Structure from Motion

3D Vision
University of Illinois

Derek Hoiem

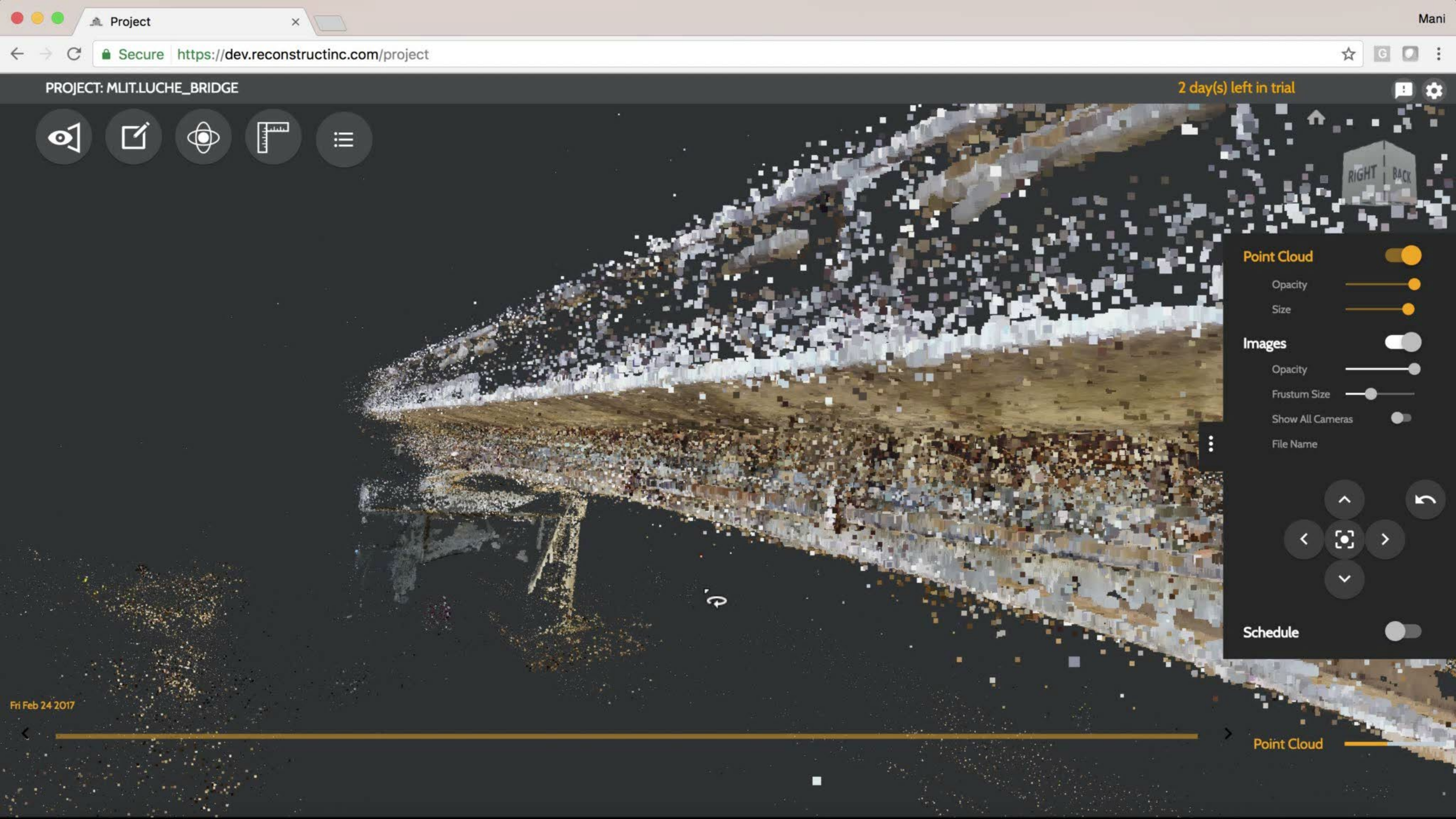
Structure from Motion (SfM)

Goal: Solve for camera poses and 3D points in scene

Example Application: Inspection

Enable inspection in hard to reach areas with drone photos and 3D reconstruction

- Create 3D model from images
- Provide tools to inspect on images and map interactions to 3D


Project
Mani
Secure https://dev.reconstructinc.com/project
PROJECT: MLIT.LUCHE_BRIDGE
2 day(s) left in trial
RIGHT
BACK
Point Cloud
Opacity
Size
Images
Opacity
Frustum Size
Show All Cameras
File Name
Schedule
Fri Feb 24 2017
Point Cloud

Incremental SfM

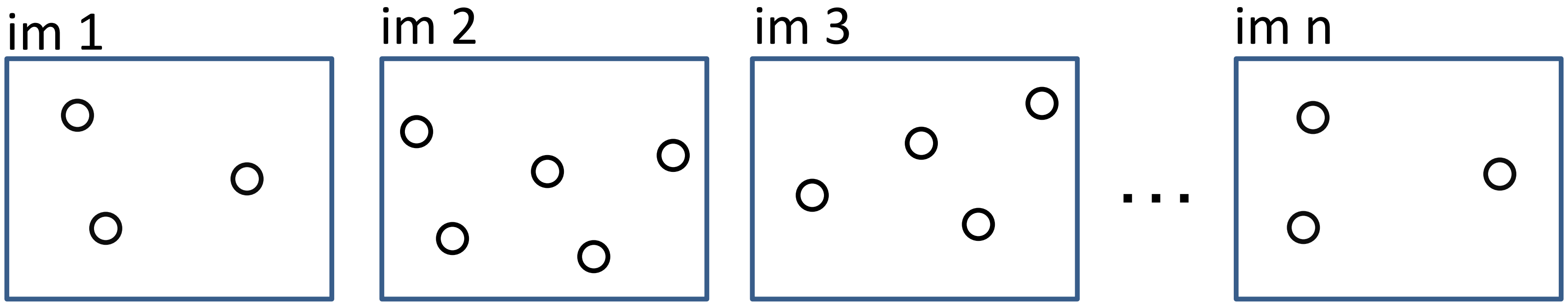
1. Compute features

2. Match images

3. Reconstruct
   a) Solve for poses and 3D points in two cameras
   b) Solve for pose of additional camera(s) that observe reconstructed 3D points
   c) Solve for new 3D points that are viewed in at least two cameras
   d) Bundle adjust to minimize reprojection error

Incremental SFM: **detect features**

- Feature types: SIFT, ORB, Hessian-Laplacian, …

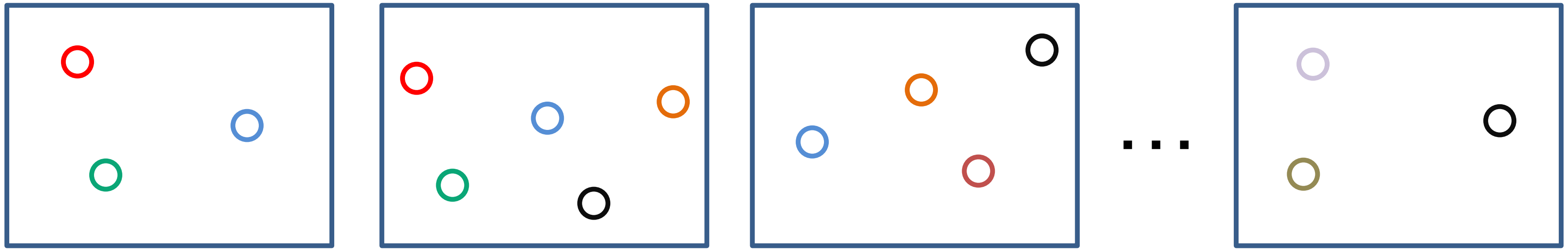


Each circle represents a set of detected features

Incremental SFM: **match features and images**

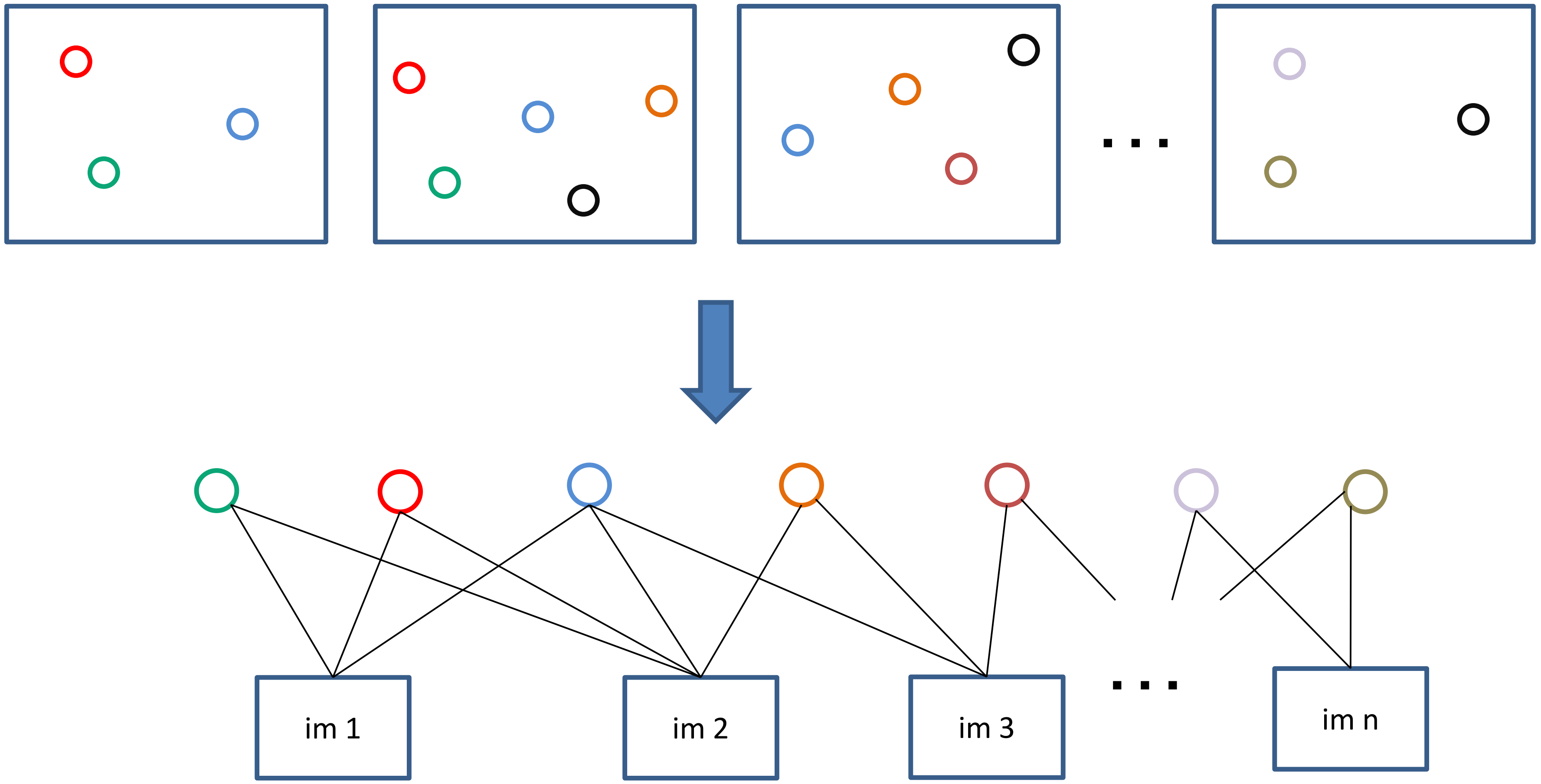
For each pair of images:

1. Match feature descriptors via approximate nearest neighbor
2. Solve for F or E and find inlier feature correspondences



Points of same color have been matched to each other

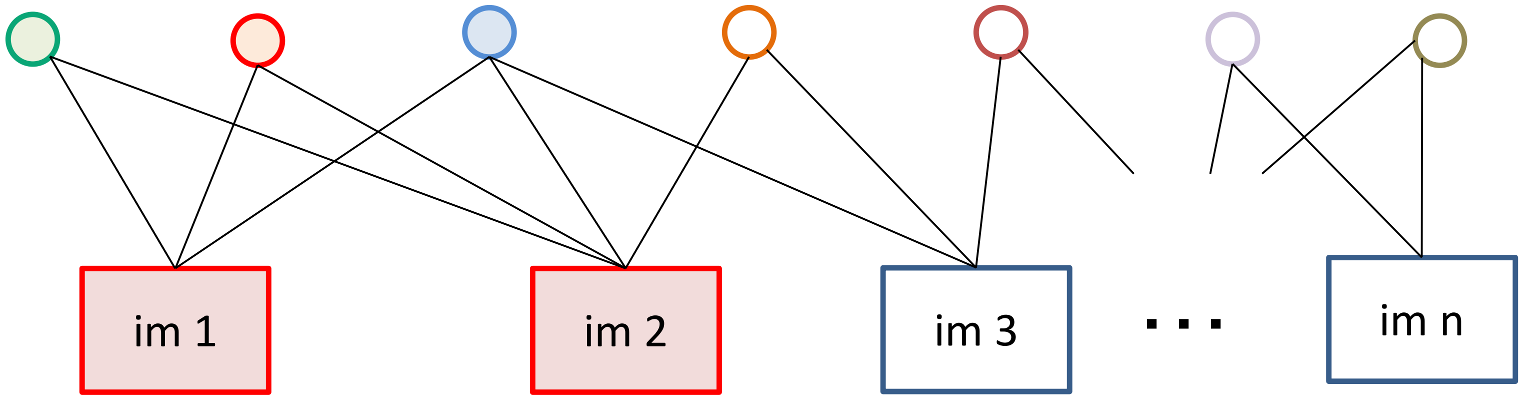
Incremental SFM: **create tracks graph**



tracks graph: bipartite graph between observed 3D points and images

Incremental SFM: **initialize reconstruction**

1. Choose two images that are likely to provide a stable estimate of relative pose
   - E.g., $\frac{\#\text{ inliers for } H}{\#\text{ inliers for } F} < 0.7$ and many inliers for F
2. Get focal lengths from EXIF, estimate essential matrix using 5-point algorithm, extract pose R_2, t_2 with $R_1 = \boldsymbol{I}, t_1 = \boldsymbol{0}$
3. Solve for 3D points given poses
4. Perform bundle adjustment to refine points and poses



