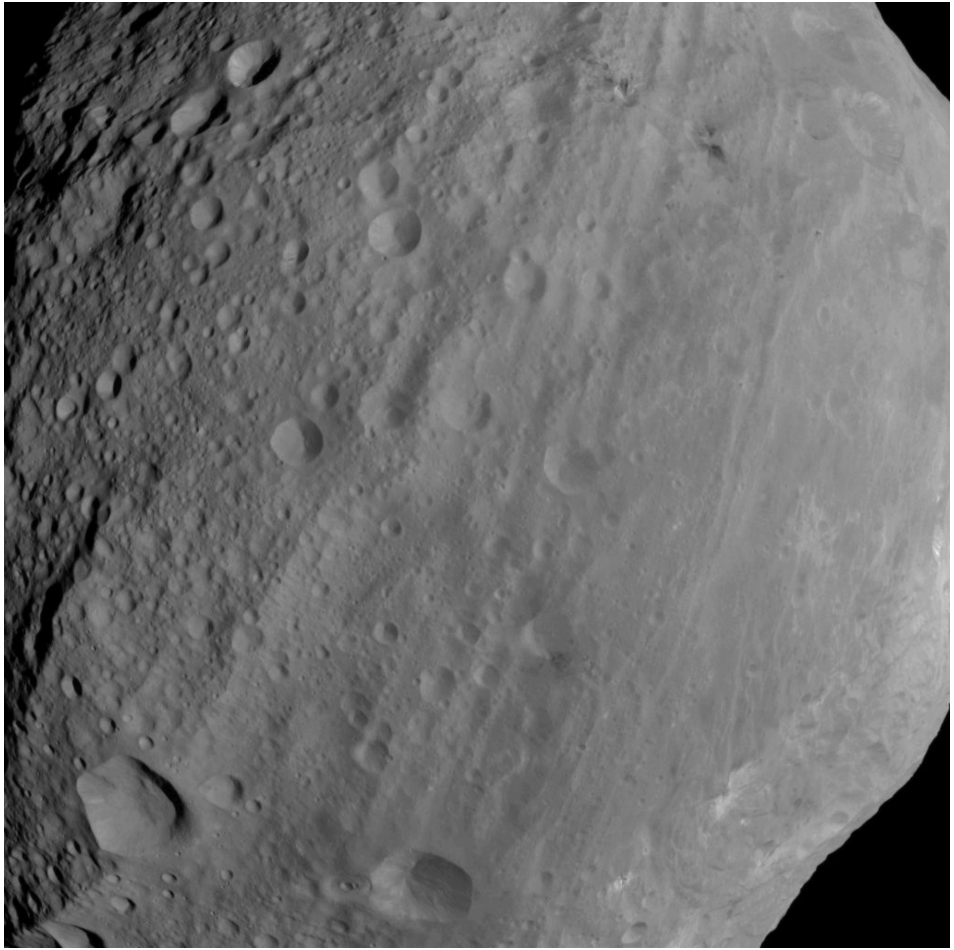
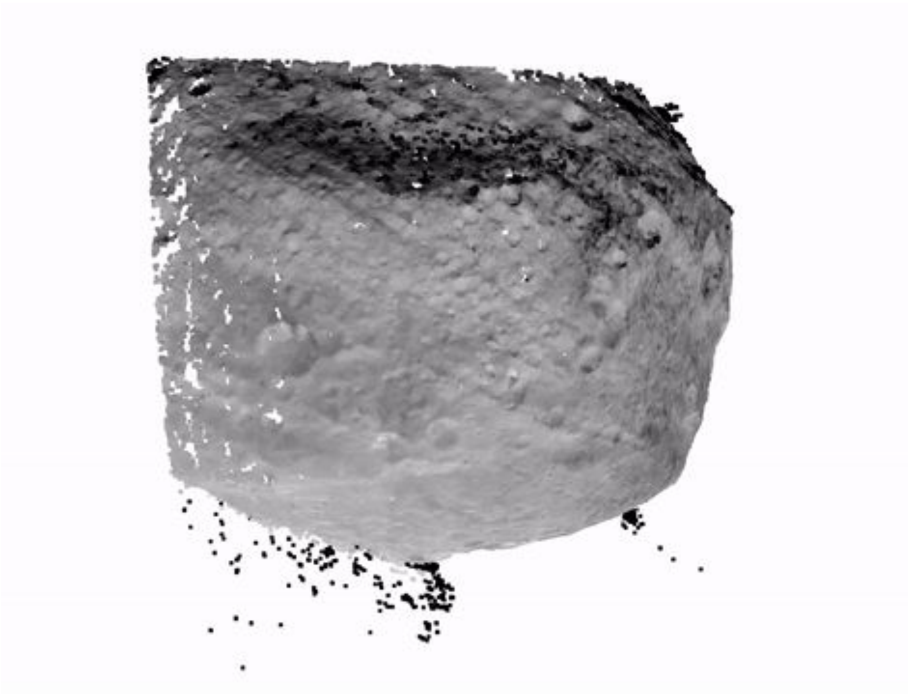
GTSFM: Georgia Tech Structure from Motion

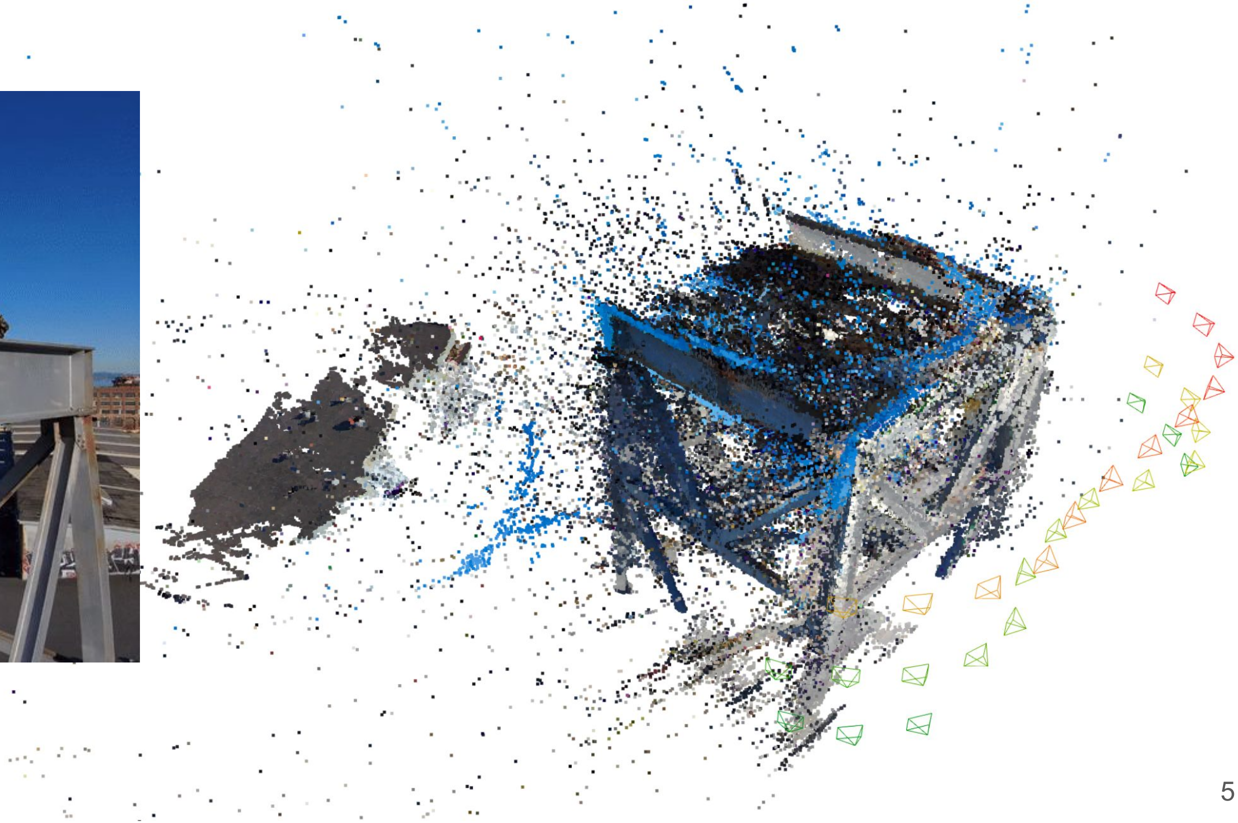
Presented by John Lambert

Feb 15, 2021

John Lambert, Ayush Baid, Akshay Krishnan, Adi Singh, Xiaolong Wu, Alex Butenko,
Ren Liu, Fan Jiang, Sushmita Warrier, Jing Wu, Travis Driver, Neha Upadhyay,
Pratyusha Maiti, Jonathan Womack, Xinpei Ni, James Hays, Frank Dellaert

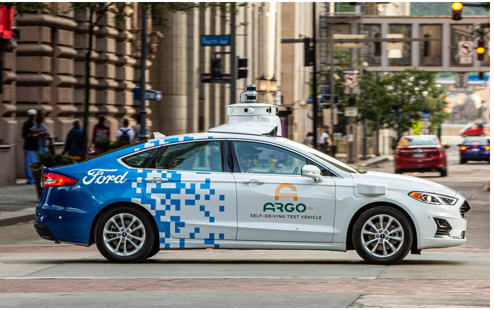


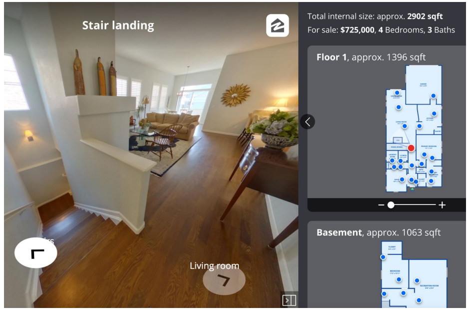




Motivation:
Why build and validate maps?

Mapping → Spatial AI



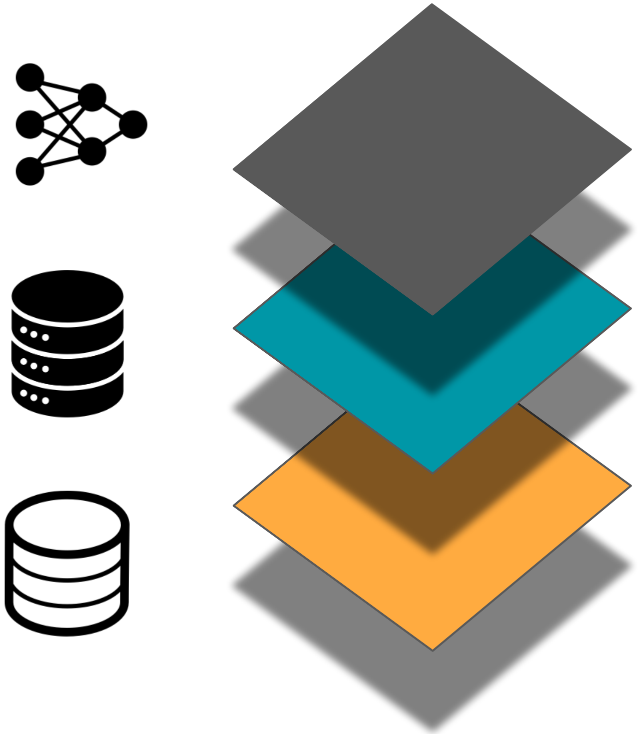


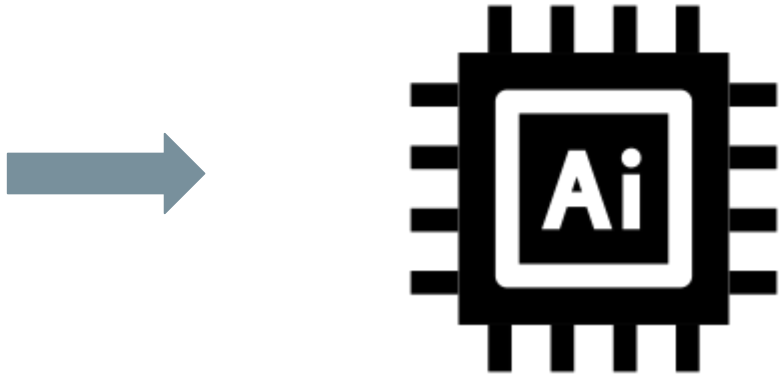
Mapping → Spatial AI

Algorithms

Semantic/
HD map

Geometric
map





Autonomy

1. Davison. FutureMapping: The Computational Structure of Spatial AI Systems. Arxiv, ‘18.
2. Sarlin et al., Pixel-Perfect Structure-from-Motion, ICCV ‘21.

Figure source: https://matterport.com/gallery/ngorongoro-oldeani-mountain-lodge

Figure source: https://www.youtube.com/watch?v=2eYSzmjT6HI

What is a map?

- Not just a geometric model.
- Any object or information that is localized in 2D or 3D that can prove useful.

