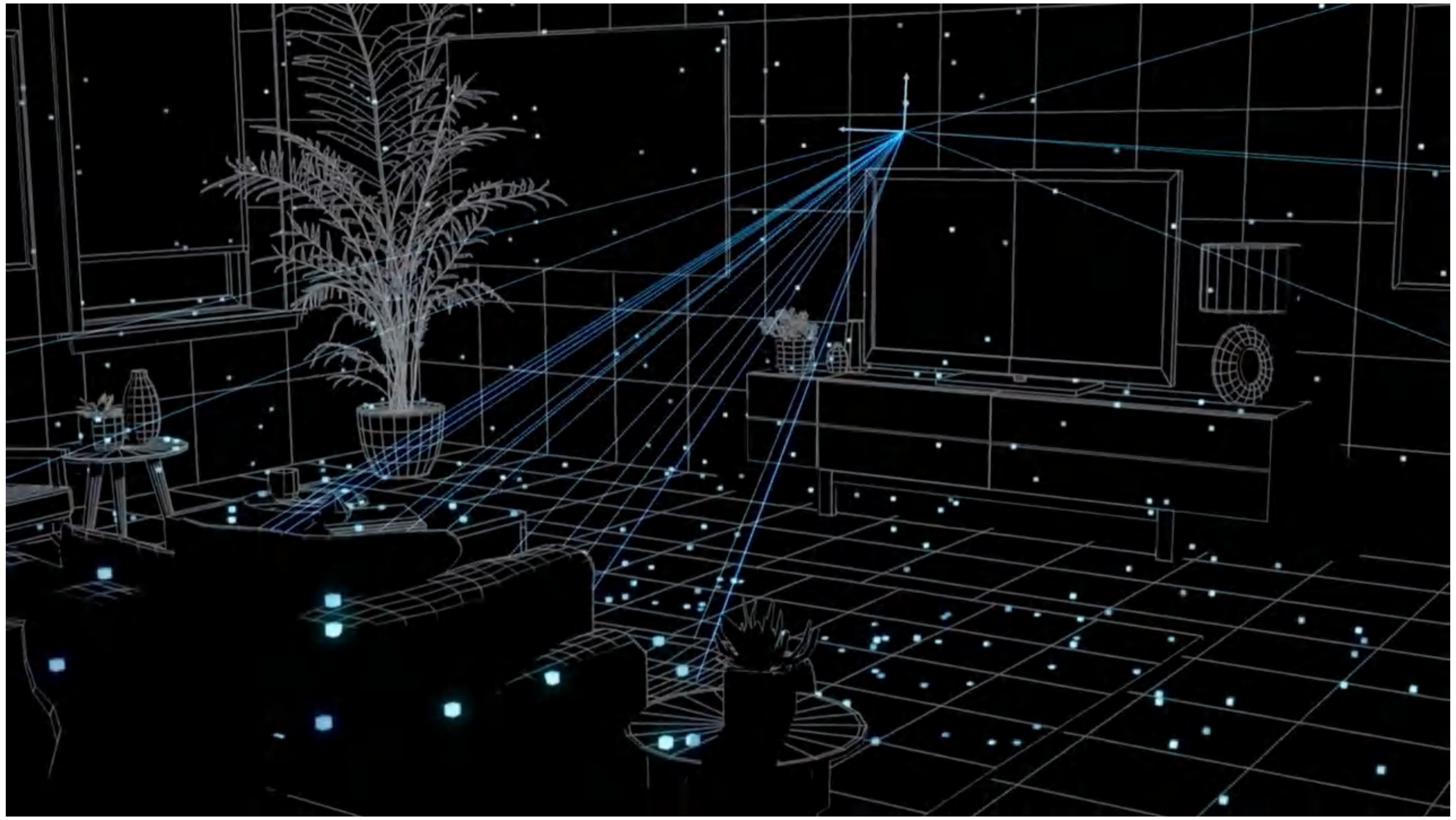
https://youtu.be/wB7HstiwcXc

Applications: Visual Reality & Augmented Reality

Hololens

https://www.youtube.com/watch?v=FMtvrTGnP04

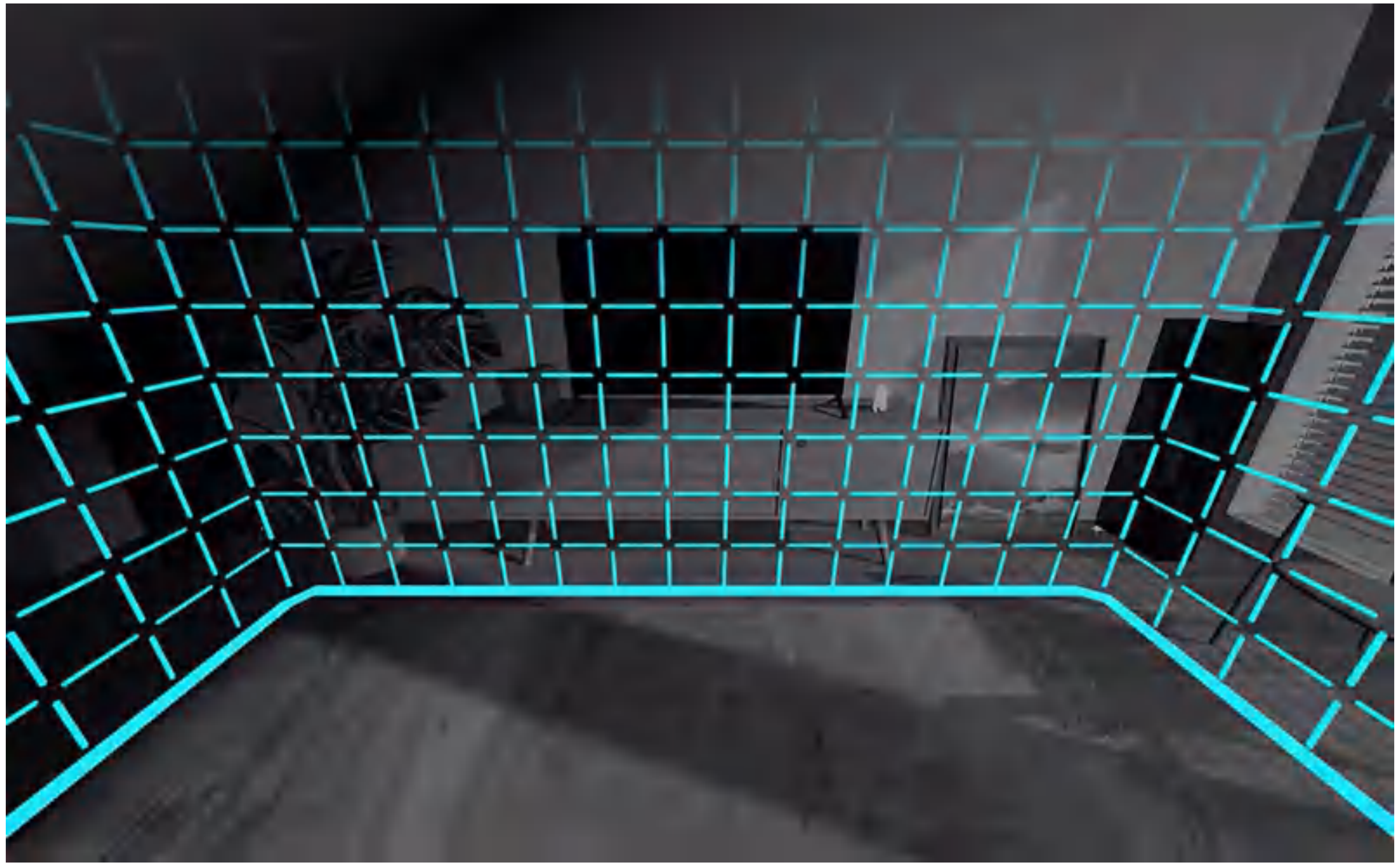
Applications: Visual Reality & Augmented Reality



Oculus

https://ai.facebook.com/blog/powered-by-ai-oculus-insight

Oculus Quest: SLAM-tracked, untethered VR



Oculus Quest: Arena Scale Demo

Applications: Self driving Cars

https://www.youtube.com/watch?v=ZR1yXFAslSk

Applications: Autonomous Drones

https://www.youtube.com/watch?v=imt2qZ7uw1s

Application: Highly Accurate 3D Maps

Scape: Building the ‘AR Cloud’: Part Three —3D Maps, the Digital Scaffolding of the 21st Century

https://medium.com/scape-technologies/building-the-ar-cloud-part-three-3d-maps-the-digital-scaffolding-of-the-21st-century-465fa55782dd

Application: AR walking directions

Questions?