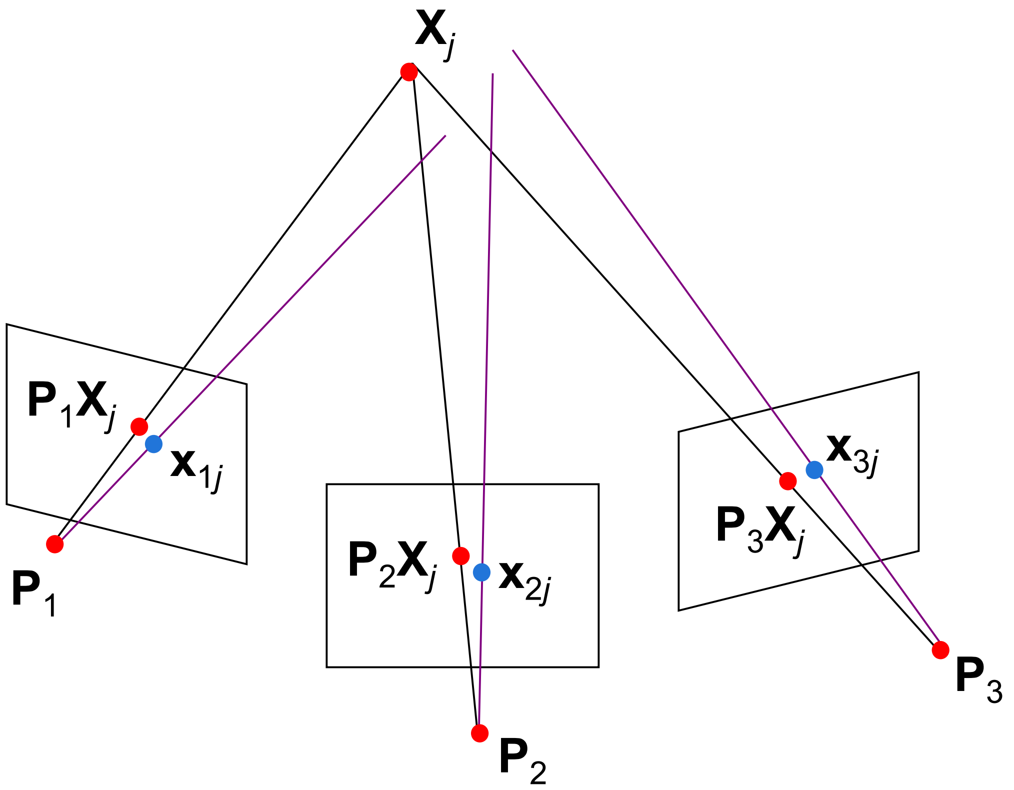
filled circles = "triangulated" points
filled rectangles = "resectioned" images (solved pose)

Bundle adjustment

- Non-linear method for refining structure and pose
- Minimizing reprojection error

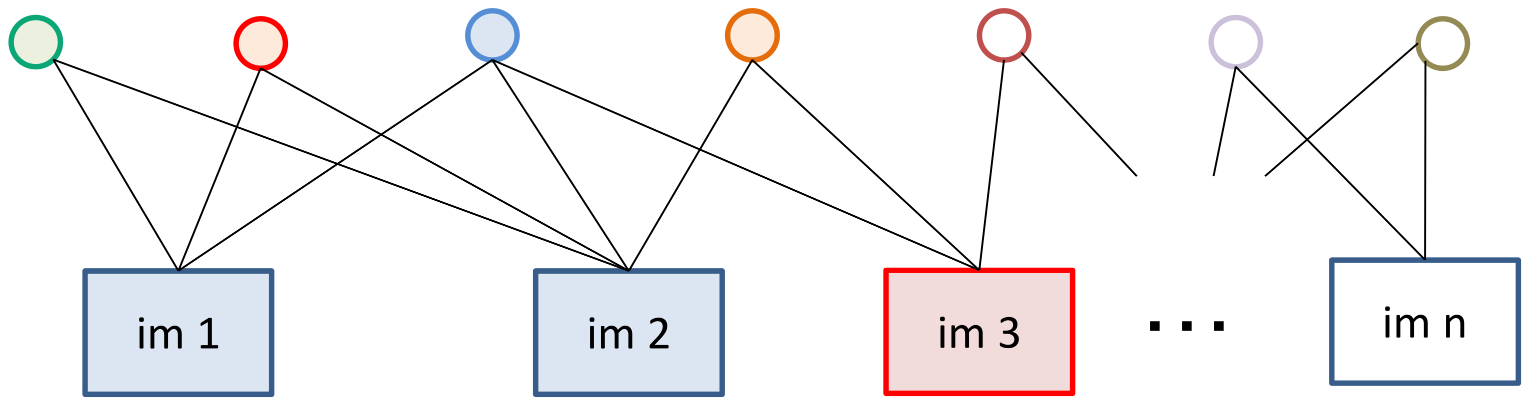
$$E(\mathbf{P}, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j\right)^2$$

Ceres Solver



Incremental SFM: **grow reconstruction**

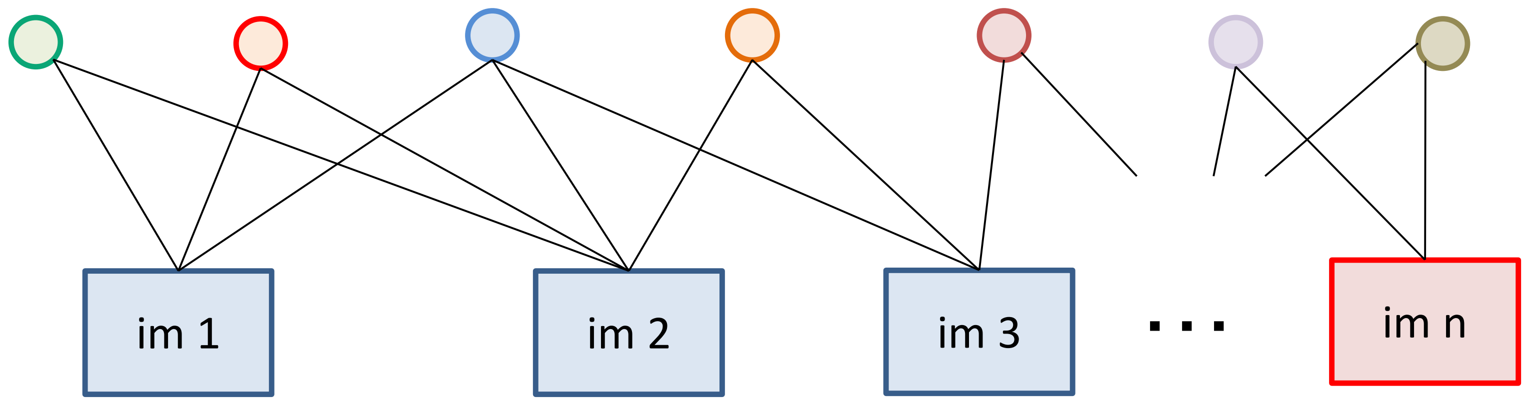
1. Resection: solve pose for image(s) that have the most triangulated points
2. Triangulate: solve for any new points that have at least two cameras
3. Bundle adjust
4. Optionally, align with GPS from EXIF or ground control points (GCP)



filled circles = "triangulated" points
filled rectangles = "resectioned" images (solved pose)

Incremental SFM: **grow reconstruction**

1. Resection: solve pose for image(s) that have the most triangulated points
2. Triangulate: solve for any new points that have at least two cameras
3. Bundle adjust
4. Optionally, align with GPS from EXIF or ground control points (GCP)



filled circles = “triangulated” points
filled rectangles = “resectioned” images (solved pose)

Why SfM is hard

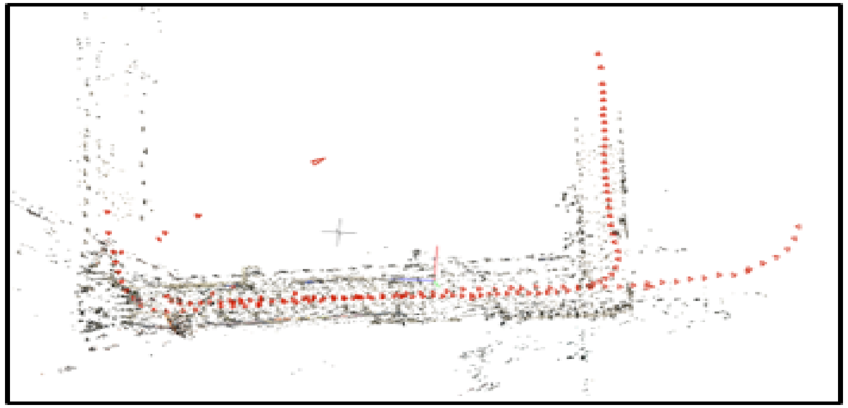
- Slow
  - Matching N^2 pairs of images takes too long (~1-4s per pair)
  - Bundle adjustment takes longer with more images and needs to be repeated as images are added: up to $O(N^3)$
  - Grow reconstruction phase is not easy to parallelize

- Bad feature matches are very common and cause misregistrations

- Insufficient feature matches cause incomplete reconstructions

	# Images	# Registered				Time [s]			
		Theia	*Bundler*	*VSFM*	*Ours*	*Theia*	*Bundler*	*VSFM*	*Ours*
Rome [14]	74,394	–	13,455	14,797	20,918	–	295,200	6,012	10,912
Quad [14]	6,514	–	5,028	5,624	5,860	–	223,200	2,124	3,791
Dubrovnik [36]	6,044	–	–	–	5,913	–	–	–	3,821
Alamo [61]	2,915	582	647	609	666	874	22,025	495	882
Ellis Island [61]	2,587	231	286	297	315	94	12,798	240	332
Gendarmenmarkt [61]	1,463	703	302	807	861	202	465,213	412	627
Madrid Metropolis [61]	1,344	351	330	309	368	95	21,633	203	251
Montreal Notre Dame [61]	2,298	464	501	491	506	207	112,171	418	723
NYC Library [61]	2,550	339	400	411	453	194	36,462	327	420
Piazza del Popolo [61]	2,251	335	376	403	437	89	33,805	275	380
Piccadilly [61]	7,351	2,270	1,087	2,161	2,336	1,427	478,956	1,236	1,961
Roman Forum [61]	2,364	1,074	885	1,320	1,409	1,302	587,451	748	1,041
Tower of London [61]	1,576	468	569	547	578	201	184,905	497	678
Trafalgar [61]	15,685	5,067	1,257	5,087	5,211	1,494	612,452	3,921	5,122
Union Square [61]	5,961	720	649	658	763	131	56,317	556	693
Vienna Cathedral [61]	6,288	858	853	890	933	764	567,213	899	1,244
Yorkminster [61]	3,368	429	379	427	456	164	34,641	661	997

from COLMAP SfM (Schonberger et al. 2016)



Bad matches in low texture, repetitive hallway cause COLMAP to fail to reconstruct loop (Kataria et al. 2020)

Incremental SfM, Take 2: improvements in green

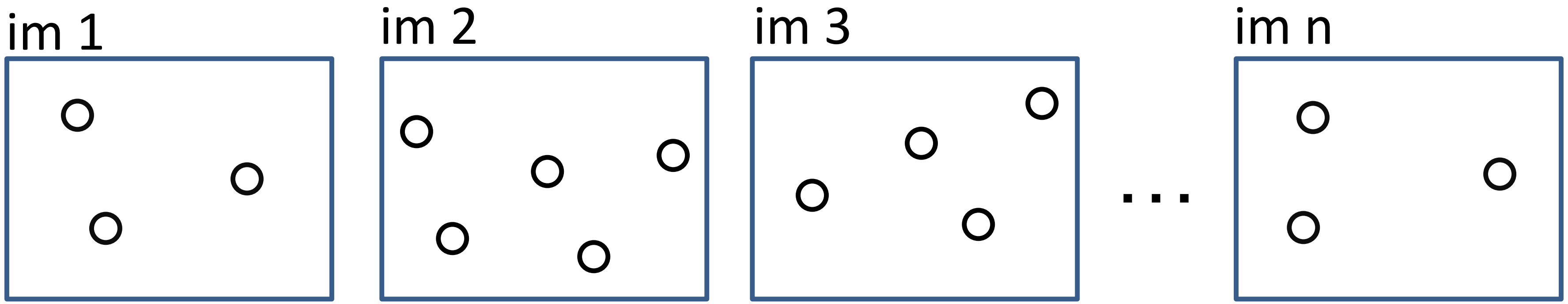
1. Compute features

2. Match images

3. Reconstruct
   a) Solve for poses and 3D points in two cameras
   b) Solve for pose of additional camera(s) that observe reconstructed 3D points
   c) Solve for new 3D points that are viewed in at least two cameras
   d) Bundle adjust to minimize reprojection error

Incremental SFM: **detect features**

- Feature types: SIFT, ORB, Hessian-Laplacian, …
- Use GPU for fast feature computation