Figure source: Zoox, https://www.youtube.com/watch?v=JAHva2-x1wg

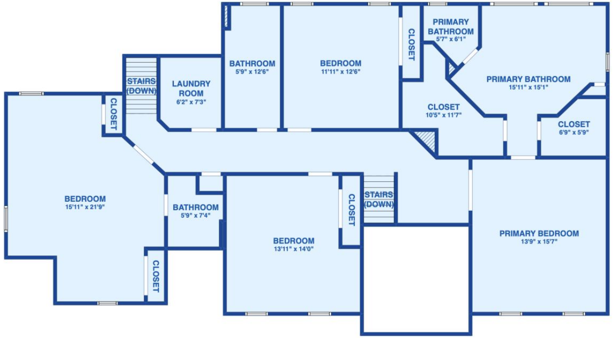


Figure source: Cruz, Zillow Indoor Dataset, CVPR 2021

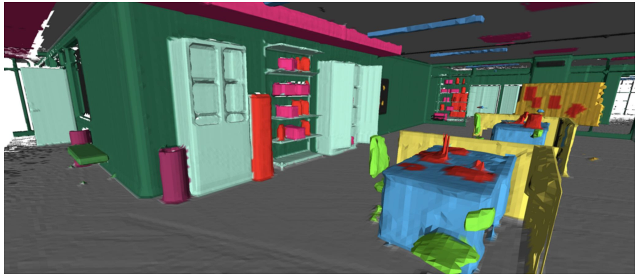


Figure Source: Rosinol, ICRA '20

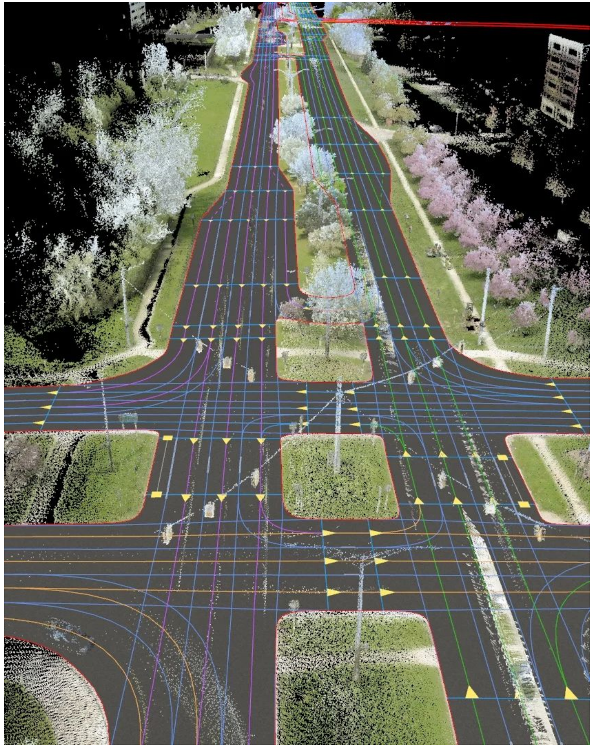


Figure source: https://360.here.com/2015/07/20/here-introduces-hd-maps-for-highly-automated-vehicle-testing/

3D Geometric Maps

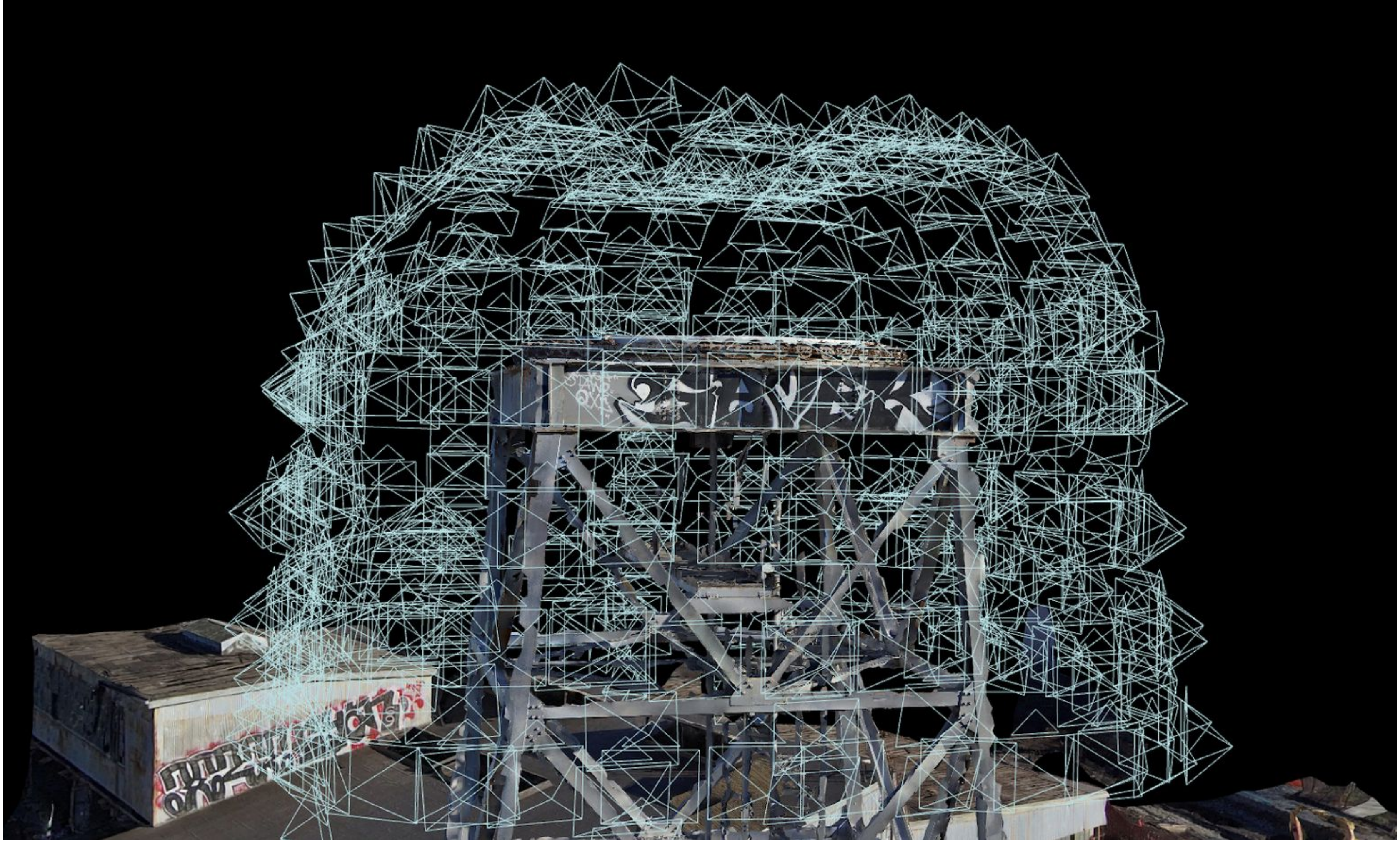
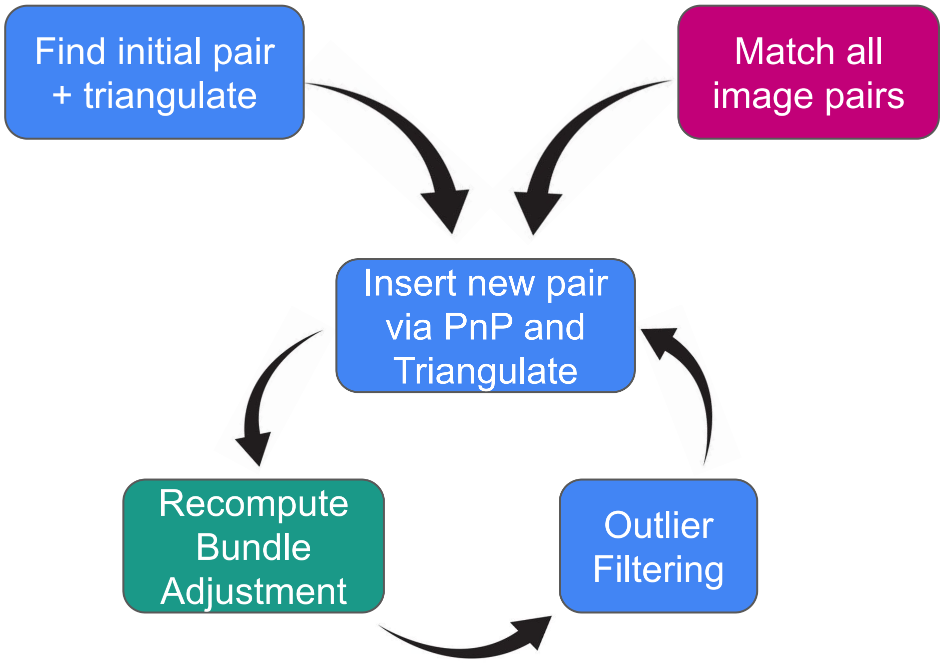
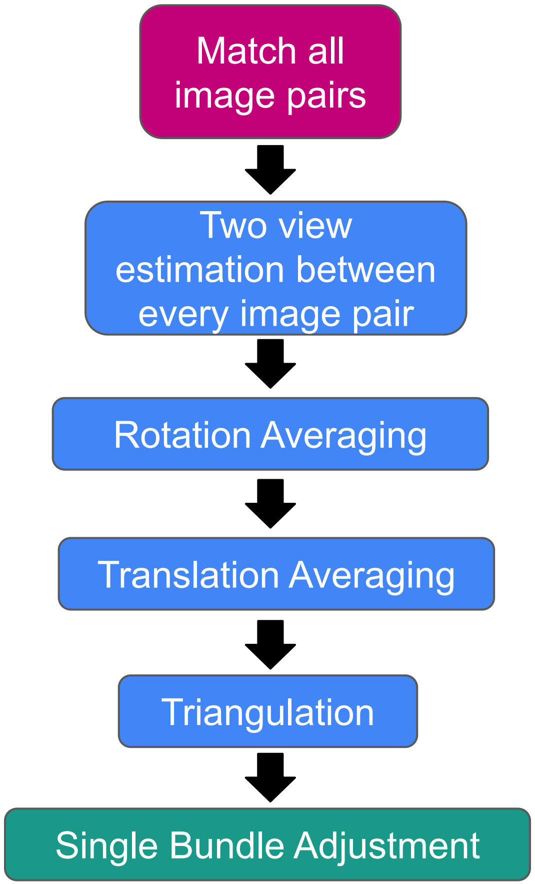


Figure source: https://www.skydio.com/blog/3d-scan-sneak-peek-crane-mast/

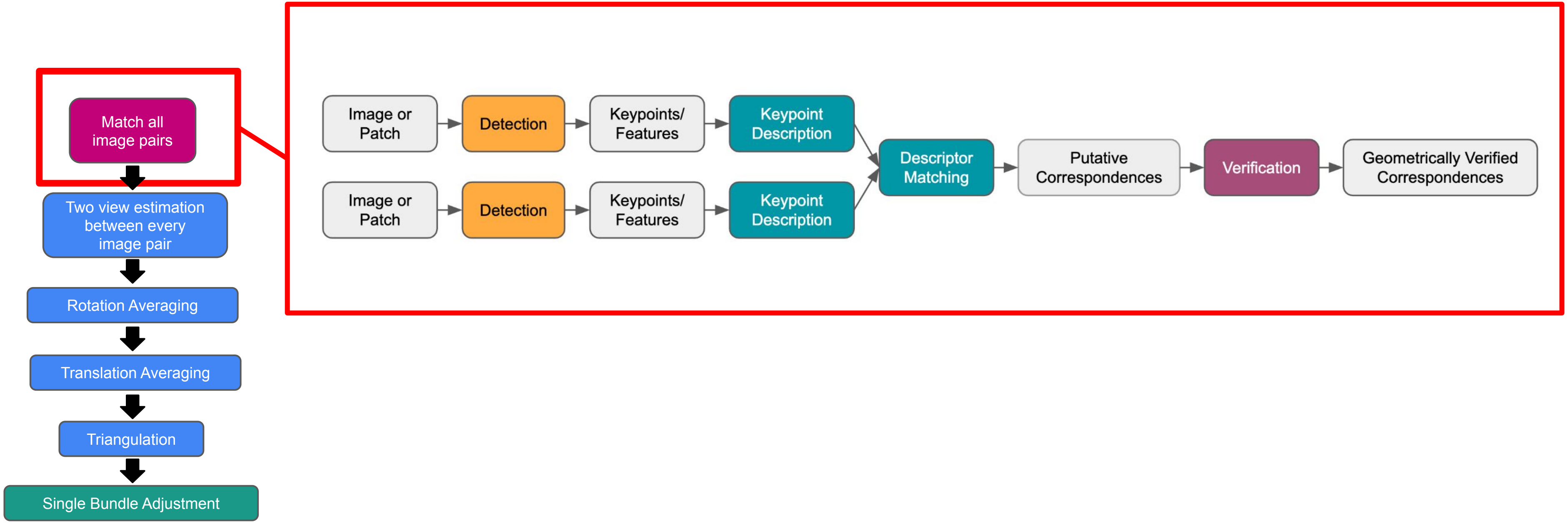


Incremental SfM

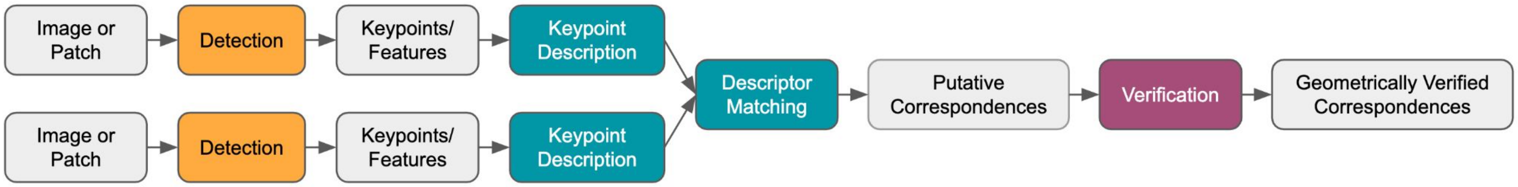


Global SfM

The Deep Front-End



What’s the point?