Each circle represents a set of detected features

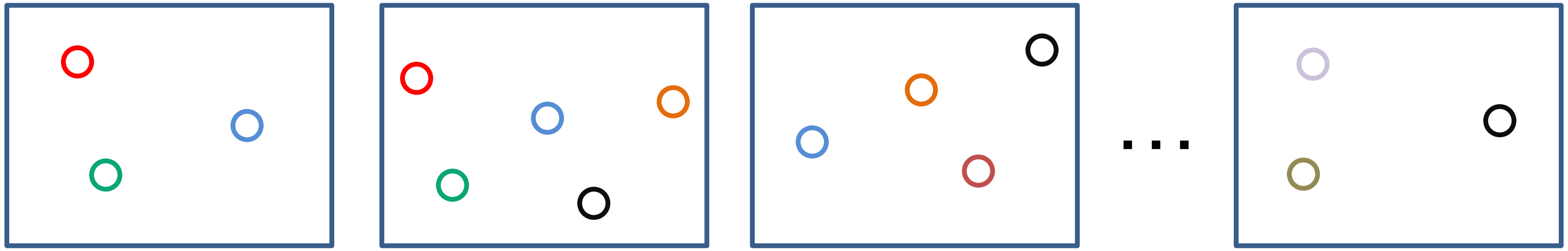
Incremental SFM: **match features and images**

Find match candidates:

- Match K closest images in GPS distance or time
- Use vocab tree on features to find K most similar images
- Potentially, add new candidates based on candidates that are already found

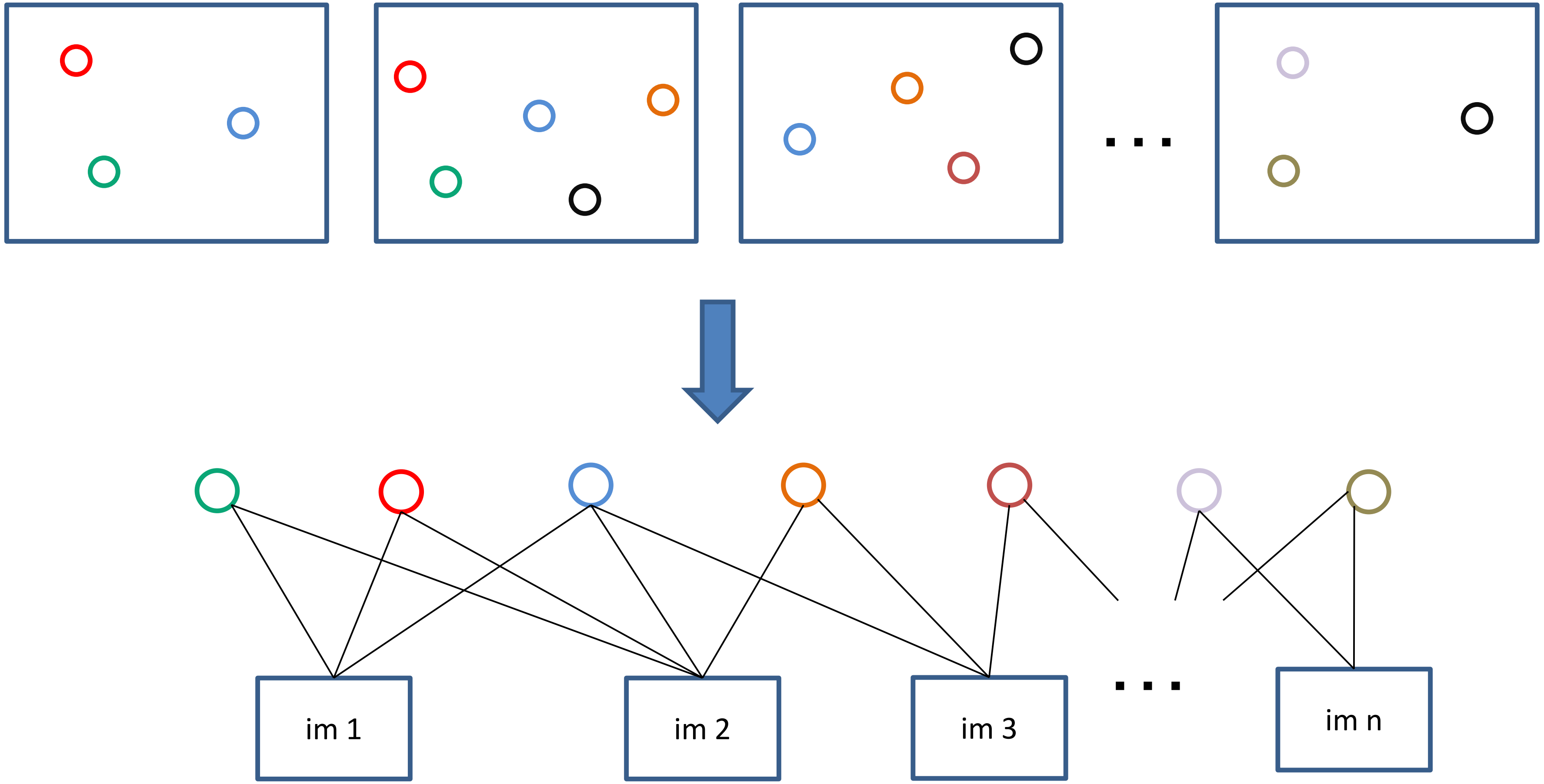
For each pair of candidate images:

1. Match feature descriptors via approximate nearest neighbor
   - GPU can be used for fast feature matching
   - Lowe's ratio test used to reject some potentially bad matches
2. Solve for F or E and find inlier feature correspondences
   - Remove feature matches that have above threshold reprojection error according to F or E
   - Discard image pairs that have below threshold number of geometrically verified matches



Points of same color have been matched to each other

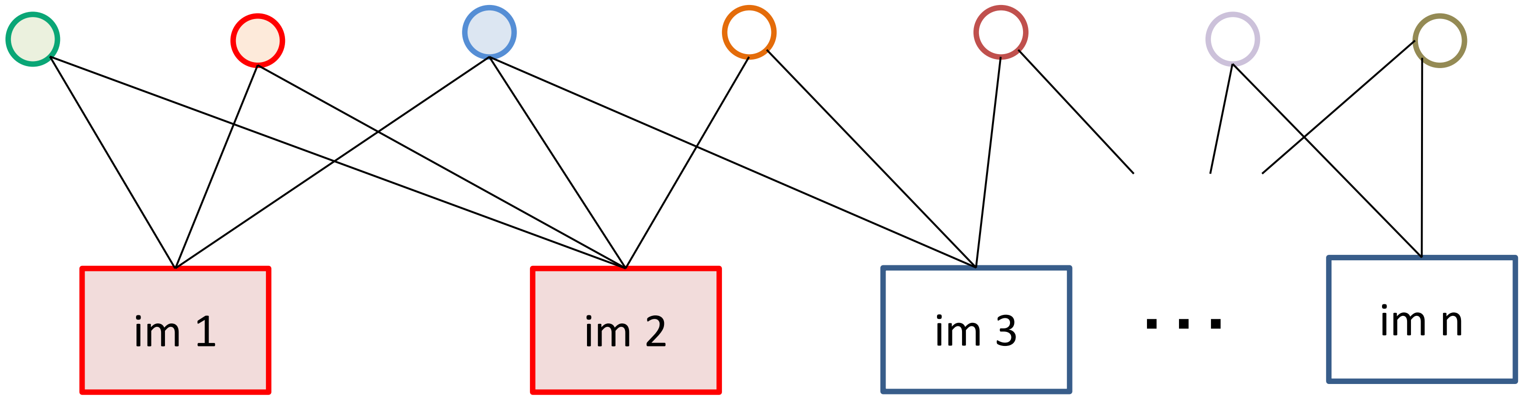
Incremental SFM: **create tracks graph**



tracks graph: bipartite graph between observed 3D points and images

Incremental SFM: **initialize reconstruction**

1. Choose two images that are likely to provide a stable estimate of relative pose
   - E.g., $\frac{\#\text{ inliers for } H}{\#\text{ inliers for } F} < 0.7$ and many inliers for F
2. Get focal lengths from EXIF, estimate essential matrix using [5-point algorithm](), extract pose R_2, t_2 with $R_1 = \boldsymbol{I}, t_1 = \boldsymbol{0}$
3. Solve for 3D points given poses
4. Perform bundle adjustment to refine points and poses



filled circles = “triangulated” points
filled rectangles = “resectioned” images (solved pose)

Triangulation: Linear Solution

$$\mathbf{x} = w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \mathbf{x}' = w\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

Given **P**, **P'**, **x**, **x'**

1. Precondition points and projection matrices
2. Create matrix **A**
3. [U, S, V] = svd(A)
4. **X** = V(:, end)

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \\ \mathbf{p}_3^T \end{bmatrix} \quad \mathbf{P}' = \begin{bmatrix} \mathbf{p}_1'^T \\ \mathbf{p}_2'^T \\ \mathbf{p}_3'^T \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} u\mathbf{p}_3^T - \mathbf{p}_1^T \\ v\mathbf{p}_3^T - \mathbf{p}_2^T \\ u'\mathbf{p}_3'^T - \mathbf{p}_1'^T \\ v'\mathbf{p}_3'^T - \mathbf{p}_2'^T \end{bmatrix}$$

Pros and Cons

- Works for any number of corresponding images
- Not projectively invariant

Code: http://www.robots.ox.ac.uk/~vgg/hzbook/code/vgg_multiview/vgg_X_from_xP_lin.m

Triangulation: Non-linear Solution

- Minimize projected error while satisfying $\hat{\boldsymbol{x}}'^T \boldsymbol{F} \hat{\boldsymbol{x}} = 0$

$$cost(\boldsymbol{X}) = dist(\boldsymbol{x}, \hat{\boldsymbol{x}})^2 + dist(\boldsymbol{x}', \hat{\boldsymbol{x}}')^2$$

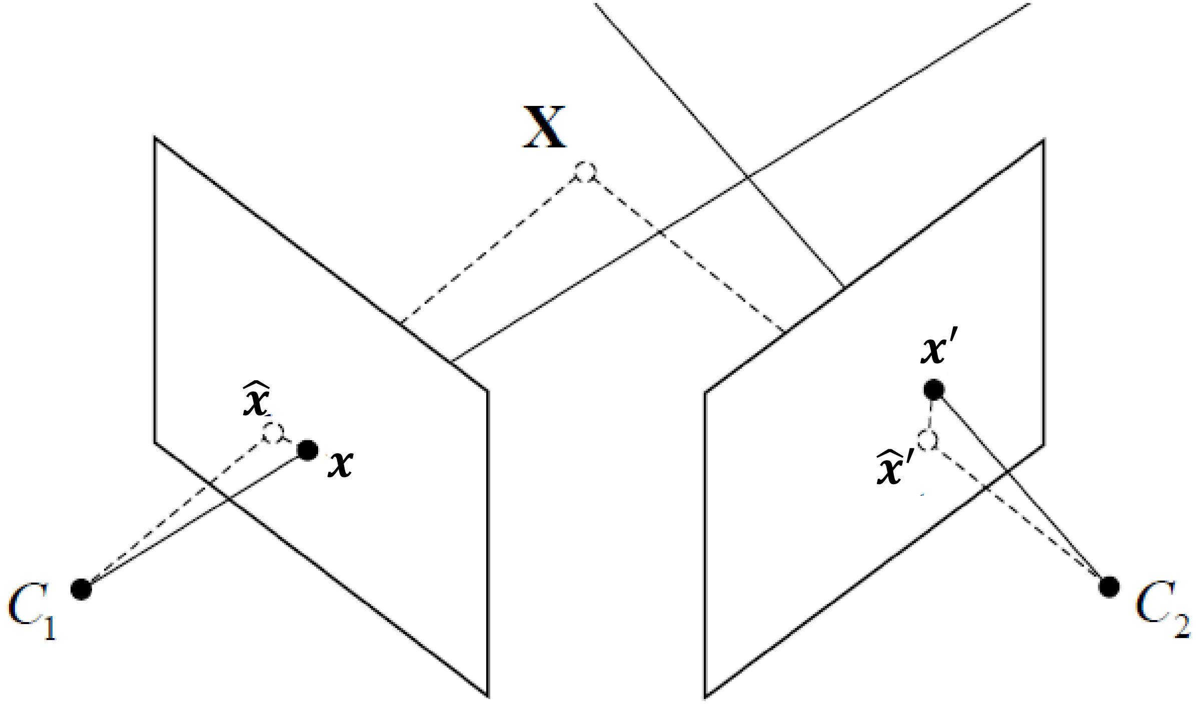


Figure source: Robertson and Cipolla (Chpt 13 of Practical Image Processing and Computer Vision)

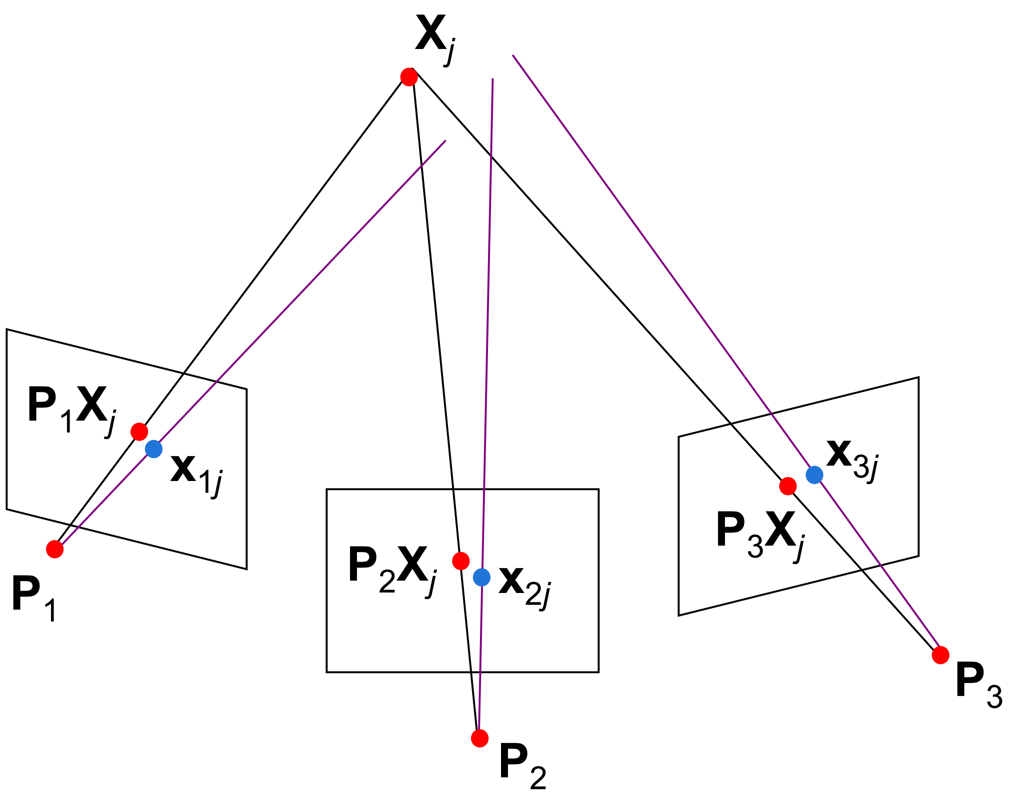
Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P}, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i \mathbf{X}_j\right)^2$$

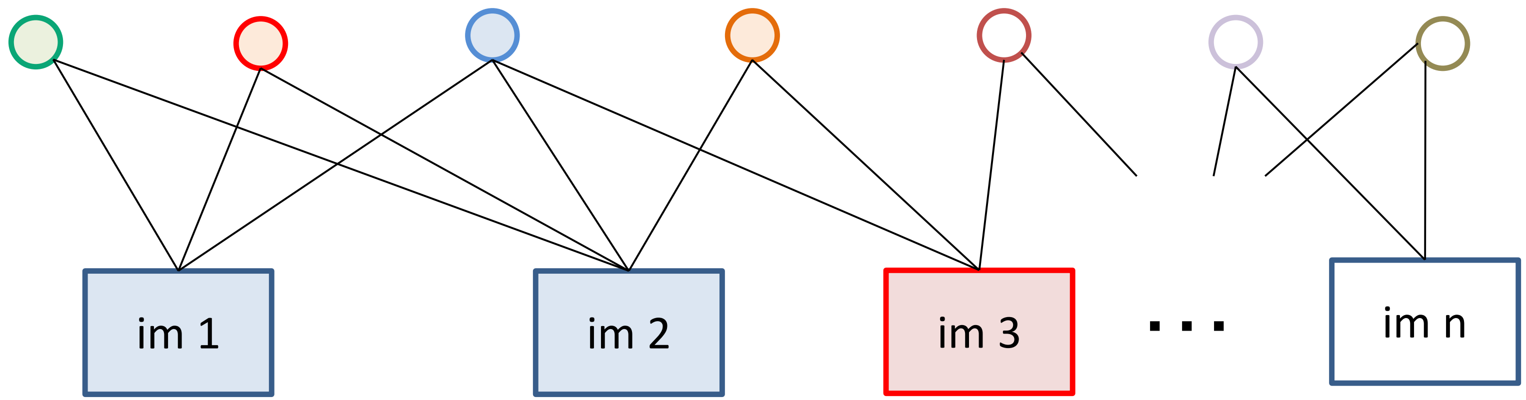
Ceres Solver

Use robust loss for reprojection error, such as Huber



Incremental SFM: **grow reconstruction**

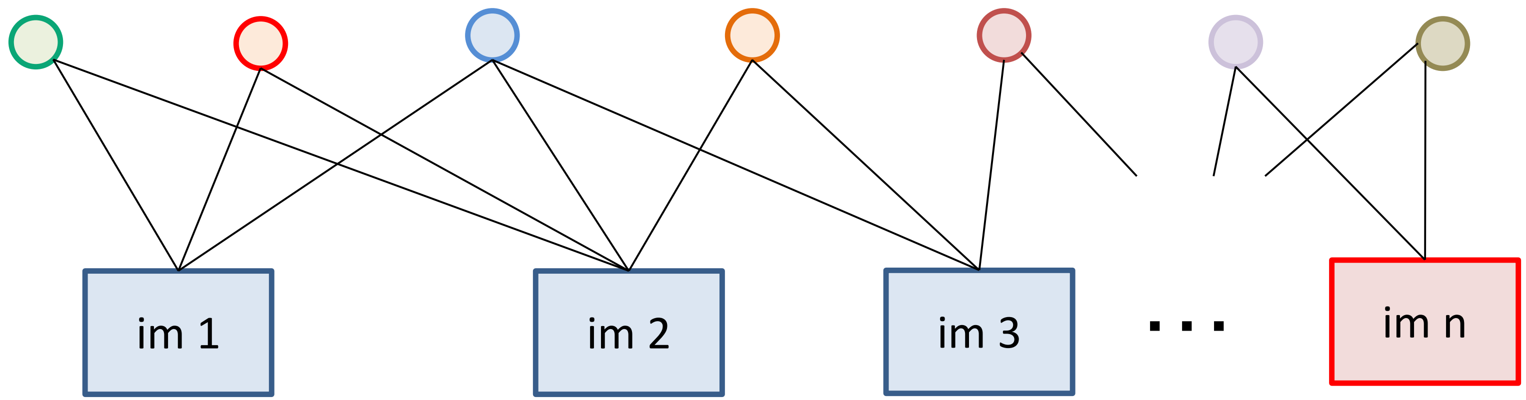
1. Sort images, e.g. by number of triangulated points
    a. Resection: solve pose for image(s) that have the most triangulated points
    b. Triangulate: solve for any new points viewed by at least two reconstructed cameras
    c. Remove 3D points that do not have enough baseline or too high reprojection error in any camera (optionally, split into multiple tracks)
    d. Bundle adjust
        - Only do full bundle adjust after some percent of new images are resectioned (huge time savings for large reconstructions)
2. Optionally, align with GPS from EXIF or ground control points (GCP)



filled circles = “triangulated” points
filled rectangles = “resectioned” images (solved pose)

Incremental SFM: **grow reconstruction**

1. Sort images, e.g. by number of triangulated points
   a. Resection: solve pose for image(s) that have the most triangulated points
   b. Triangulate: solve for any new points viewed by at least two reconstructed cameras
   c. Remove 3D points that do not have enough baseline or too high reprojection error in any camera (optionally, split into multiple tracks)
   d. Bundle adjust
      - Only do full bundle adjust after some percent of new images are resectioned (huge time savings for large reconstructions)
2. Optionally, align with GPS from EXIF or ground control points (GCP)



filled circles = “triangulated” points
filled rectangles = “resectioned” images (solved pose)

Improving Structure from Motion with Reliable Resectioning

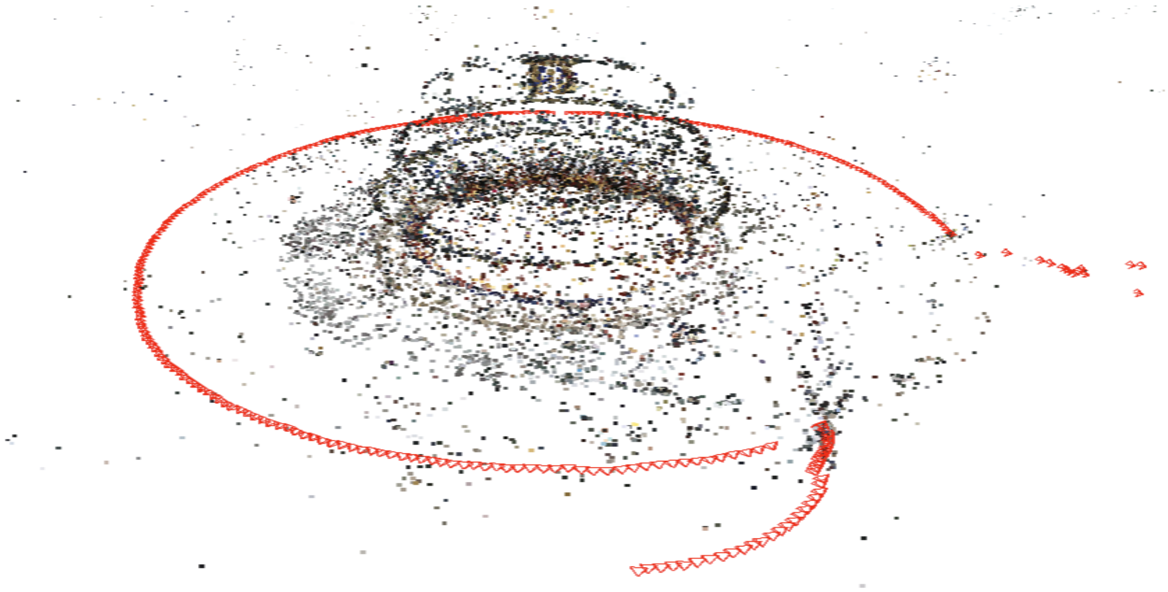
Rajbir Kataria
University of Illinois
Urbana-Champaign

Joseph DeGol
Microsoft

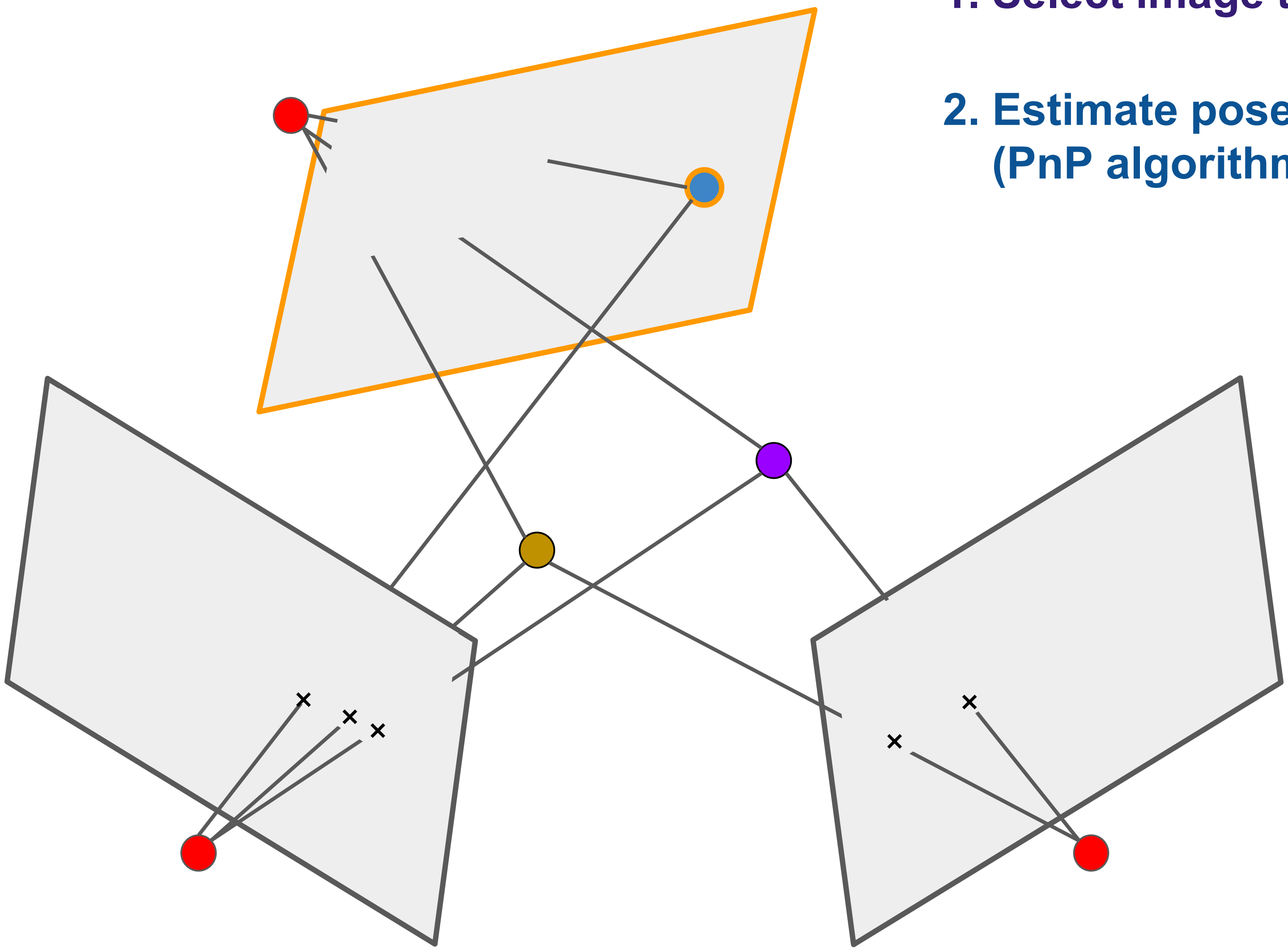
Derek Hoiem
University of Illinois
Urbana-Champaign