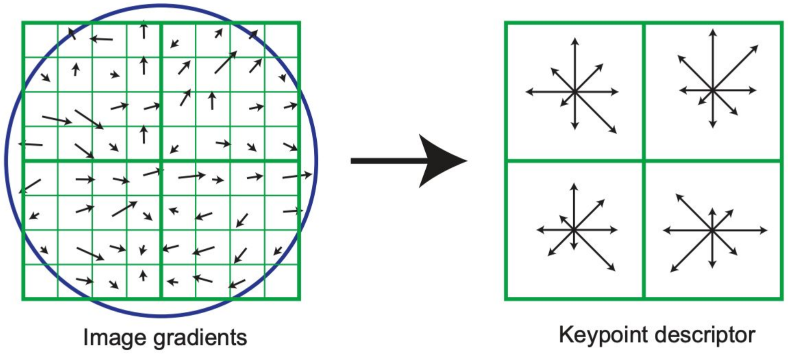


Feature Detectors	Feature Descriptors	Feature Matchers	Correspondence Verifiers
FAST, TILDE, QuadNet, DDet/CovDet, Key.Net, GLAMPoints, ...	PCA-SIFT, Winder 07, ConvOpt, MatchNet, DeepDesc, L2Net, TFeat, UCN, HardNet, SOSNet, BeyondCartesian, ...	SuperGlue	Deep F-Matrix, LearnedCorr, Eig-Free, N3-Net, NM-Net, OA-Net, NGRANSAC, ...
ContextDesc, D2-Net, LF-Net, R2D2, IMIPS, LIFT, SuperPoint, ReinforcedSuperPoint, ...			

*CNN- or GNN-based.

Correspondence: paper vs. practice

System	Feature Matching Module
VisualSfM (2013)	SIFT
OpenMVG (2013)	SIFT + A-Contrario RANSAC
OpenSfM (2014)	Hessian Affine + SIFT Descriptor + RANSAC
COLMAP* (2016)	SIFT + LoRANSAC



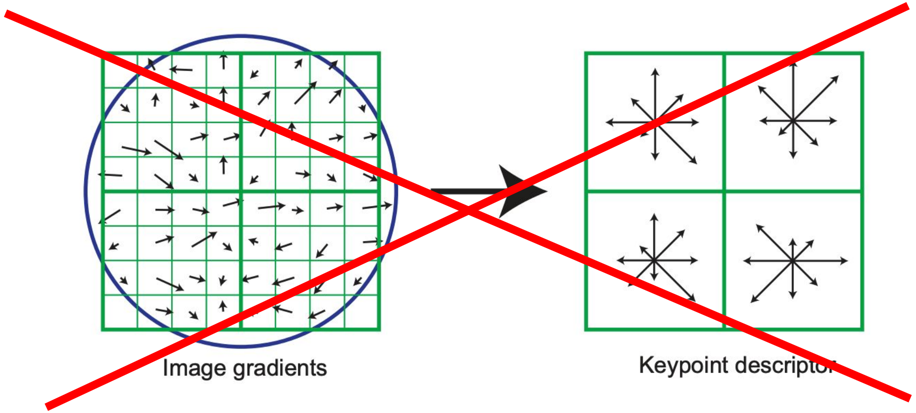
*State of the Art (per Knapitsch et al., 2017)

Figure sources: Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004.

Correspondence: paper vs. practice

System	Feature Matching Module
VisualSfM (2013)	SIFT
OpenMVG (2013)	SIFT + A-Contrario RANSAC
OpenSfM (2014)	Hessian Affine + SIFT Descriptor + RANSAC
COLMAP* (2016)	SIFT + LoRANSAC

*State of the Art (per Knapitsch et al., 2017)



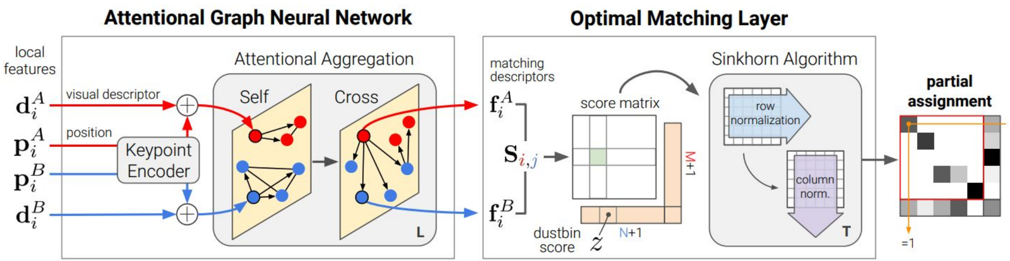
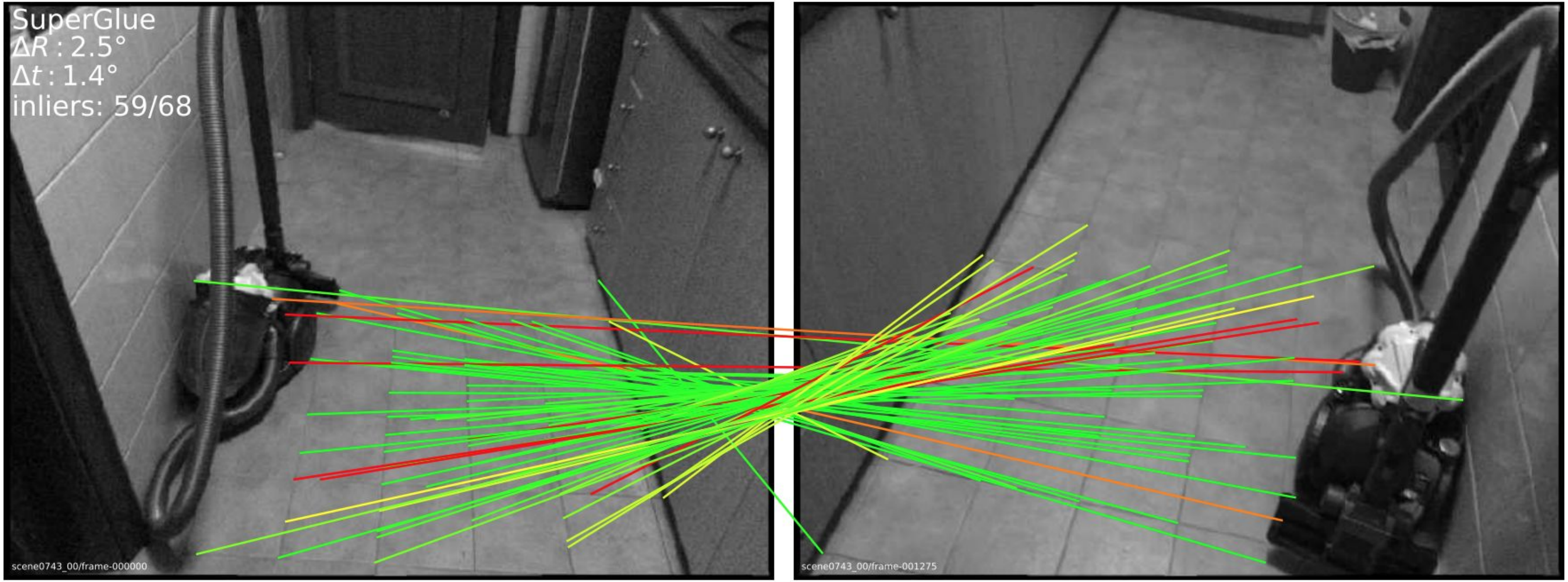
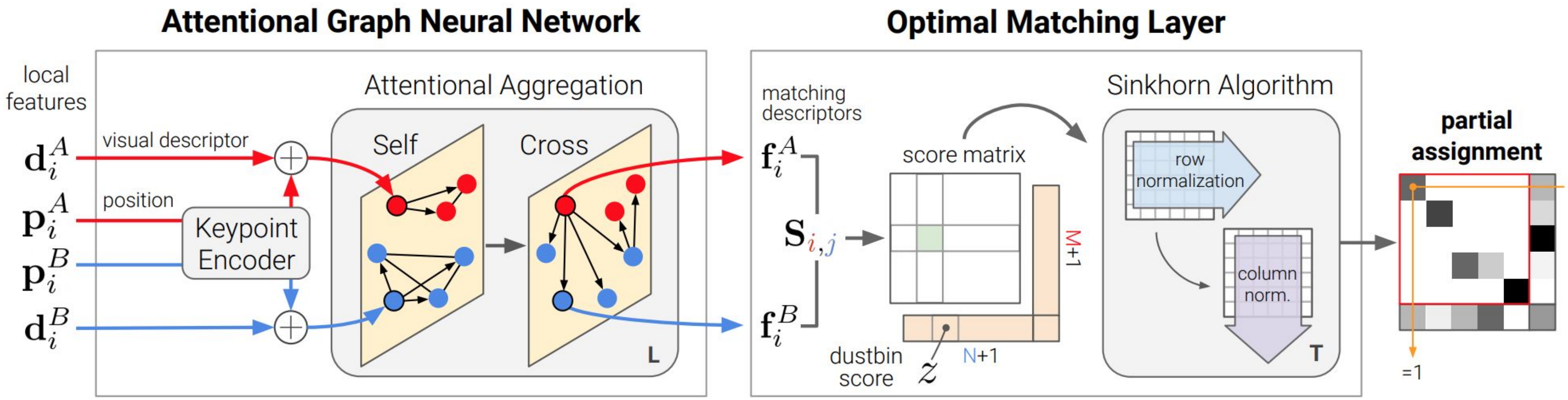


Figure sources: Lowe, Distinctive Image Features from Scale-Invariant Keypoints, IJCV 2004.
Sarlin, SuperGlue, CVPR 2020.


SuperGlue
$\Delta R: 2.5°$
$\Delta t: 1.4°$
inliers: 59/68
scene0743_00/frame-000000
scene0743_00/frame-001275

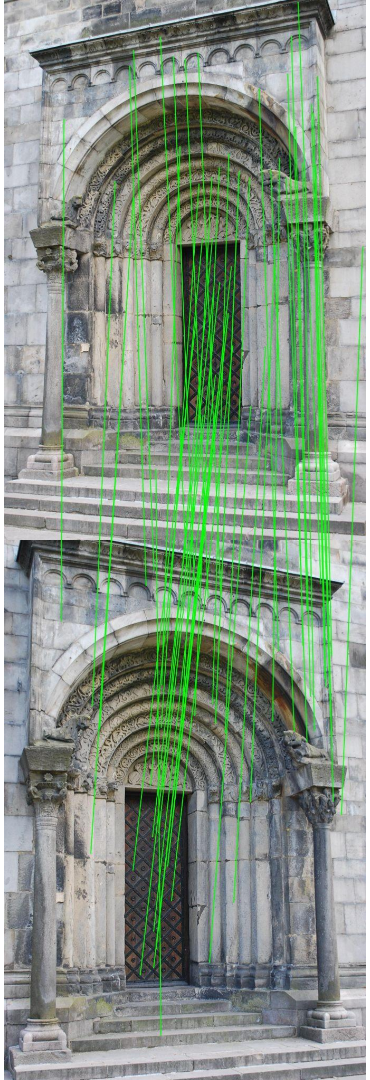
SuperGlue Network Architecture

Goal: design a neural network that predicts the assignment P from two sets of local features.



Building 3d Geometric Maps Using Deep Learning

Feature Matching



Decomposing **F** into **R** and **T**

If we have calibrated cameras we have $\mathbf{K}$ and $\mathbf{K}'$

Essential matrix: $\mathbf{E} = \mathbf{K}'^{\top}\mathbf{F}\mathbf{K}$

Decomposing **F** into **R** and **T**

If we have calibrated cameras we have $\mathbf{K}$ and $\mathbf{K}'$

Essential matrix: $\mathbf{E} = \mathbf{K}'^{\top}\mathbf{F}\mathbf{K}$

$$\mathbf{E} = [\mathbf{t}]_{\times}\mathbf{R}$$

SVD: $\mathbf{E} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\top}$ Let $\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Slide Credit: Kris Kitani

Decomposing **F** into **R** and **T**

If we have calibrated cameras we have $\mathbf{K}$ and $\mathbf{K}'$

Essential matrix: $\mathbf{E} = \mathbf{K}'^{\top}\mathbf{F}\mathbf{K}$

SVD: $\mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\top}$ Let $\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

We get FOUR solutions:

$$\mathbf{E} = [\mathbf{R}|\mathbf{T}]$$

$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^{\top}$ $\mathbf{R}_2 = \mathbf{U}\mathbf{W}^{\top}\mathbf{V}^{\top}$

two possible rotations

$\mathbf{T}_1 = U_3$ $\mathbf{T}_2 = -U_3$

two possible translations

Slide Credit: Kris Kitani

We get FOUR solutions:

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top \qquad \mathbf{T}_1 = U_3$$

$$\mathbf{R}_1 = \mathbf{U}\mathbf{W}\mathbf{V}^\top \qquad \mathbf{T}_2 = -U_3$$

$$\mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top \qquad \mathbf{T}_2 = -U_3$$

$$\mathbf{R}_2 = \mathbf{U}\mathbf{W}^\top\mathbf{V}^\top \qquad \mathbf{T}_1 = U_3$$

Which one do we choose?