3DV 2000

False matches on repeated structures cause catastrophic failures



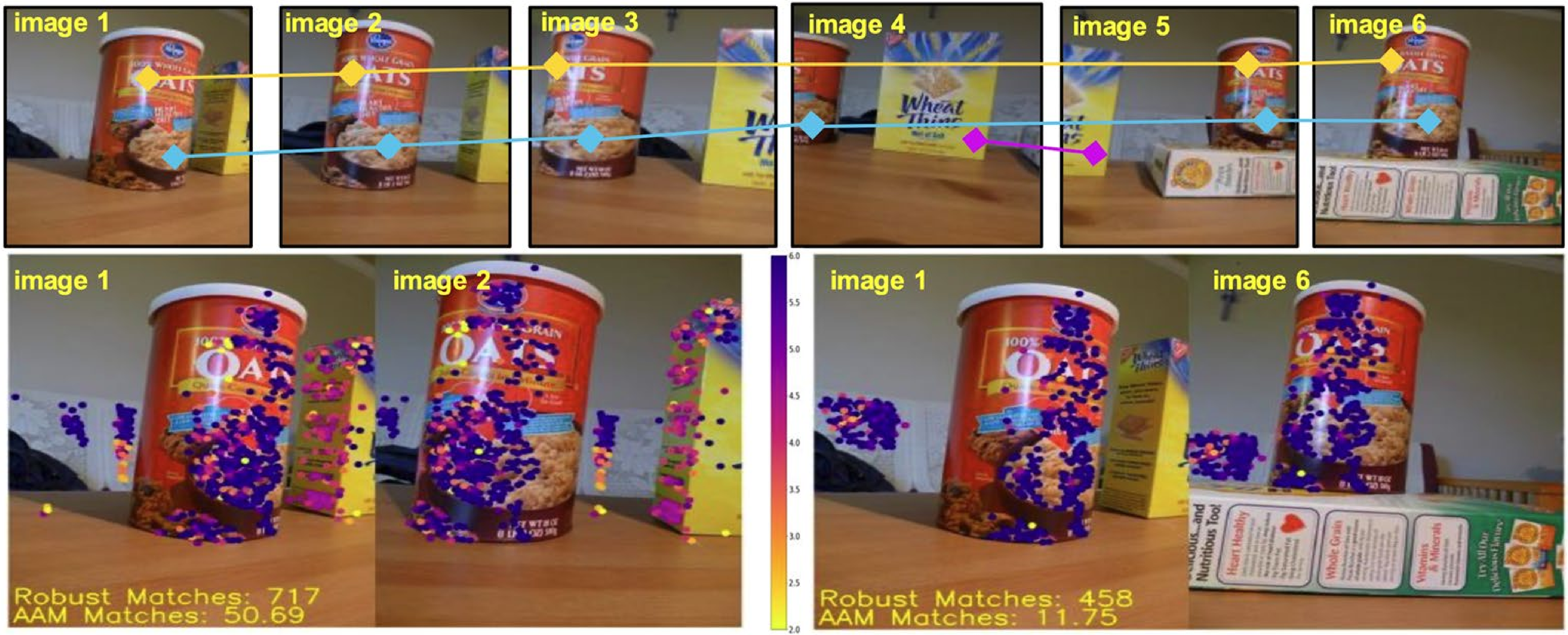
Resectioning is a critical step



1. **Select image that views the most triangulated points**

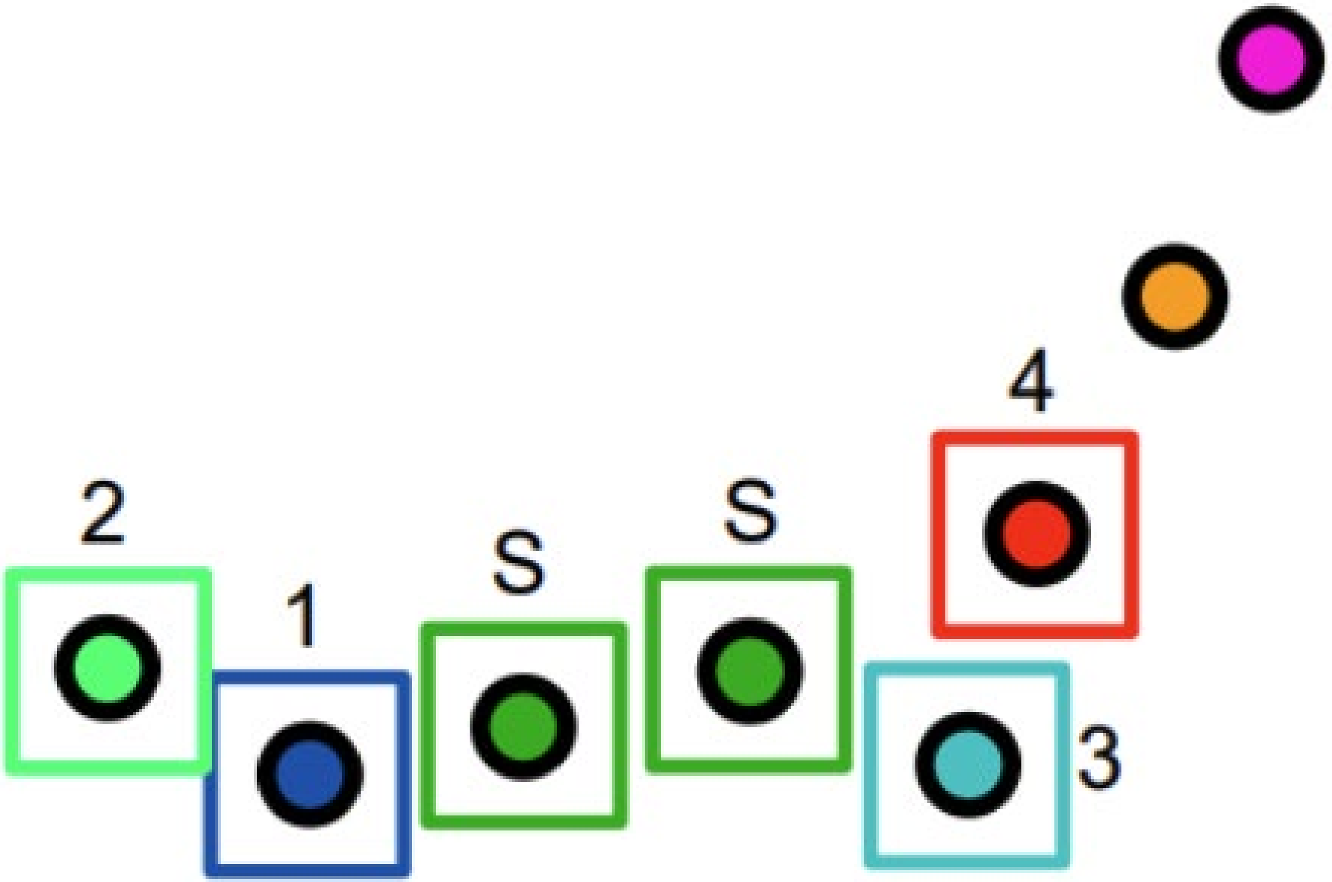
2. **Estimate pose of image using all the triangulated points (PnP algorithm using RANSAC)**

Ambiguity-adjusted match score (AAM): Discount longer tracks that are more likely to correspond to duplicate structures



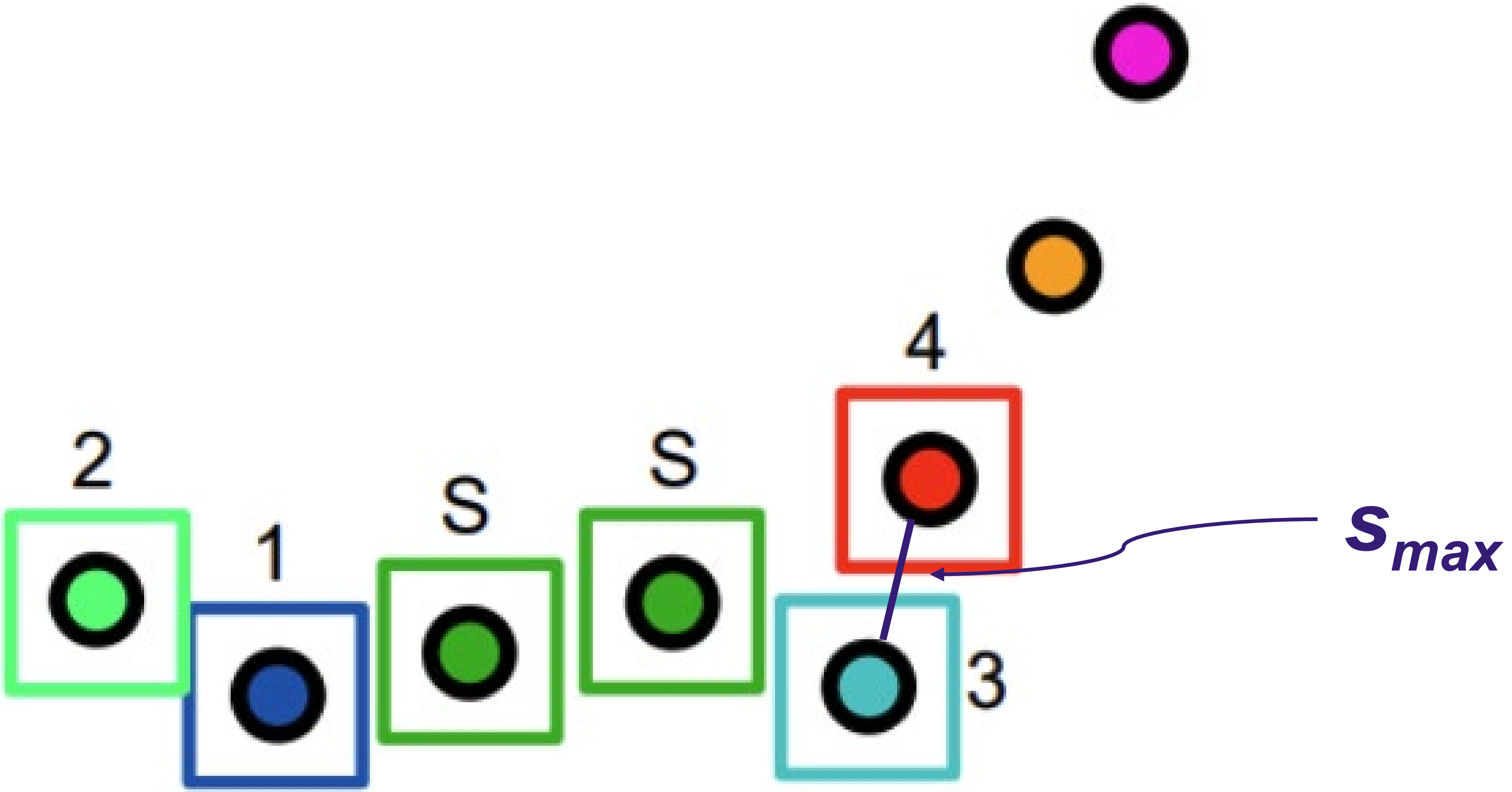
Local resectioning order uses most similar image

We use points from a smaller set of reliable images to determine **resectioning order** and **pose estimation**



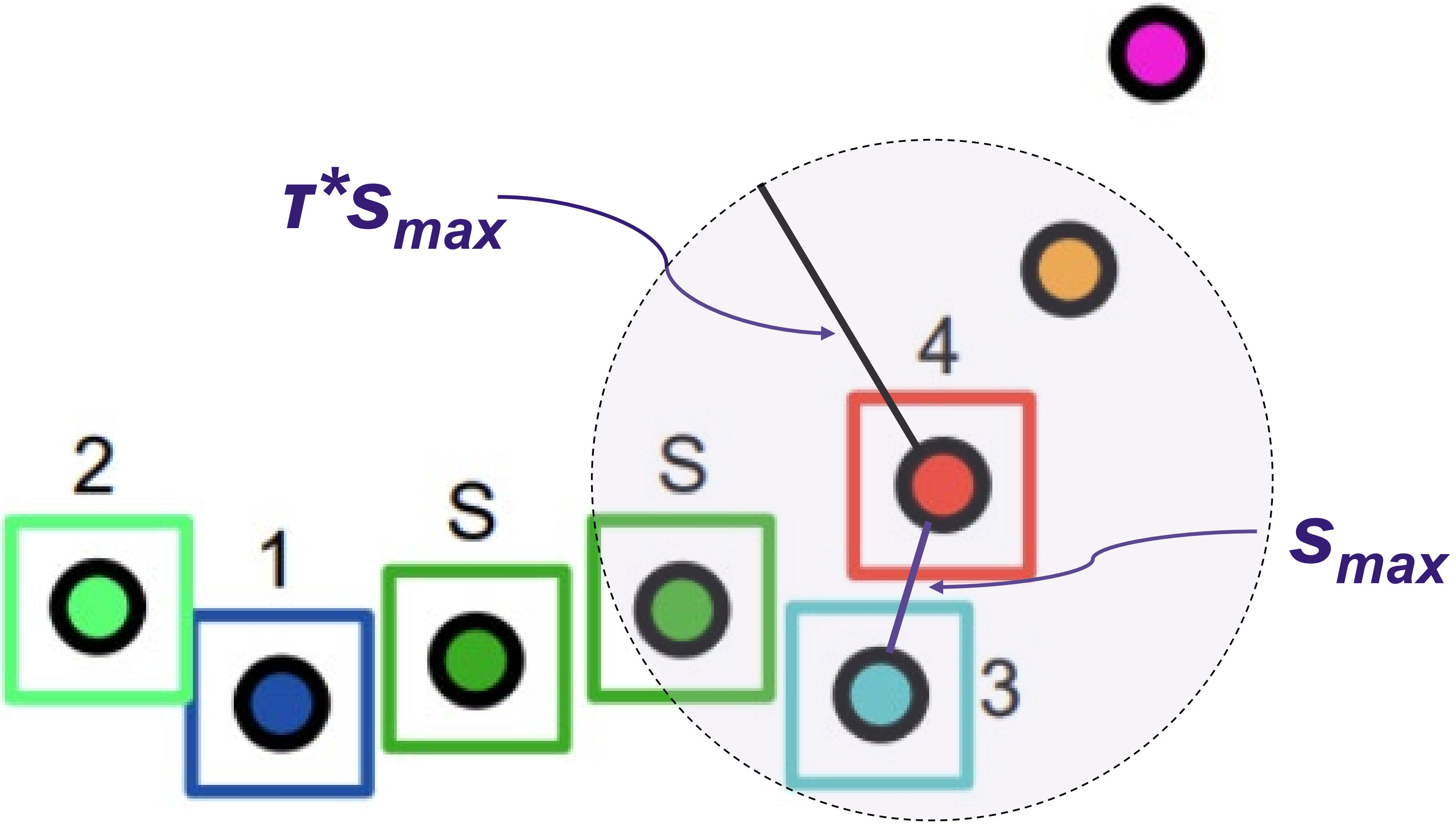
Local pose estimation uses reliable images only

We use points from a smaller set of reliable images to determine **resectioning order** and **pose estimation**



Local pose estimation uses reliable images only

Use points from a smaller set of reliable images to determine **resectioning order** and **pose estimation**