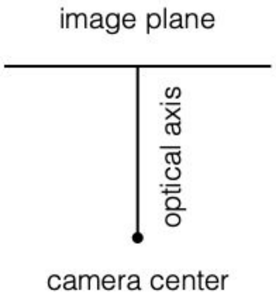
Compute determinant of R, valid solution must be equal to 1
(note: det(R) = -1 means rotation and reflection)

Compute 3D point using triangulation, valid solution has positive Z value
(Note: negative Z means point is behind the camera)

Slide Credit: Kris Kitani

Let's visualize the four configurations…

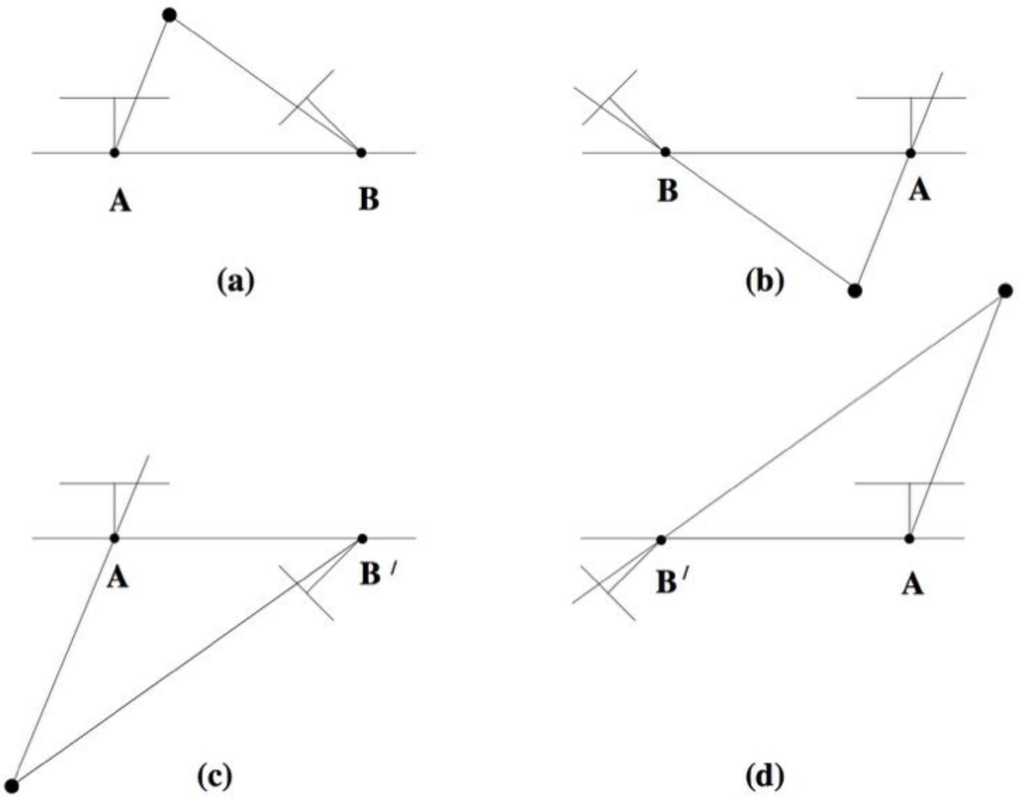
Camera Icon



Find the configuration where the points is in front of both cameras

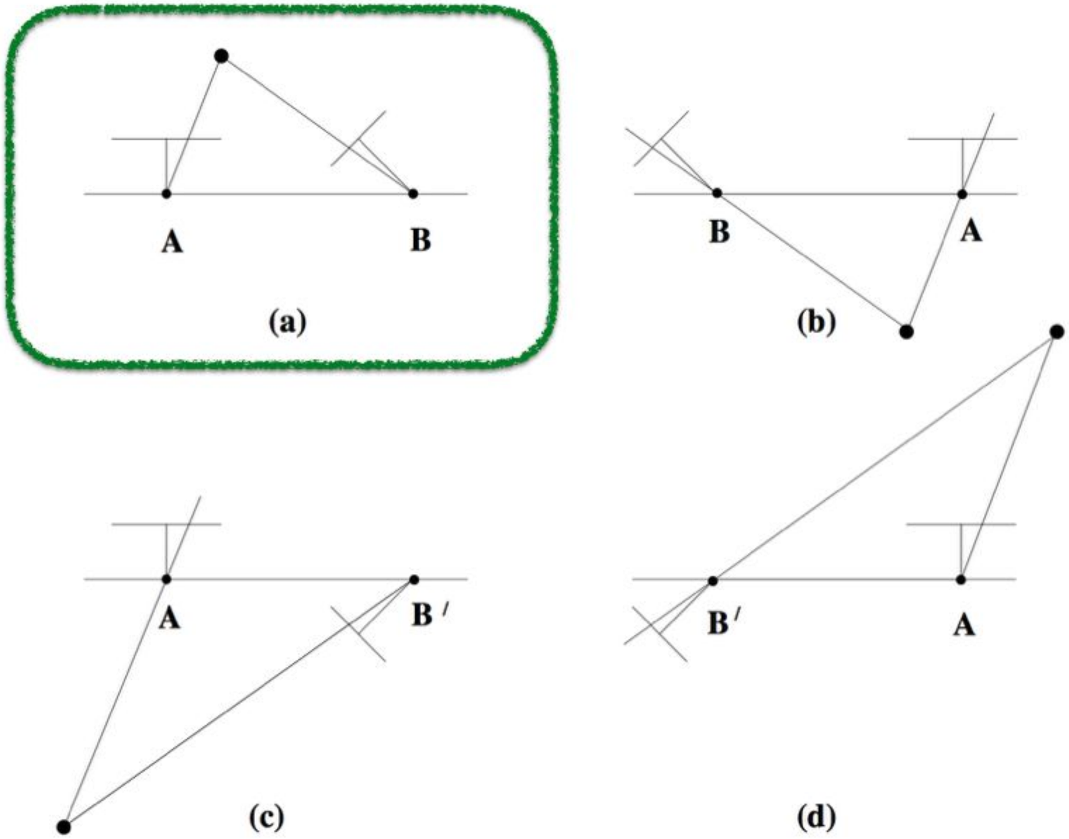
Slide Credit: Kris Kitani

Find the configuration where the points is in front of both cameras



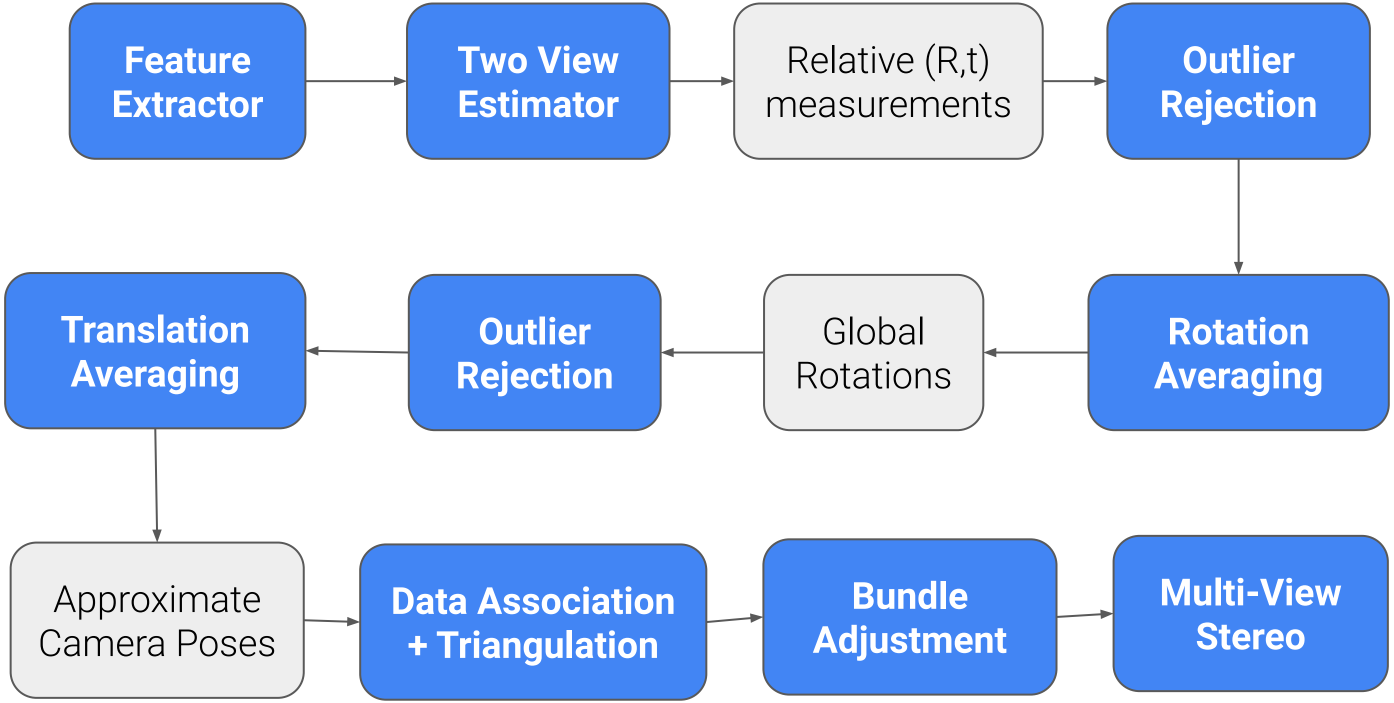
Slide Credit: Kris Kitani

Find the configuration where the points is in front of both cameras



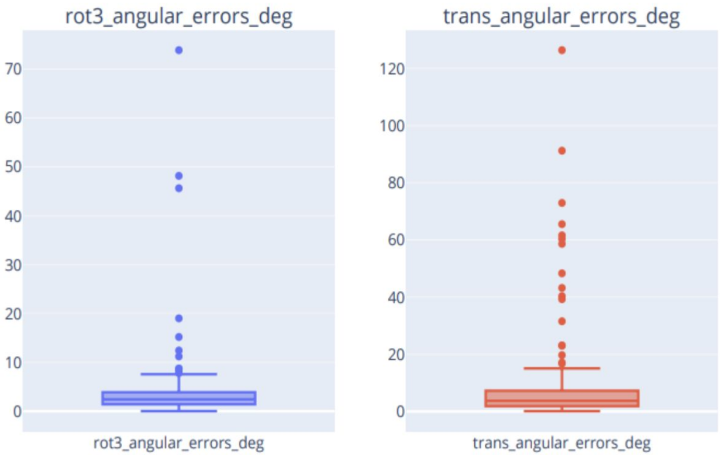
Slide Credit: Kris Kitani

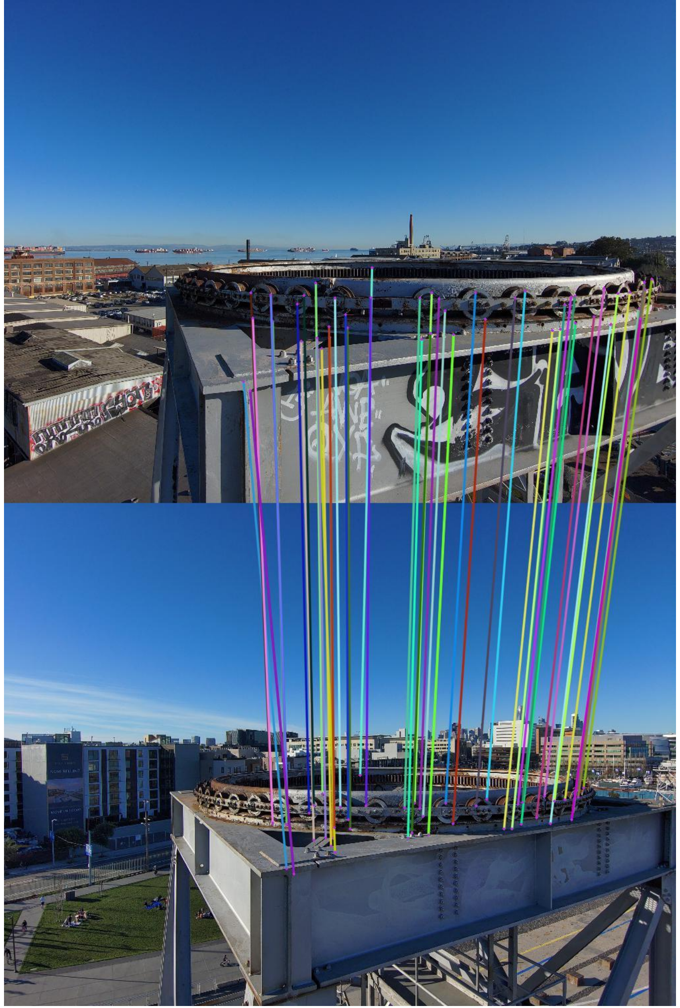
Global SfM Revisited

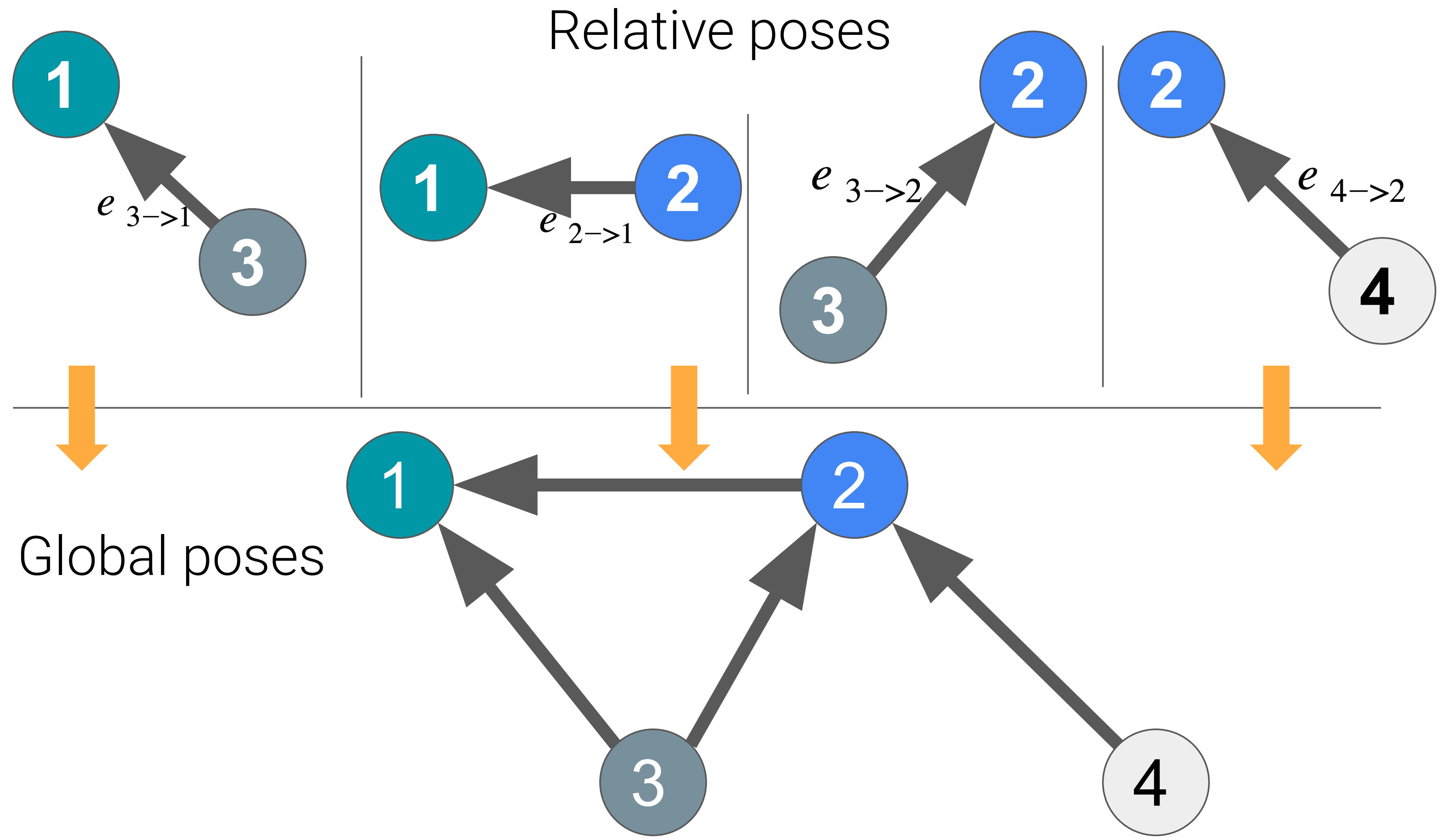


Why is global SfM hard?

The noisy front-end! (Without noise, would be nearly exact)



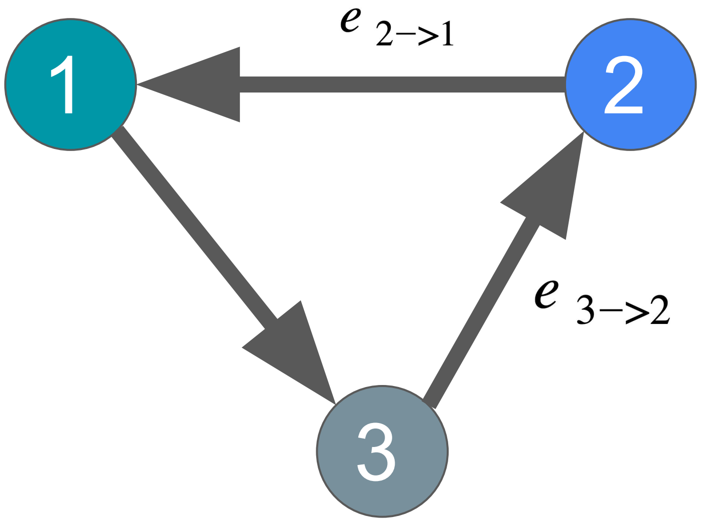




Relative poses
1
3
e 3->1
1
2
e 2->1
2
3
e 3->2
2
4
e 4->2
Global poses
1
2
3
4

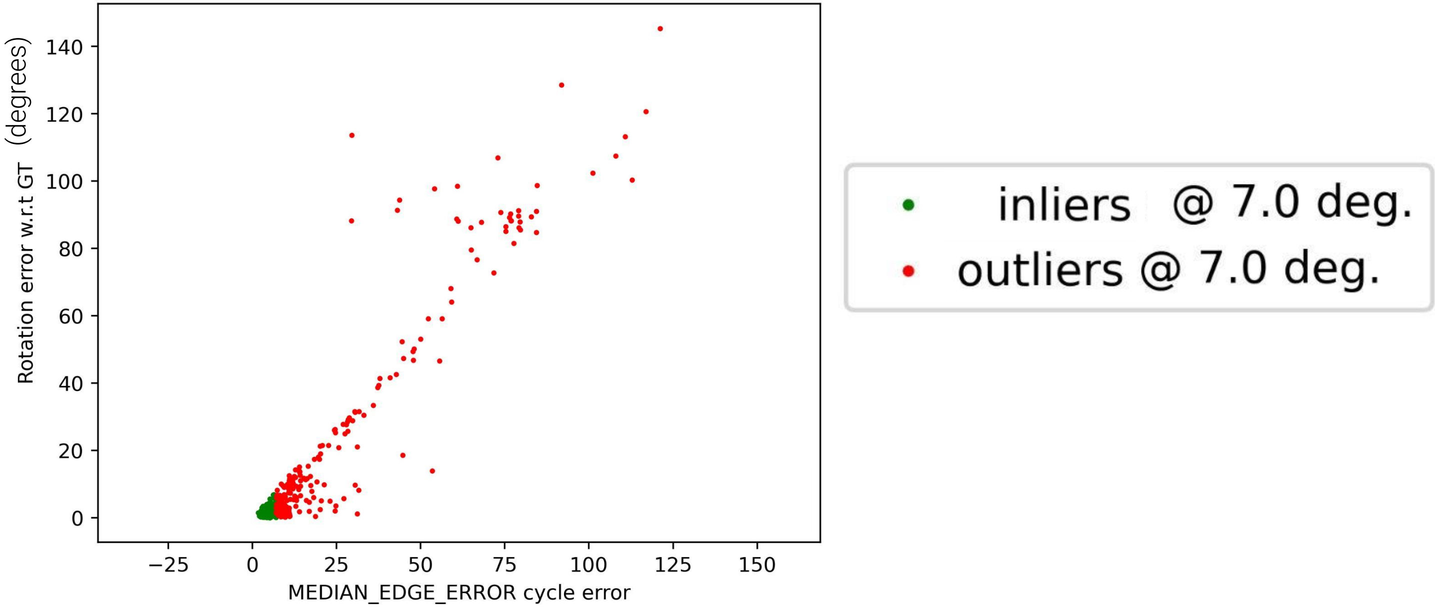
Rotation Cycle Consistency:

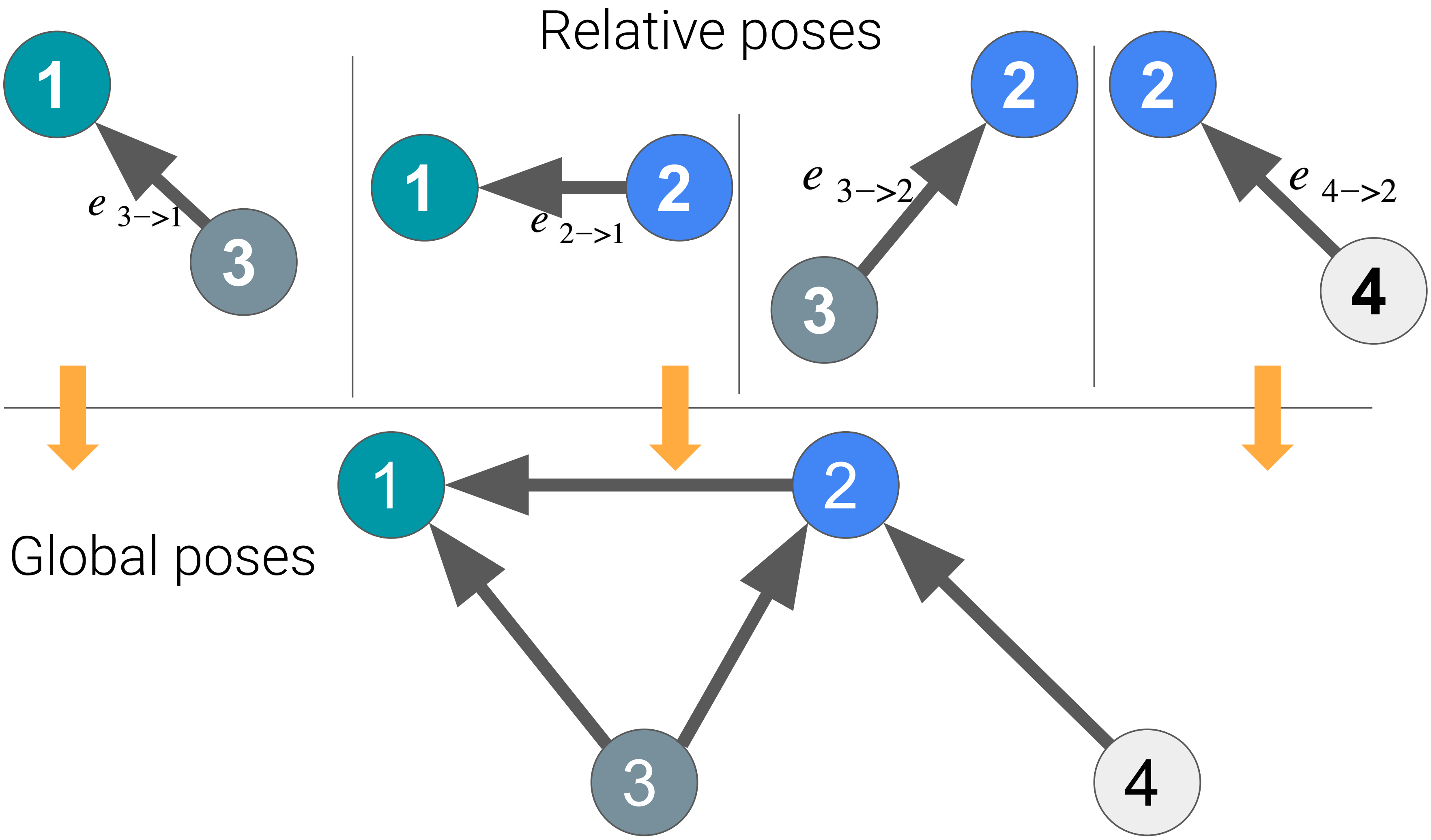
$$\mathbf{R}_L = \mathbf{R}_{e_{|L|}} \times \cdots \times \mathbf{R}_{e_1} = I, \quad e_i \in L$$



Rotation Cycle Consistency:

$$\mathbf{R}_L = \mathbf{R}_{e_{|L|}} \times \cdots \times \mathbf{R}_{e_1} = I, \quad e_i \in L$$




Relative poses
1
3
e 3−>1
1
2
e 2−>1
2
3
e 3−>2
2
4
e 4−>2
Global poses
1
2
3
4

Rotation Averaging

given a collection of rotation matrices

$$\mathbf{R}_1, \ldots, \mathbf{R}_n \in \mathbb{R}^{3\times 3}$$

find the average rotation $\bar{\mathbf{R}}$.

How can we average rotations?

3D rotation matrices do not form a vector space. An easy way to see this is to try to add the following two rotation matrices, I and R, where R is a 180° rotation about the z-axis, `gtsam.Rot3.RzRyRx(x=0, y=0, z=np.deg2rad(180)).matrix()`:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, R = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

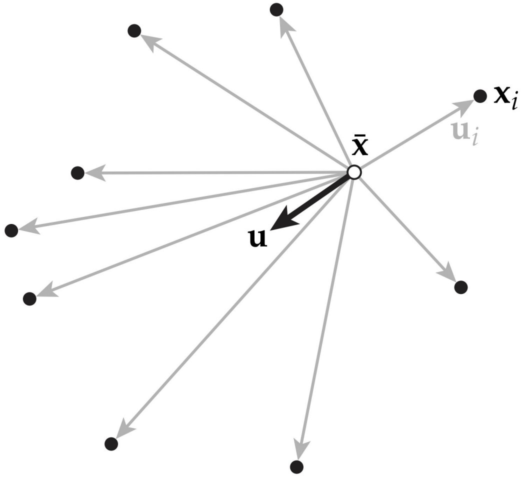
$$I + R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

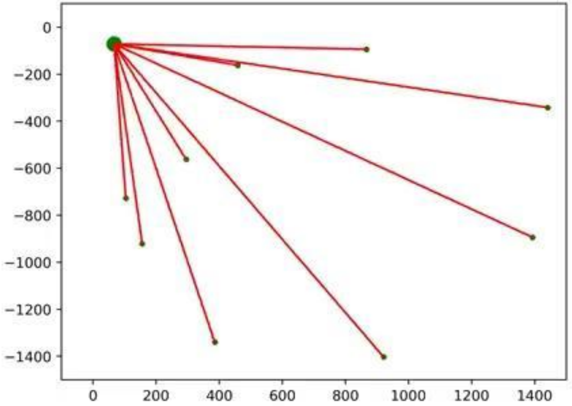
which is not a rotation (it squashes flat the x– and y– components)