Our method improves standard pipelines

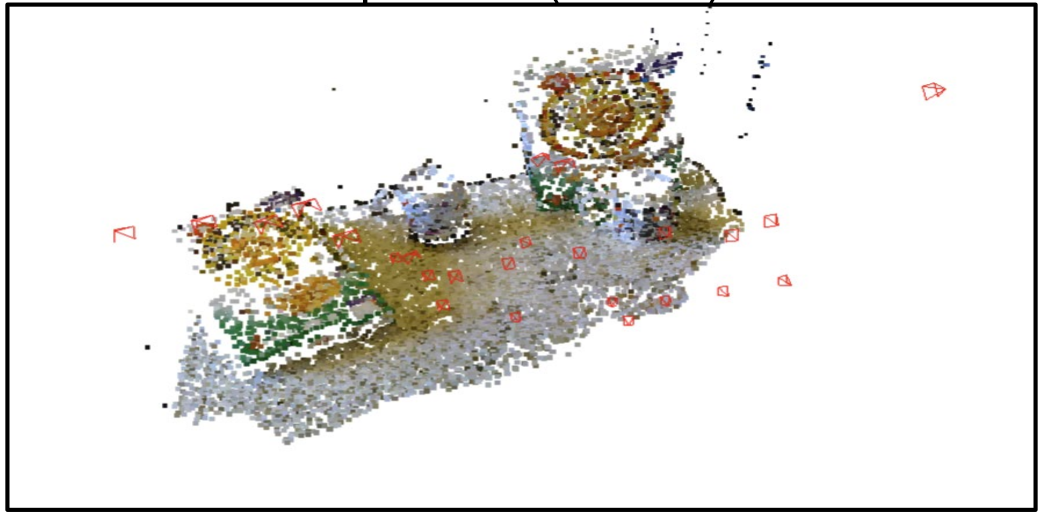
Local resectioning using ambiguity-adjusted matches compared against baselines (standard OpenSfM and COLMAP pipelines)

	OpenSfM	**OpenSfM w/ Our Resectioning**	COLMAP	**COLMAP w/ Our Resectioning**
Duplicate Structures Dataset	6 Failures	6 Successes	6 Failures	2 Failures 4 Successes
UIUCTag Dataset	9 Failures 3 Partial Successes 4 Successes	3 Failures 13 Successes	7 Failures 2 Partial Successes 7 Successes	5 Failures 3 Partial Successes 8 Successes
TanksAndTemples Dataset	4 Failures 3 Successes	1 Failure 6 Successes	1 Failure 6 Successes	7 Successes

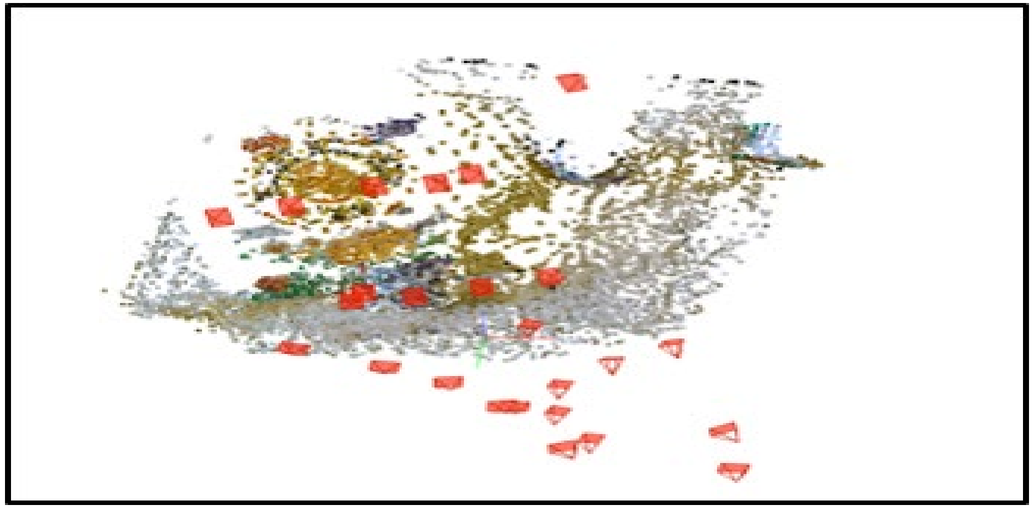
Successful reconstruction of Cereal (DuplicateStructures)

OpenSfM

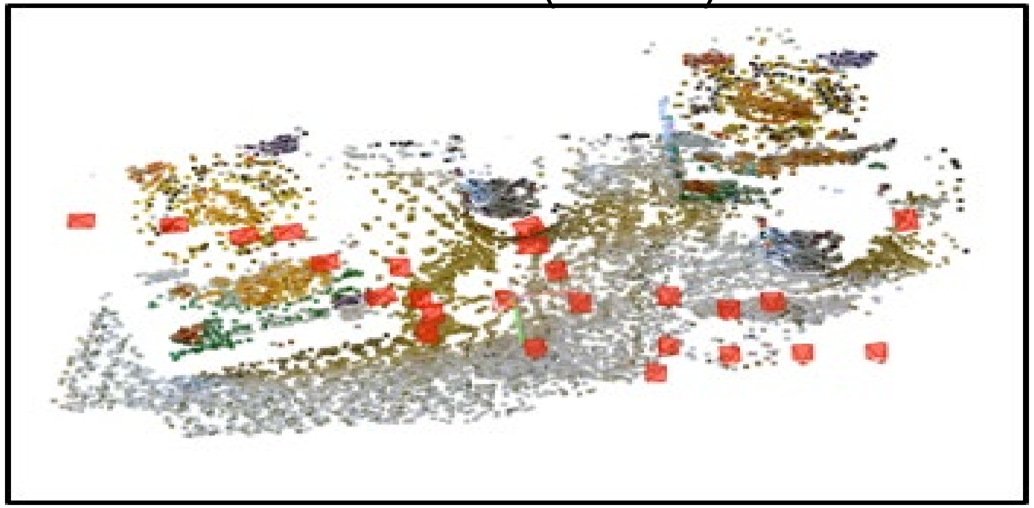
OpenSfM (OURS)



COLMAP

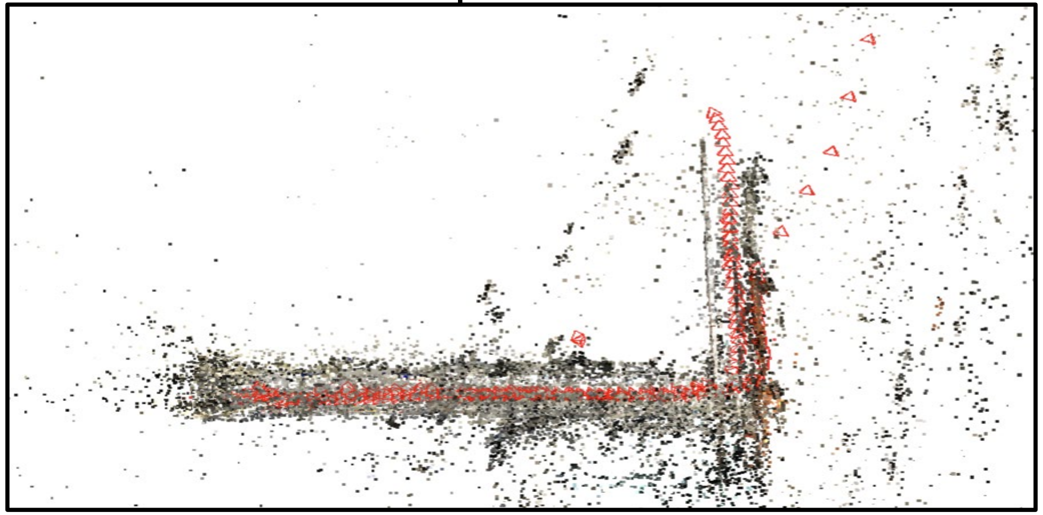


COLMAP (OURS)

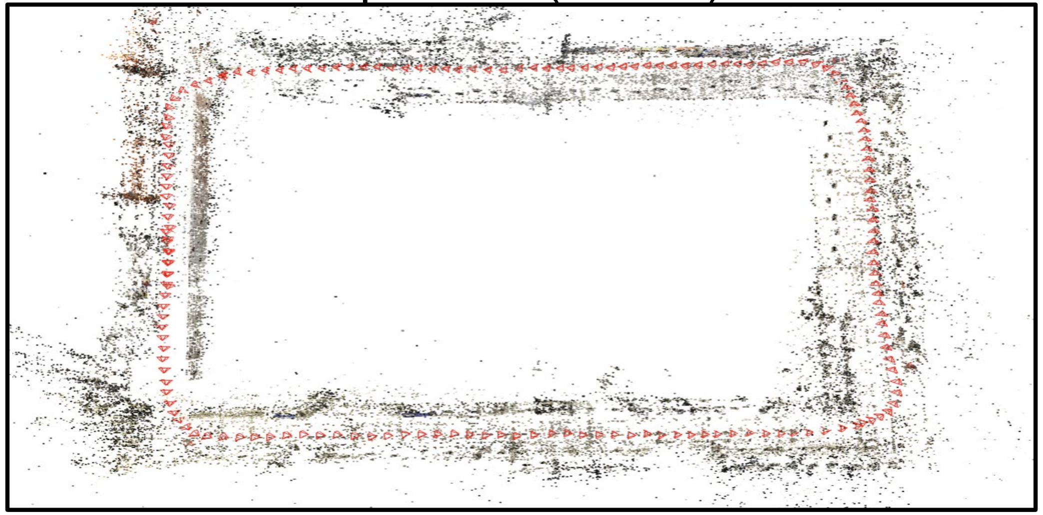


Successful reconstruction of ece_floor3_loop_cw (UIUCTag)

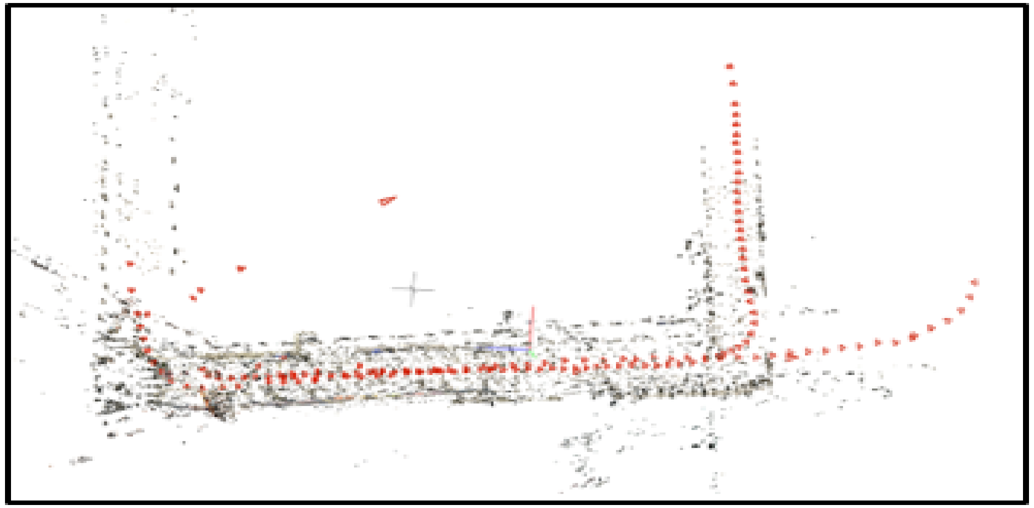
OpenSfM



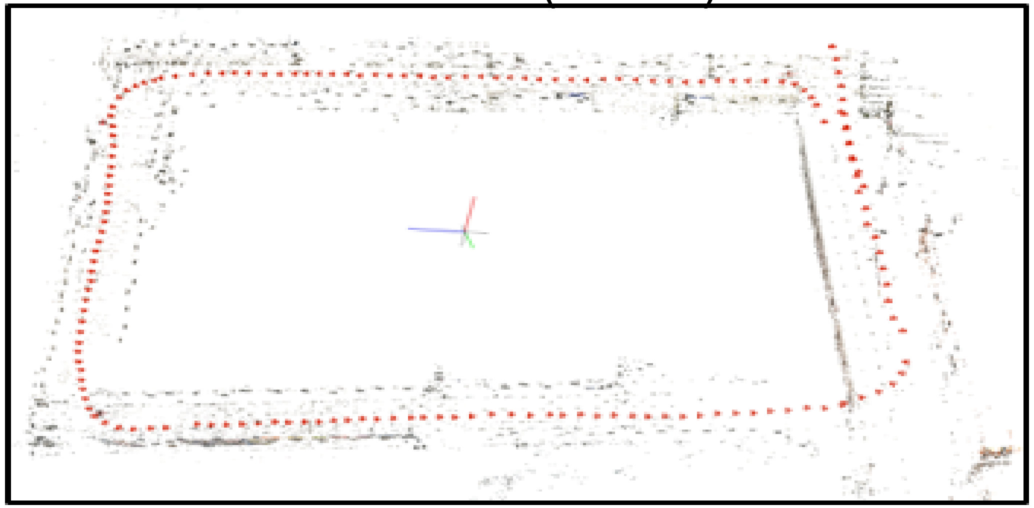
OpenSfM (OURS)