Single Rotation Averaging

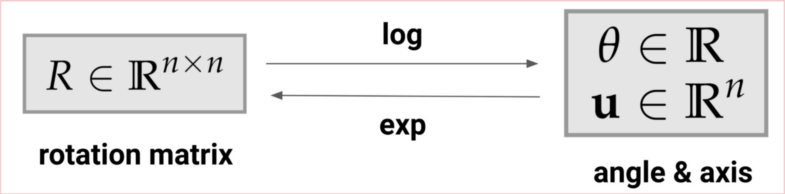
Weiszfeld's algorithm

- Pick an initial guess $\bar{\mathbf{x}} \in \mathbb{R}^2$
- Do
  - (1) $\mathbf{u}_i \leftarrow \mathbf{x}_i - \bar{\mathbf{x}}$
  - (2) $\mathbf{u} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \mathbf{u}_i$
  - (3) $\bar{\mathbf{x}} \leftarrow \bar{\mathbf{x}} + \tau \mathbf{u}$
- While $||\mathbf{u}|| > \epsilon$





Single Rotation Averaging



- Pick an initial guess $\bar{\mathbf{R}} \in \mathbf{R}^{3\times 3}$
- Do
  - (1) $\boldsymbol{\omega}_i \leftarrow \log\left(\mathbf{R}_i \bar{\mathbf{R}}^{-1}\right)$
  - (2) $\boldsymbol{\omega} \leftarrow \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\omega}_i$
  - (3) $\bar{\mathbf{R}} \leftarrow \exp(\tau\,\boldsymbol{\omega})\bar{\mathbf{R}}$
- While $\|\boldsymbol{\omega}\| > \epsilon$

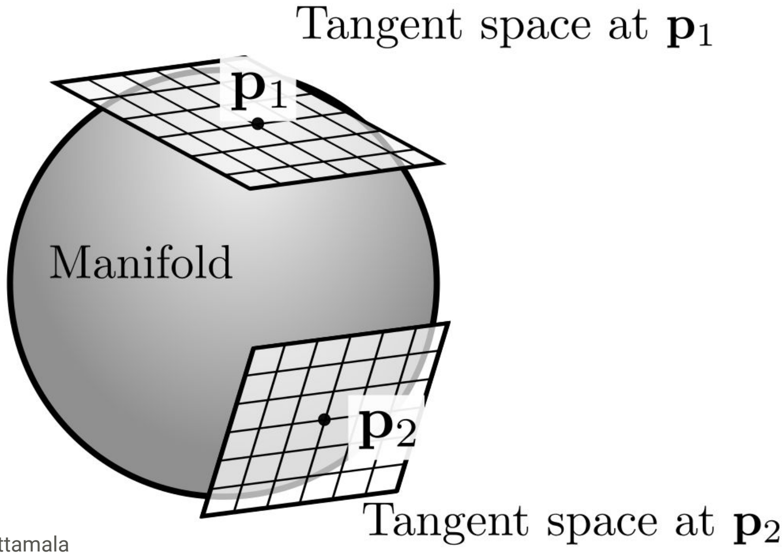


Figure Source: Matias Mattamala

Multiple Rotation Averaging

Same principle, but now we'll solve a least squares problem in the "tangent" space.

Algorithm 1 Lie-Algebraic Relative Rotation Averaging

Input: $\{\mathbf{R}_{ij1}, \cdots, \mathbf{R}_{ijk}\}$ ($|\mathcal{E}|$ relative rotations)
Output: $\mathbf{R}_{global} = \{\mathbf{R}_1, \cdots, \mathbf{R}_N\}$ ($|\mathcal{V}|$ absolute rotations)
Initialisation: $\mathbf{R}_{global}$ to an initial guess

while $||\Delta\boldsymbol{\omega}_{rel}|| < \epsilon$ **do**

1. $\Delta\mathbf{R}_{ij} = \mathbf{R}_j{}^{-1}\mathbf{R}_{ij}\mathbf{R}_i$
2. $\Delta\boldsymbol{\omega}_{ij} = \log(\Delta\mathbf{R}_{ij})$
3. Solve $\mathbf{A}\Delta\boldsymbol{\omega}_{global} = \Delta\boldsymbol{\omega}_{rel}$
4. $\forall k \in [1, N], \mathbf{R}_k = \mathbf{R}_k exp(\Delta\boldsymbol{\omega}_k)$

end while

See [Govindu, CVPR 04](), Chatterjee ICCV 2013

Translation Averaging

Given camera rotations in a global frame, and pairwise translation directions, can we recover the position of each camera (translation in a global frame)?

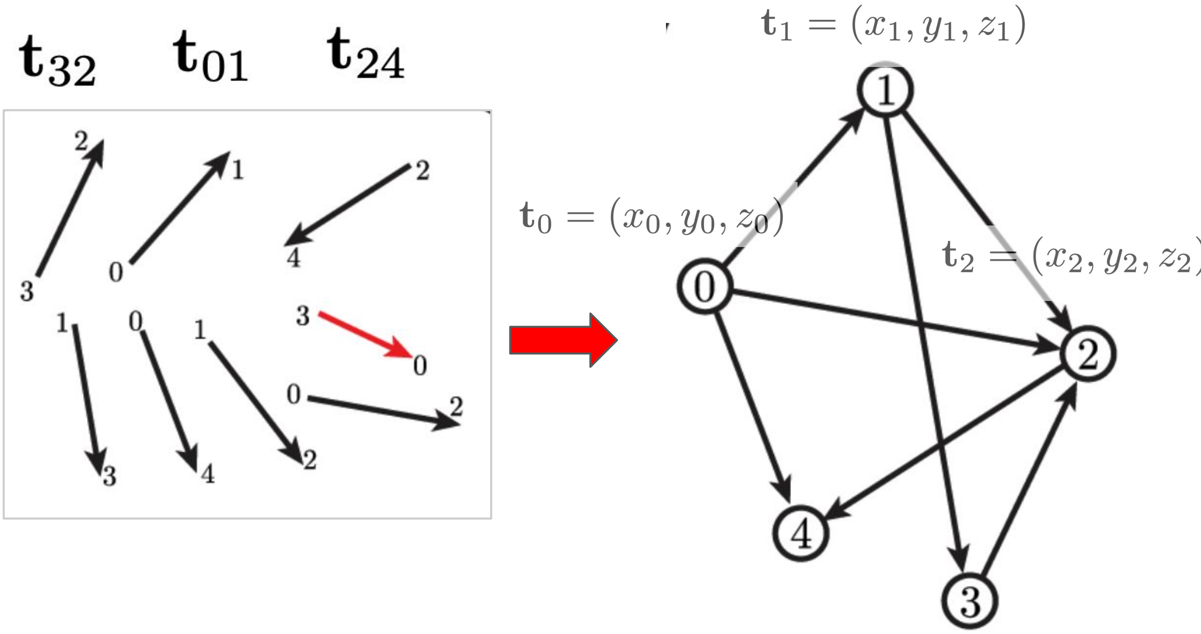
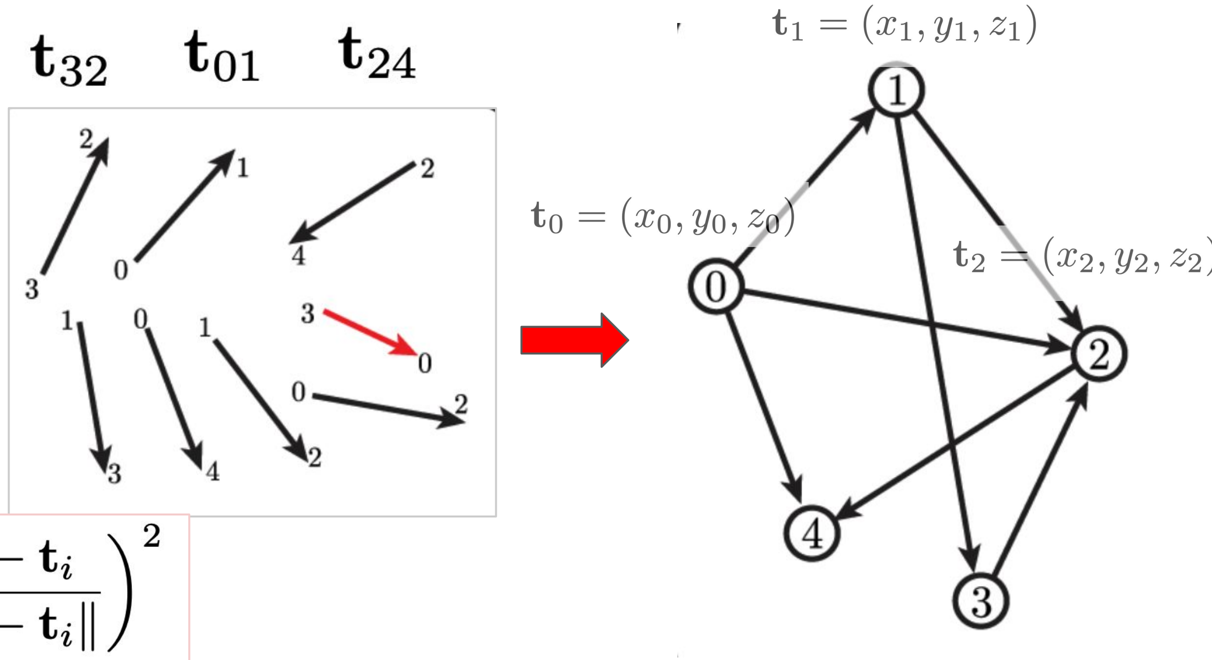


Figure source: Kyle Wilson and Noah Snavely, Robust Global Translations with 1DSfM. ECCV '14.

Translation Averaging



$$err_{ch}(\mathcal{T}) = \sum_{(i,j)\in E} d_{ch}\left(\hat{\mathbf{t}}_{ij}, \frac{\mathbf{t}_j - \mathbf{t}_i}{\|\mathbf{t}_j - \mathbf{t}_i\|}\right)^2$$

$$d_{ch}(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2$$

Figure source: Kyle Wilson and Noah Snavely, Robust Global Translations with 1DSfM. ECCV ‘14.

Data Association

Find connected components in keypoint match graph -> Union Find Algorithm

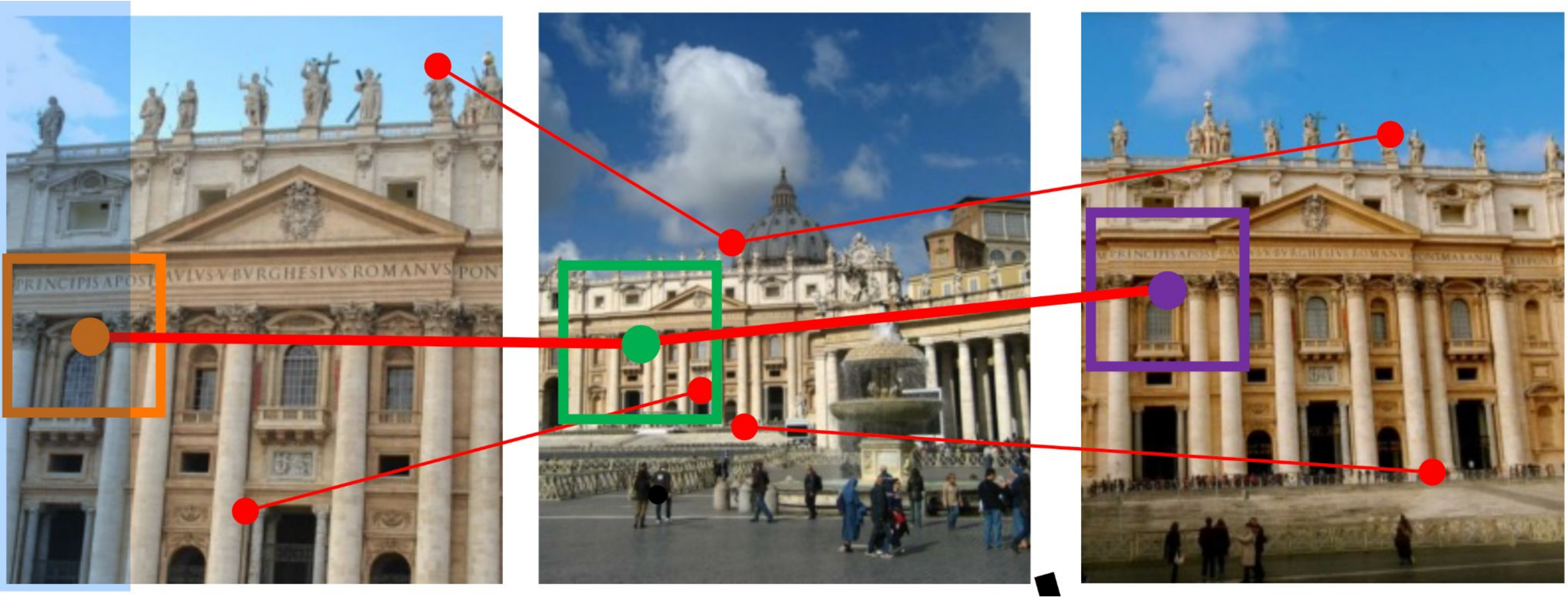
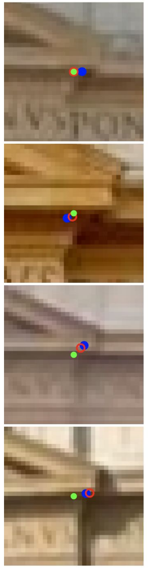


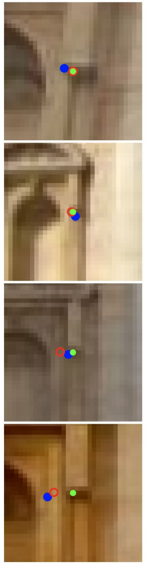
Figure Source: Lindenberger et al., Pixel-Perfect Structure-from-Motion with Featuremetric Refinement, ICCV 21

Data Association: obtain point “tracks”

Track 1



Track 2



Track 3

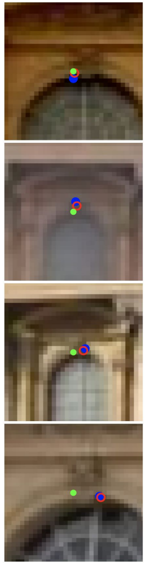
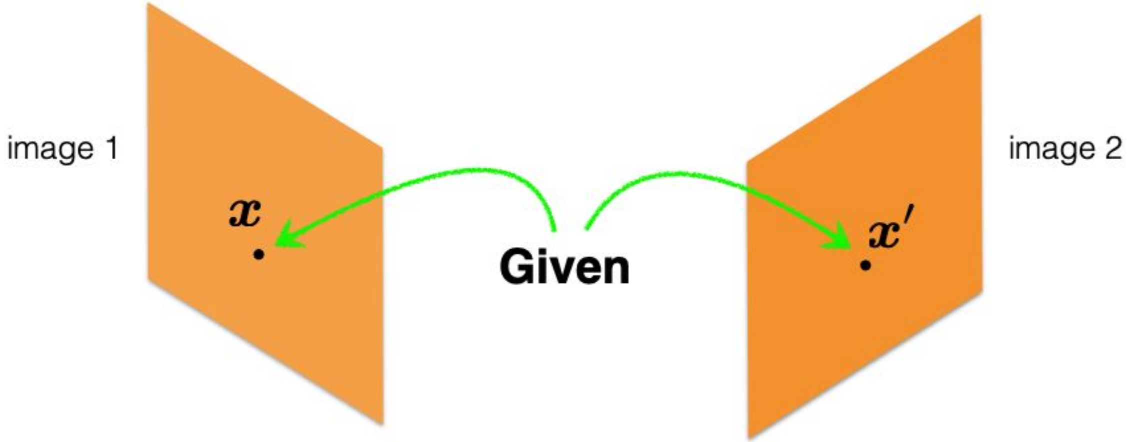


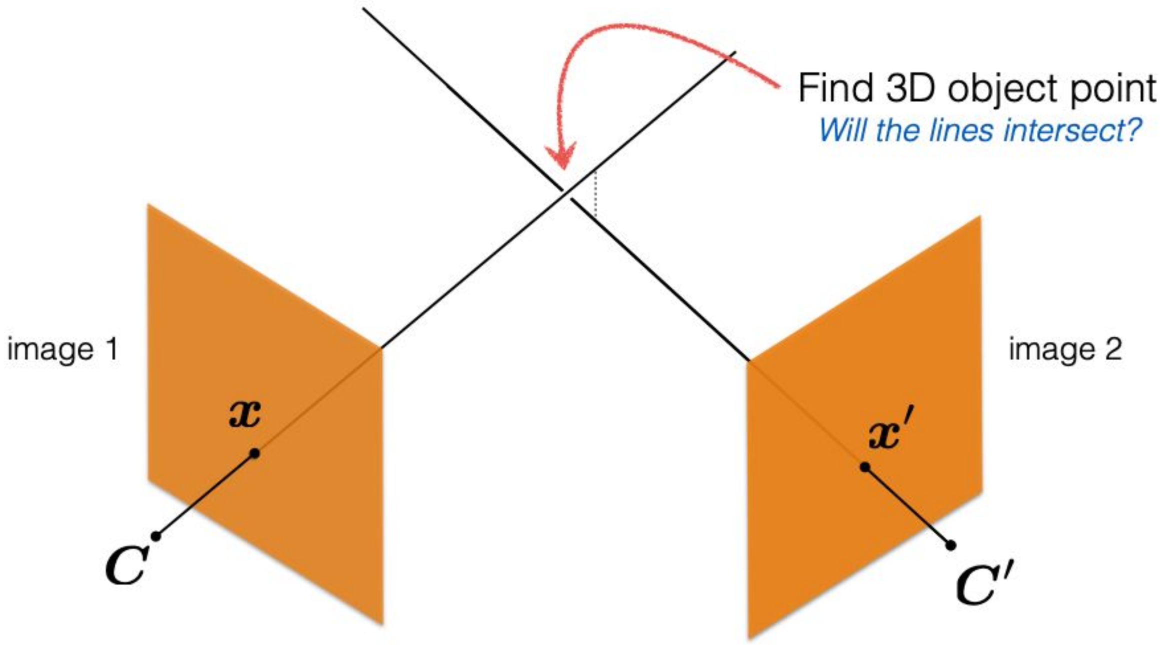
Figure Source: Lindenberger et al., ICCV 21

Triangulation



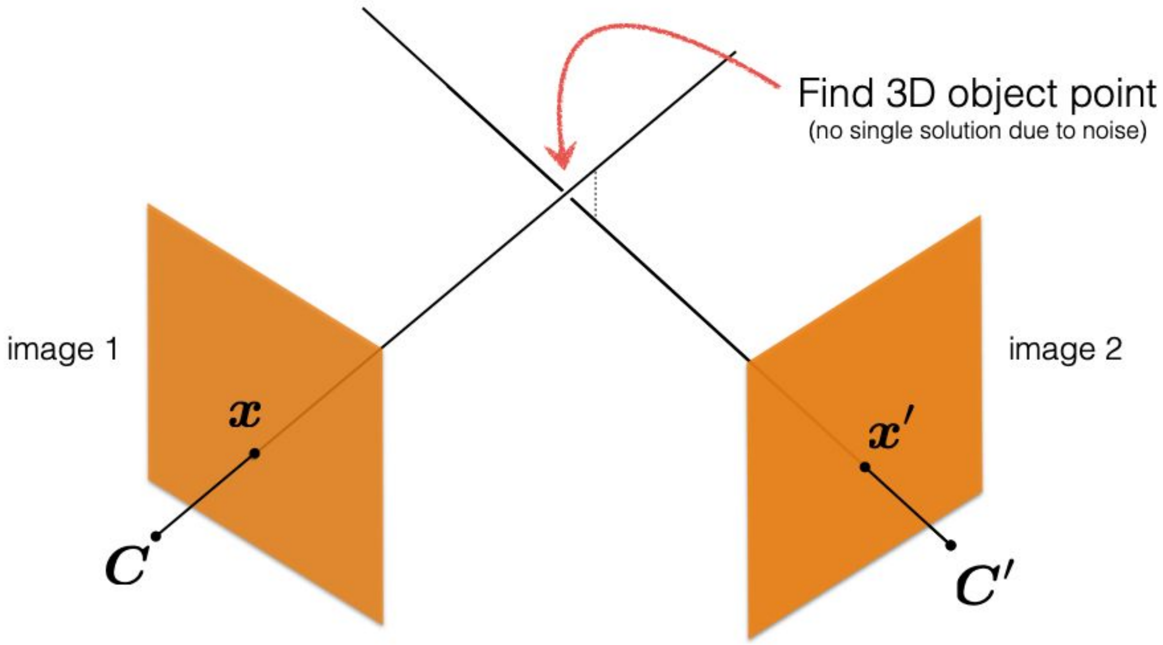
Slide Credit: Kris Kitani

Triangulation



Slide Credit: Kris Kitani

Triangulation



Slide Credit: Kris Kitani

Triangulation

Given a set of (noisy) matched points

$$\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$$

and camera matrices

$$\mathbf{P}, \mathbf{P}'$$

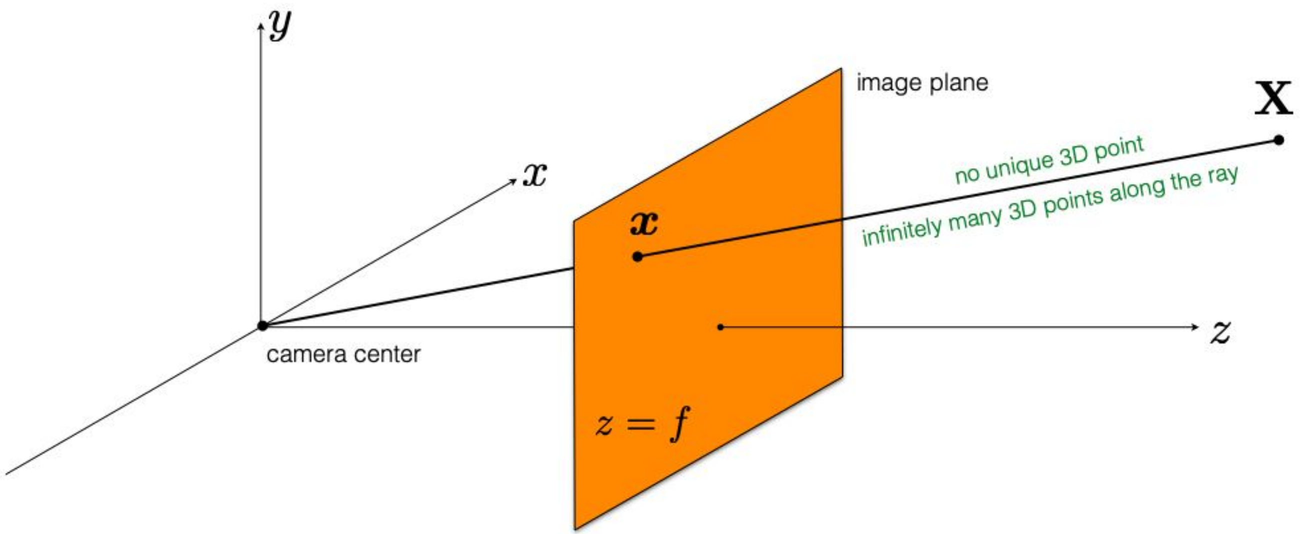
Estimate the 3D point

$$\mathbf{X}$$

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

known known

*Can we compute **X** from a single correspondence **x**?*



Slide Credit: Kris Kitani

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

known known

*Can we compute **X** from <u>two</u> correspondences **x** and **x'**?*

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

known known

*Can we compute **X** from <u>two</u> correspondences **x** and **x'**?*

yes if perfect measurements

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

known known

Can we compute ***X*** *from* *<u>two</u>* *correspondences* ***x*** *and* ***x'****?*

yes if perfect measurements

There will not be a point that satisfies both constraints because the measurements are usually noisy

$$\mathbf{x}' = \mathbf{P}'\boldsymbol{X} \qquad \mathbf{x} = \mathbf{P}\boldsymbol{X}$$

Need to find the **best fit**

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

(homogeneous coordinate)

Also, this is a similarity relation because it involves homogeneous coordinates

$$\mathbf{x} = \alpha\mathbf{P}\boldsymbol{X}$$

(inhomogeneous coordinate)

Same ray direction but differs by a scale factor

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

How do we solve for unknowns in a similarity relation?

Slide Credit: Kris Kitani

$$\mathbf{x} = \mathbf{P}\boldsymbol{X}$$

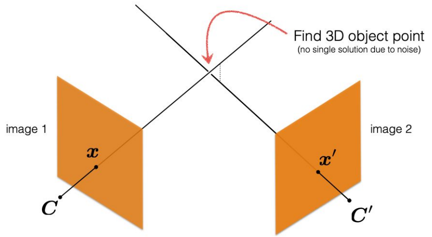
(homogeneous coordinate)

Also, this is a similarity relation because it involves homogeneous coordinates

$$\mathbf{x} = \alpha\mathbf{P}\boldsymbol{X}$$

(inhomogeneous coordinate)

Same ray direction but differs by a scale factor



$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \alpha \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

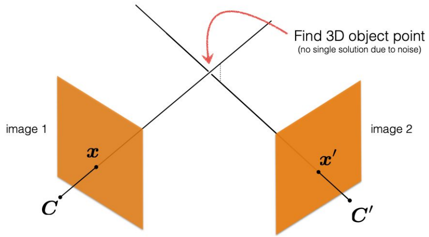
How do we solve for unknowns in a similarity relation?

Direct Linear Transform

Remove scale factor, convert to linear system and solve with SVD.

Triangulation

$$\mathbf{x} = P\mathbf{X}$$
$$\mathbf{x}' = P'\mathbf{X}$$
$$\mathbf{x}'' = P''\mathbf{X}$$
$$\vdots$$



We can use a cross product to get 3 equations for each measurement (2d image point):

$$\mathbf{x} \times \mathbf{x} = \mathbf{x} \times (P\mathbf{X})$$

$$\begin{bmatrix} 0 & -1 & y \\ 1 & 0 & -x \\ -y & x & 0 \end{bmatrix} \mathbf{x} = \mathbf{x} \times P\mathbf{X}$$

$$\begin{bmatrix} 0 & -1 & y \\ 1 & 0 & -x \\ -y & x & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{x} \times P\mathbf{X}$$

$$0 = \mathbf{x} \times P\mathbf{X}$$

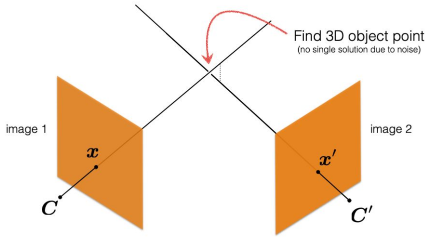
$$0 = \begin{bmatrix} 0 & -1 & y \\ 1 & 0 & -x \\ -y & x & 0 \end{bmatrix} \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} \mathbf{X}$$

Triangulation

You can see above that a linear of combination of the rows of P is being formed. Following Hartley and Zisserman, let $\mathbf{p}_i^T$ represent the i'th row of P.

$$0 = \begin{bmatrix} 0 & -1 & y \\ 1 & 0 & -x \\ -y & x & 0 \end{bmatrix} \begin{bmatrix} - & \mathbf{p}^{1T} & - \\ - & \mathbf{p}^{2T} & - \\ - & \mathbf{p}^{3T} & - \end{bmatrix} \mathbf{X}$$

$$y(\mathbf{p}_3^T\mathbf{X}) - (\mathbf{p}_2^T\mathbf{X}) = 0$$
$$\mathbf{p}_1^T\mathbf{X} - x(\mathbf{p}_3^T\mathbf{X}) = 0$$
$$x(\mathbf{p}_2^T\mathbf{X}) - y(\mathbf{p}_1^T\mathbf{X}) = 0$$



give three equations for each image point, of which two are linearly independent – Third line is a linear combination of the first and second lines. (x times the first line plus y times the second line) – See [9].