COLMAP

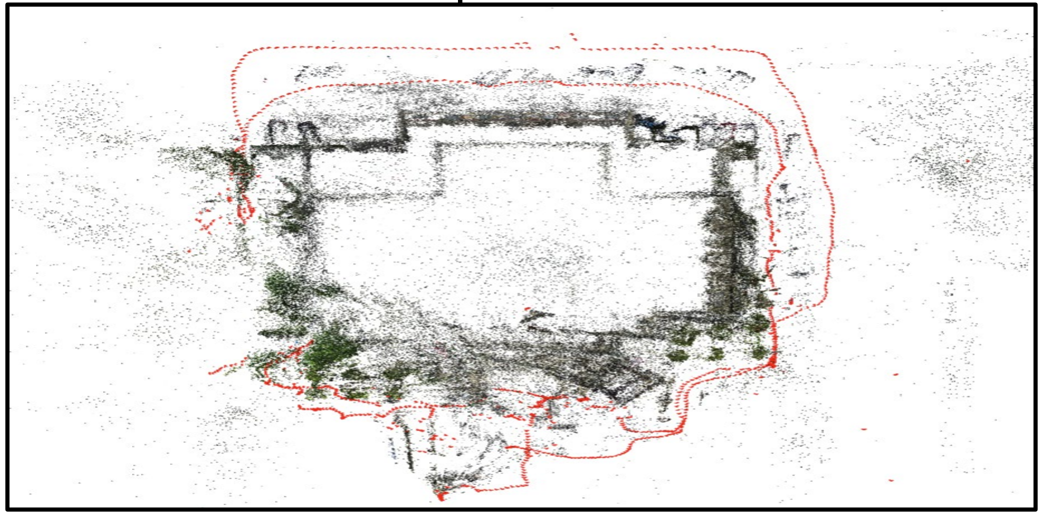


COLMAP (OURS)

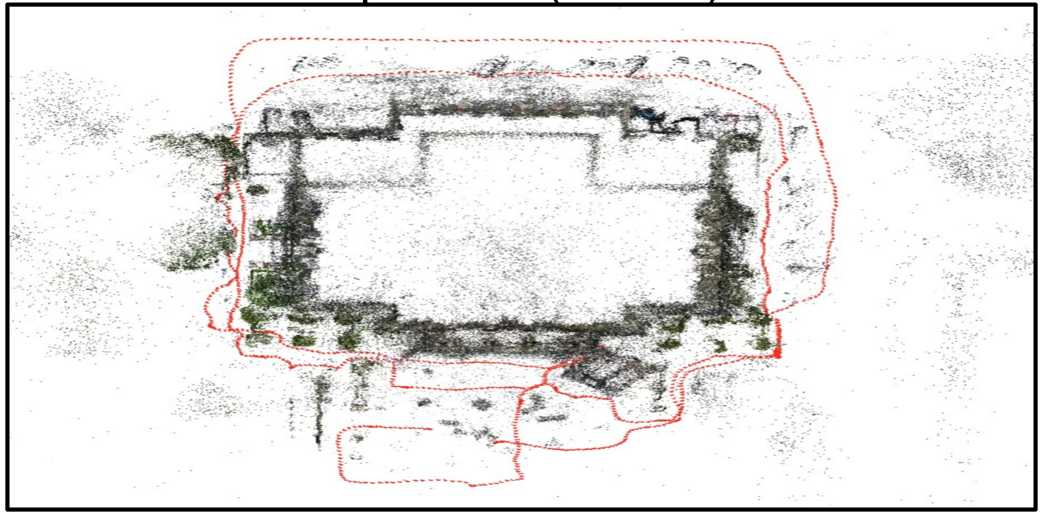


Successful reconstruction of Courthouse (TanksAndTemples)

OpenSfM

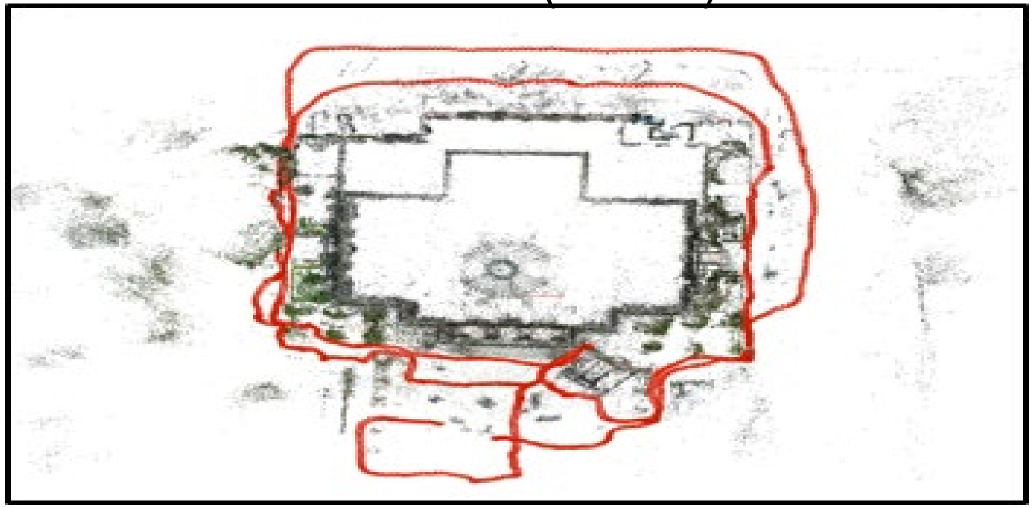


OpenSfM (OURS)



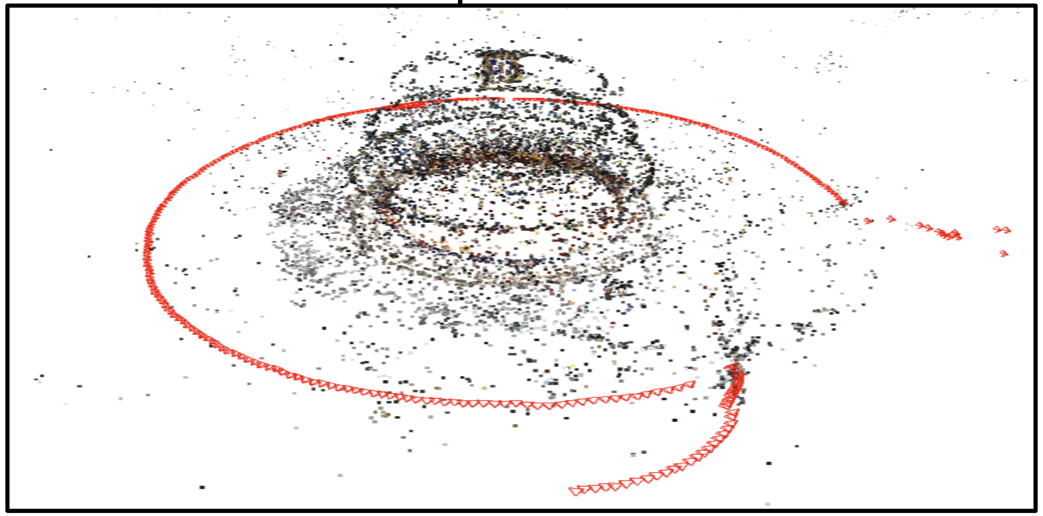
COLMAP

COLMAP (OURS)

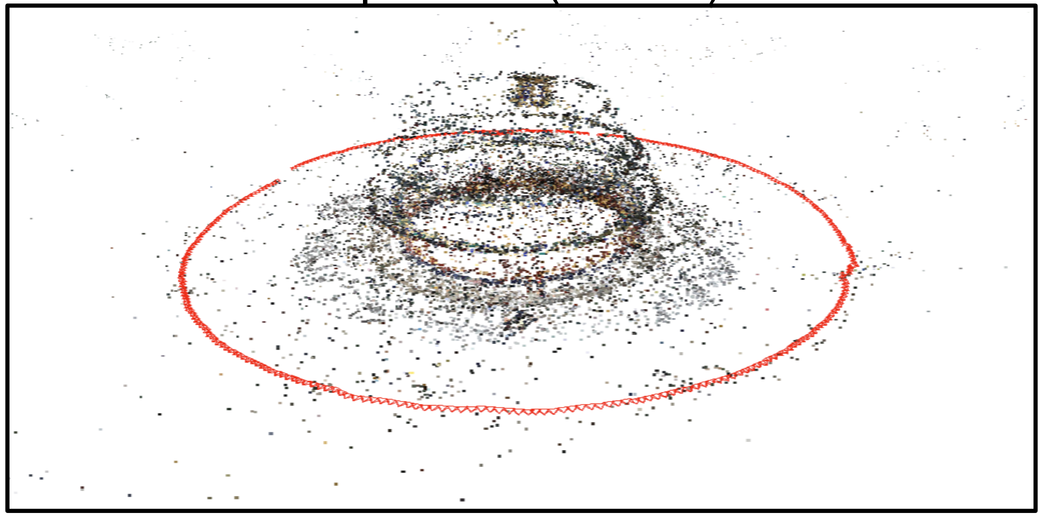


Successful reconstruction of TempleOfHeaven (Internet)

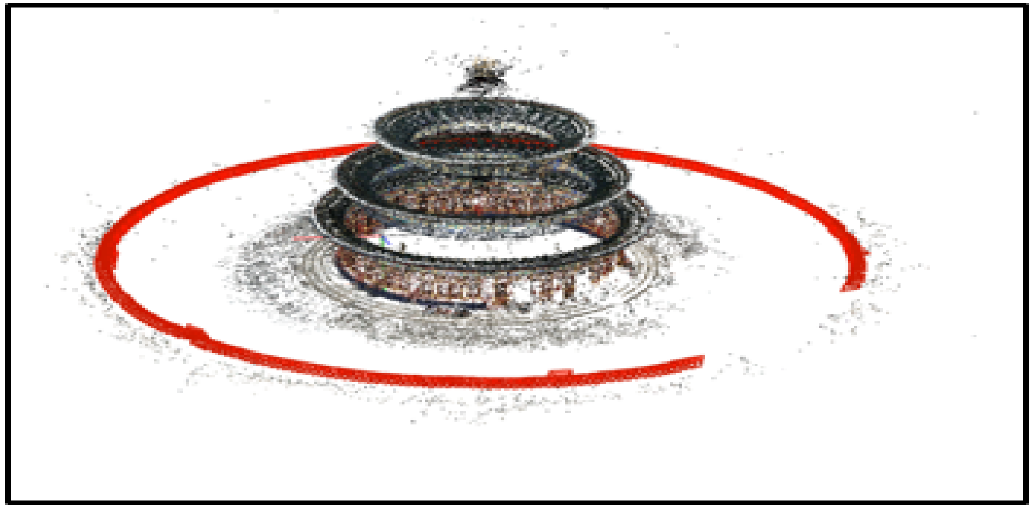
OpenSfM



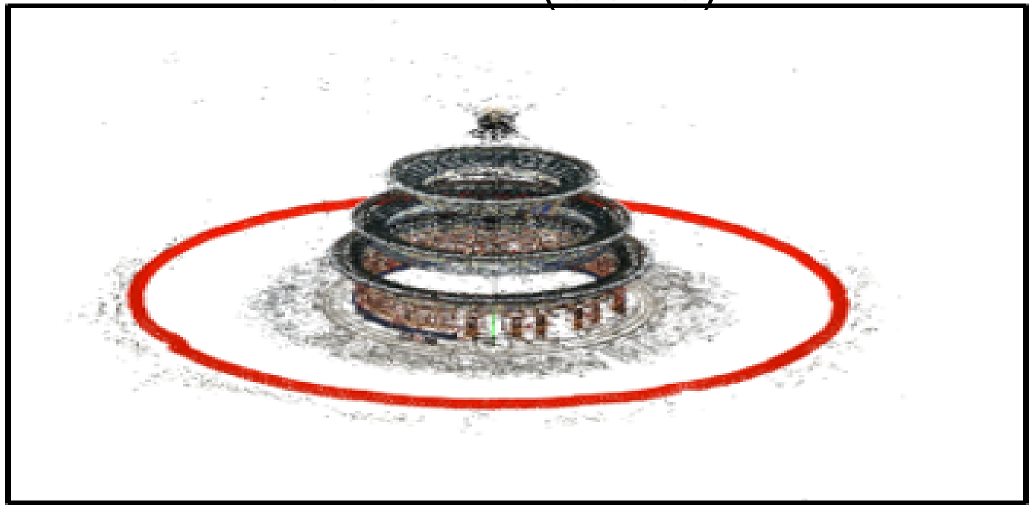
OpenSfM (OURS)



COLMAP



COLMAP (OURS)



Robust Global Translations with 1DSfM

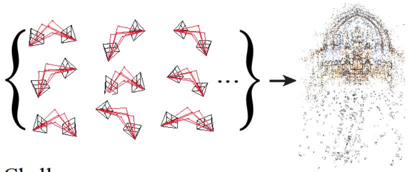
Kyle Wilson
Noah Snavely
{wilsonkl, snavely}@cs.cornell.edu
Code and Datasets: www.cs.cornell.edu/projects/1DSfM

Problem Statement

Incremental SfM is expensive and error-prone. We explore global methods to solve the problem in one shot.

Goal:

Build a 3D model in one shot given many two-view models. We use Chatterjee and Govindu [1] to solve for rotations, and focus only on translations.



Challenges:

- Many formulations of the translations problem are non-convex. A solver must find a good solution **reliably**.
- Translations problems generally contain **outliers**. These bad measurements can reduce solution quality and make it harder for solvers to converge.

Contributions:

1DSfM: a simple way to detect outlier translation measurements using 1D subproblems

Solver: a new approach to solving translations problems using nonlinear optimization

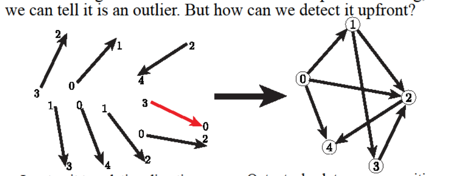
Takeaway:

We pose a translations problem as a standard nonlinear optimization, which, coupled with outlier removal, yields good results even when initialized randomly.

Contribution 1: Outlier Removal with 1DSfM

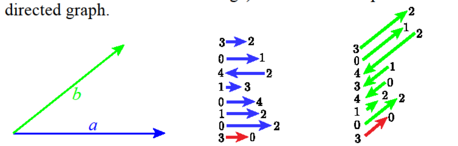
Left: an example translations problem
Right: the correct solution

An outlier edge is shown in red. Given the output embedding, we can tell it is an outlier. But how can we detect it upfront?



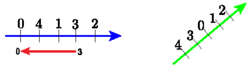
Input unit translation directions — Output: absolute camera positions

1D subproblems are easier: we project the problem onto a single unit vector, so each edge becomes a simple plus/minus sign (due to the unknown scale of each edge) which we can represent as a directed graph.



Two projection directions — 1D subproblem (a) — 1D subproblem (b)

These 1D problems are instances of Minimum Feedback Arc Set [2]. Solving them means choosing a best ordering. Outlier edges may not be consistent with the others.



solution to (a) — solution to (b)

Outliers won't be detected in some projections. We project in many random directions and reject edges that are frequently inconsistent.

Contribution 2: New Translations Solver

We want to solve problems of this general form:

Given:	a directed graph $G = (V, E)$ 3D translation directions $t : E \to S^2$
Compute:	an embedding $X : V \to \mathbb{R}^3$ (up to scale and translation)
Such that:	the translation directions induced by X are close to t

We compare poses in the **measurement space** of unit vectors with the squared chordal distance.

$$\hat{X} = \underset{X}{\operatorname{argmin}} \sum_{(i,j)\in E} d_{ch}\left(t_{ij}, \frac{X_j - X_i}{\|X_j - X_i\|}\right)^2$$

$$d_{ch}(u, v) = \|u - v\|$$

Properties:

- Nonlinear Least Squares problem (**NLLS**)—we use Ceres [3]
- Well-behaved error surface, especially after 1DSfM
- Can additionally use a Huber loss for even greater robustness
- Geometrically meaningful: **MLE** of the error model below

$$f(t_{ij}|X) \propto exp\left[\frac{-d^2_{ch}}{\sigma^2}\right]$$

Convergence:

- NLLS is a local optimizer—global convergence not guaranteed
- Surprisingly, **we find good solutions, even from random initializations**
- Plausibility: for a noise free problem, the error surface is decreasing towards the global optimum. It deviates from this behavior slowly as noise increases:

$$d^2_{ch}(t, X_\lambda) \le d^2_{ch}(t, X_1) + d^2_{ch}(t, X_{opt})$$

$$\text{where } X_\lambda = \lambda X_1 + (1-\lambda)X_{opt}, \quad 0 \le \lambda \le 1$$

Results

- 13 large datasets—all new (except Notre Dame, from [5])
- state of the art results
- datasets and code available

We evaluate our results by robustly rigidly aligning solutions to models produced by Bundler, in incremental SfM solver [5].

The numbers below are errors in meters after a final bundle adjustment.

			no 1DSfM		with 1DSfM		[4]
Name	Size	N_c	$\tilde{x}$	$\bar{x}$	$\tilde{x}$	$\bar{x}$	$\tilde{x}$
Piccadilly	80	2152	**0.3**	9e3	0.7	**7e2**	10
Union Square	300	789	**3.2**	2e2	3.4	**9e1**	10
Roman Forum	200	1084	2.7	9e5	**0.2**	**3e0**	37
Vienna Cathedral	120	836	0.7	7e4	**0.4**	2e4	12
Piazza del Popolo	60	328	**1.6**	**9e1**	2.2	2e2	16
NYC Library	130	332	**0.2**	8e1	0.4	**1e0**	1.4
Alamo	70	577	**0.2**	**7e5**	0.3	2e7	2.4
Metropolis	200	341	0.6	**3e1**	**0.5**	7e1	18
Yorkminster	150	437	0.4	9e3	**0.1**	**5e2**	6.7
Montreal N.D.	30	450	**0.1**	**4e-1**	0.4	1e0	9.8
Tower of London	300	572	**0.2**	3e4	1.0	**4e1**	44
Ellis Island	180	227	0.3	3e0	0.3	3e0	8.0
Notre Dame	300	553	**0.8**	7e4	1.9	**7e0**	2.1

Dataset sizes are given in both meters and number of cameras. The table shows median and mean camera error.

We significantly outperform an existing method [4]. 1DSfM often results in a similar median error, but a greatly improved average. Runtimes are 3-12x faster than [5].

References

[1] Chatterjee, A., Govindu, V.M. Efficient and robust large-scale rotation averaging. ICCV 2013.
[2] Eades, P., Lin, X., Smyth,W.F. A fast and effective heuristic for the feedback arc set problem. Information Processing Letters (1993).
[3] Agarwal, S., Mierle, K., Others. Ceres solver. https://code.google.com/p/ceres-solver/
[4] Govindu, V.M. Combining two-view constraints for motion estimation. CVPR 2001.
[5] Snavely, N., Seitz, S., Szeliski, R.: Photo tourism: Exploring photo collections in 3D. SIGGRAPH 2006.

All Results

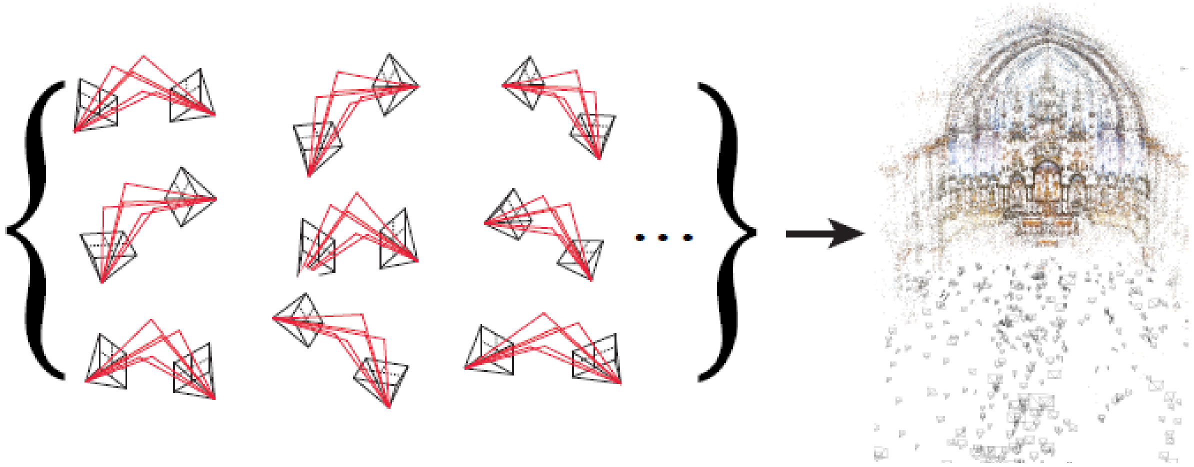
Alamo — NYC Library — Ellis Island — Tower of London — Roman Forum — Vienna Cathedral — Montreal Notre Dame — Piazza del Popolo — Metropolis — Notre Dame — Yorkminster — Union Square — Piccadilly (2152 images) — Trafalgar (4597 images)

Problem Statement

Incremental SfM is expensive and error-prone. We explore global methods to solve the problem in one shot.

Goal:

Build a 3D model in one shot given many two-view models. We use Chatterjee and Govindu [1] to solve for rotations, and focus only on translations.



Challenges:

- Many formulations of the translations problem are non-convex. A solver must find a good solution **reliably**.
- Translations problems generally contain **outliers**. These bad measurements can reduce solution quality and make it harder for solvers to converge.

Contributions:

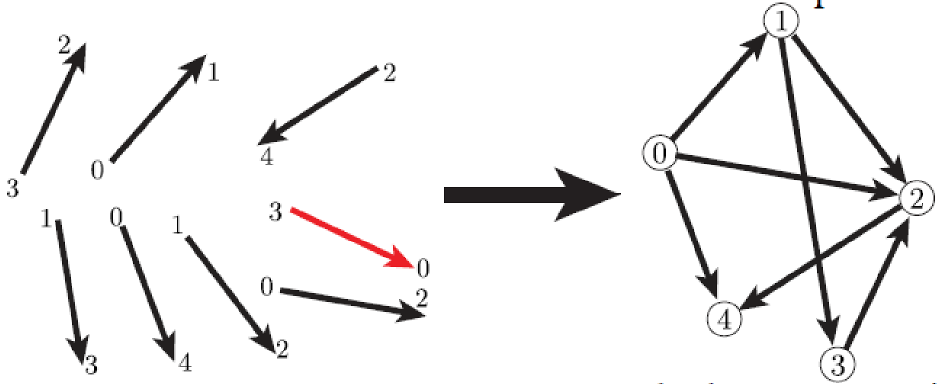
1DSfM: a simple way to detect outlier translation measurements using 1D subproblems

Solver: a new approach to solving translations problems using nonlinear optimization

Contribution 1: Outlier Removal with 1DSfM

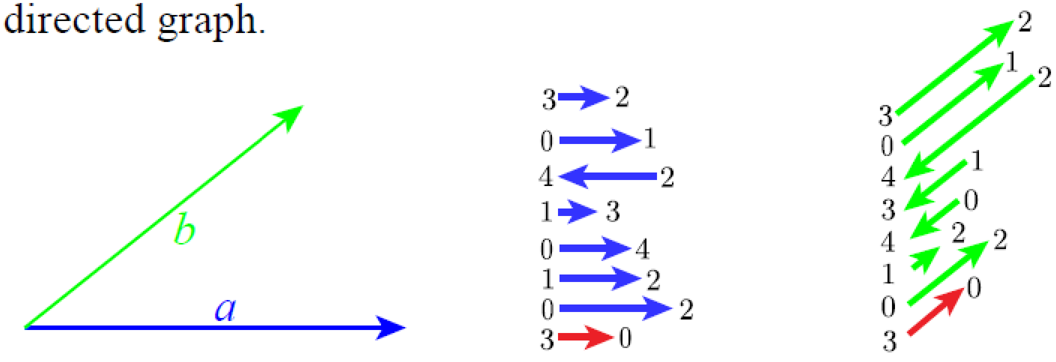
Left: an example translations problem
Right: the correct solution

An outlier edge is shown in red. Given the output embedding, we can tell it is an outlier. But how can we detect it upfront?



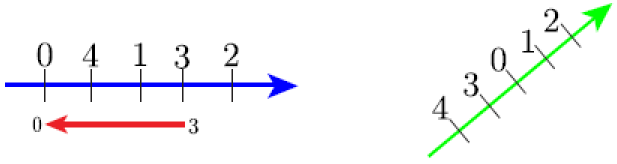
Input unit translation directions   Output: absolute camera positions

1D subproblems are easier: we project the problem onto a single unit vector, so each edge becomes a simple plus/minus sign (due to the unknown scale of each edge) which we can represent as a directed graph.



Two projection directions   1D subproblem (a)   1D subproblem (b)

These 1D problems are instances of Minimum Feedback Arc Set [2]. Solving them means choosing a best ordering. Outlier edges may not be consistent with the others.



solution to (a)   solution to (b)

Outliers won't be detected in some projections. We project in many random directions and reject edges that are frequently inconsistent.

Contribution 2: New Translations Solver

We want to solve problems of this general form:

Given:	a directed graph $G = (V, E)$ 3D translation directions $t : E \to S^2$
Compute:	an embedding $X : V \to \mathbb{R}^3$ (up to scale and translation)
Such that:	the translation directions induced by X are close to t

We compare poses in the **measurement space** of unit vectors with the squared chordal distance.

$$\hat{X} = \underset{X}{\operatorname{argmin}} \sum_{(i,j)\in E} d_{ch}\left(t_{ij}, \frac{X_j - X_i}{\|X_j - X_i\|}\right)^2$$

$$d_{ch}(u, v) = \|u - v\|$$

Properties:

- Nonlinear Least Squares problem (**NLLS**)—we use Ceres [3]
- Well-behaved error surface, especially after 1DSfM
- Can additionally use a Huber loss for even greater robustness

Results

- 13 large datasets—all new (except Notre Dame, from [5])
- state of the art results
- datasets and code available

We evaluate our results by robustly rigidly aligning solutions to models produced by Bundler, in incremental SfM solver [5].

The numbers below are errors in meters after a final bundle adjustment.

			no 1DSfM		with 1DSfM		[4]
Name	Size	N_c	$\widetilde{x}$	$\bar{x}$	$\widetilde{x}$	$\bar{x}$	$\widetilde{x}$
Piccadilly	80	2152	**0.3**	9e3	0.7	**7e2**	10
Union Square	300	789	**3.2**	2e2	3.4	**9e1**	10
Roman Forum	200	1084	2.7	9e5	**0.2**	**3e0**	37
Vienna Cathedral	120	836	0.7	7e4	**0.4**	2e4	12
Piazza del Popolo	60	328	**1.6**	**9e1**	2.2	2e2	16
NYC Library	130	332	**0.2**	8e1	0.4	**1e0**	1.4
Alamo	70	577	**0.2**	**7e5**	0.3	2e7	2.4
Metropolis	200	341	0.6	**3e1**	**0.5**	7e1	18
Yorkminster	150	437	0.4	9e3	**0.1**	**5e2**	6.7
Montreal N.D.	30	450	**0.1**	**4e-1**	0.4	1e0	9.8
Tower of London	300	572	**0.2**	3e4	1.0	**4e1**	44
Ellis Island	180	227	0.3	3e0	0.3	3e0	8.0
Notre Dame	300	553	**0.8**	7e4	1.9	**7e0**	2.1

Dataset sizes are given in both meters and number of cameras. The table shows median and mean camera error.

Incremental vs. Global SfM

- Incremental includes more outlier checks and generates more precise results but take much longer

- Global is much faster but does not as effectively remove outliers and provides an approximate solution that is not precise enough (in my experience) for MVS

Open problems / research ideas

- Improved matching
  - Learned features, especially for handling large viewpoint, scale, or time differences, or features for low-texture regions
- Improved outlier rejection
  - Perhaps global SfM outlier checks can benefit incremental SfM
- Improved speed
  - Hybrid global/incremental and hierarchical systems
  - Online SfM / MVS
- Improved standard evaluations
  - More real-world scenarios like inspection instead of internet collections

Summary

- Structure-from-Motion usually works (95% of the time)
  - But it matters when it doesn't work
- Incremental SfM is most precise, but Global SfM is faster
- Main practical challenges (beyond speed) stem from feature matching in poor light environments, textureless surfaces, and large baselines and scale differences