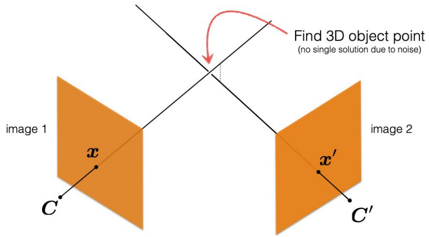
Since we can multiply both sides of any equation by -1, we will often see the second constraint written as

$$\mathbf{p}_1^T\mathbf{X} - x(\mathbf{p}_3^T\mathbf{X}) = 0$$
$$(-1)\mathbf{p}_1^T\mathbf{X} - (-1)x(\mathbf{p}_3^T\mathbf{X}) = 0 * (-1)$$
$$x(\mathbf{p}_3^T\mathbf{X}) - \mathbf{p}_1^T\mathbf{X} = 0$$

Triangulation

We end up with a tall but skinny data matrix A for a homogeneous system of equations:

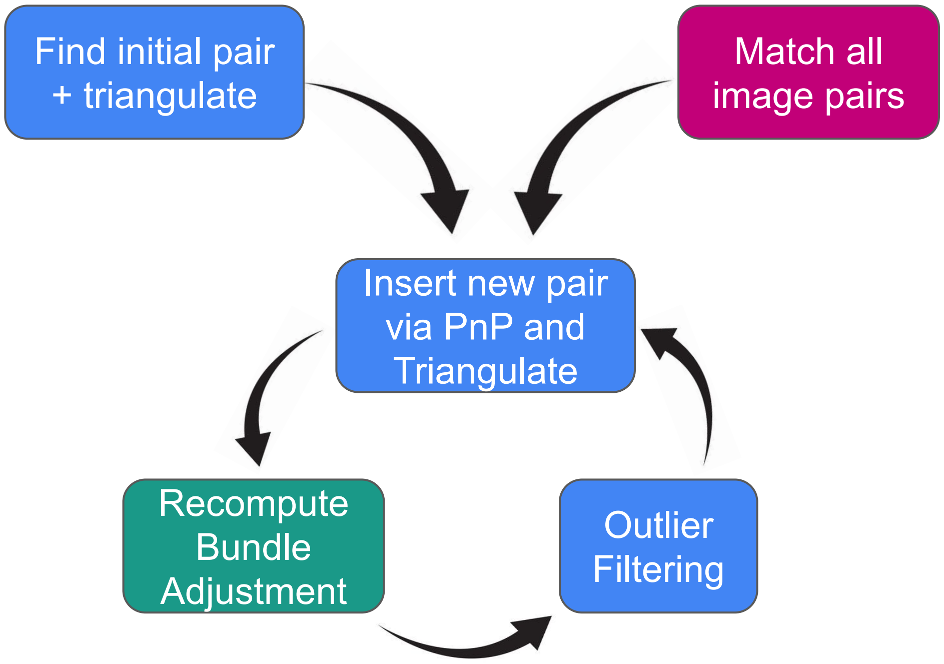
$$A\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{0}$$



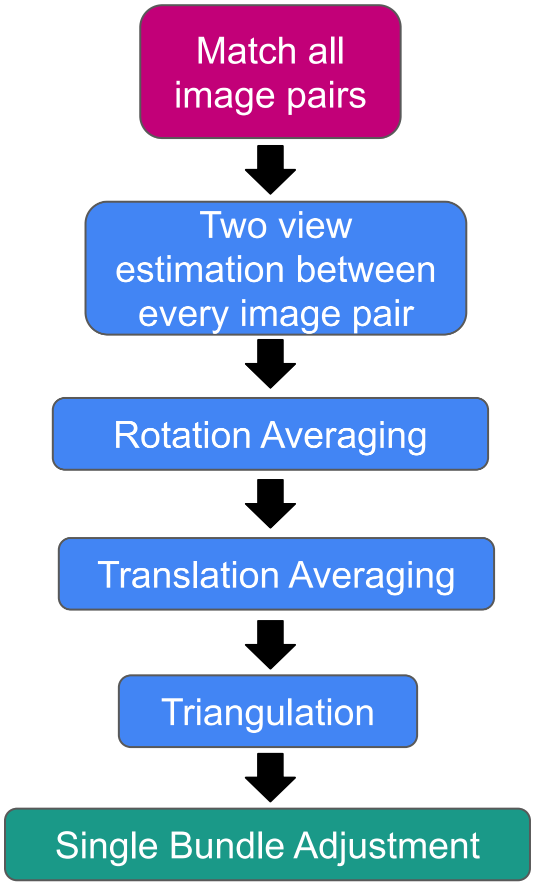
For 2 views, A could be expressed as:

$$A\mathbf{X} = \begin{bmatrix} x(\mathbf{p}_3^T) - \mathbf{p}_1^T \\ y(\mathbf{p}_3^T) - (\mathbf{p}_2^T) \\ x'(\mathbf{p}_3'^T) - \mathbf{p}_1'^T \\ y'(\mathbf{p}_3'^T) - (\mathbf{p}_2' T) \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{0}$$

The code is then simple – since one 2D to 3D point correspondence give you 2 equations, a tall A matrix of shape $(2m, 4)$ is formed for m measurements.

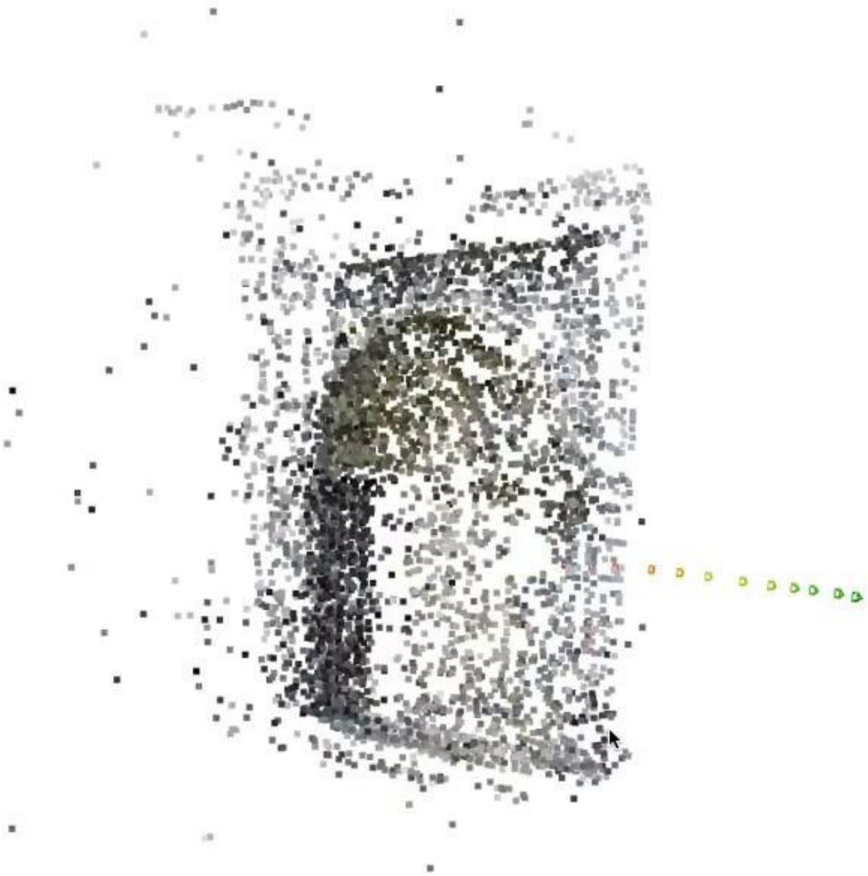


Incremental SfM

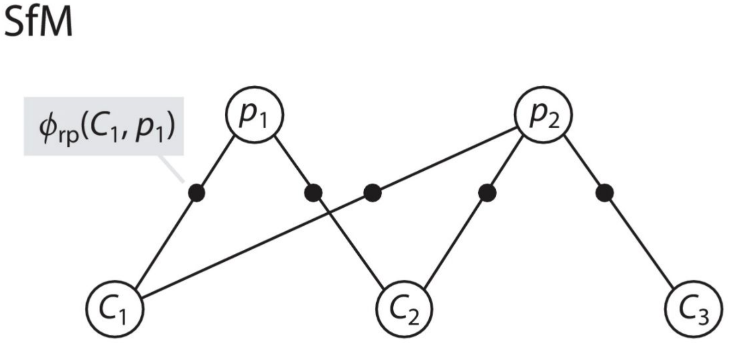


Global SfM

Triangulation Results:
Refinement is Needed!



Bundle Adjustment



$$\min_{\mathbf{X}_1,\mathbf{X}_2,\ldots,M_1,M_2,\ldots} \sum_{i=1}^{m} \sum_{j=1}^{n} \|x_{ij} - Proj(\mathbf{X_j}, M_i)\|^2$$

Figure source: Frank Dellaert, *Factor Graphs: Exploiting Structure in Robotics*

Bundle Adjustment

$$X^{\text{MAP}} = \underset{X}{\operatorname{argmax}} \prod_i \phi_i(X_i).$$

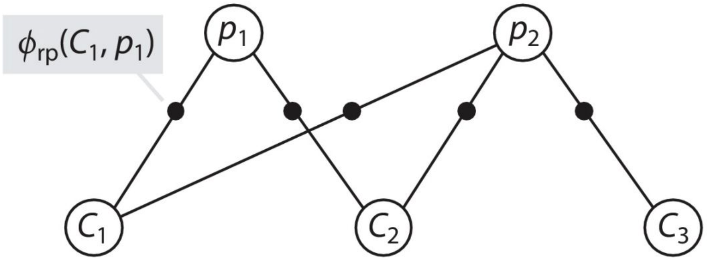
$$\phi_i(X_i) \propto \exp\left\{-\frac{1}{2}\left\|h_i(X_i) - z_i\right\|_{\Sigma_i}^2\right\},$$

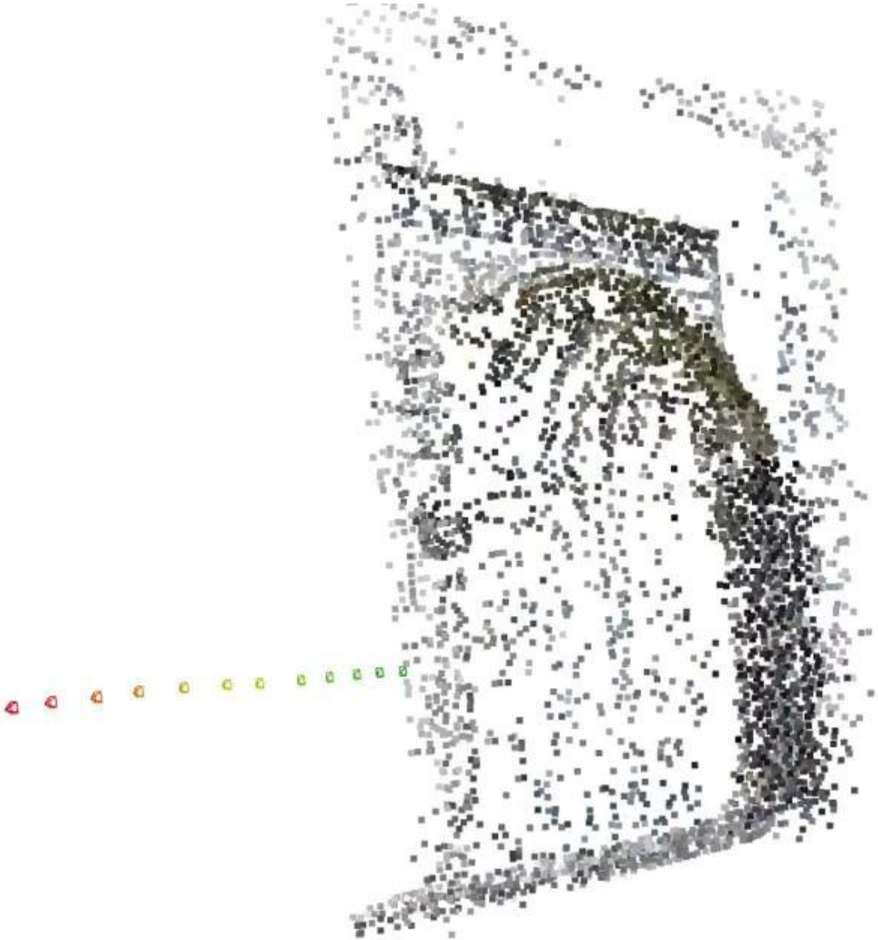
$$X^{\text{MAP}} = \underset{X}{\operatorname{argmin}} \sum_i \left\|h_i(X_i) - z_i\right\|_{\Sigma_i}^2.$$

$$h_i(X_i) = h_i(X_i^0 + \Delta_i) \approx h_i(X_i^0) + H_i\Delta_i,$$

$$\Delta^* = \underset{\Delta}{\operatorname{argmin}} \; \|A\Delta - b\|_2^2,$$

SfM

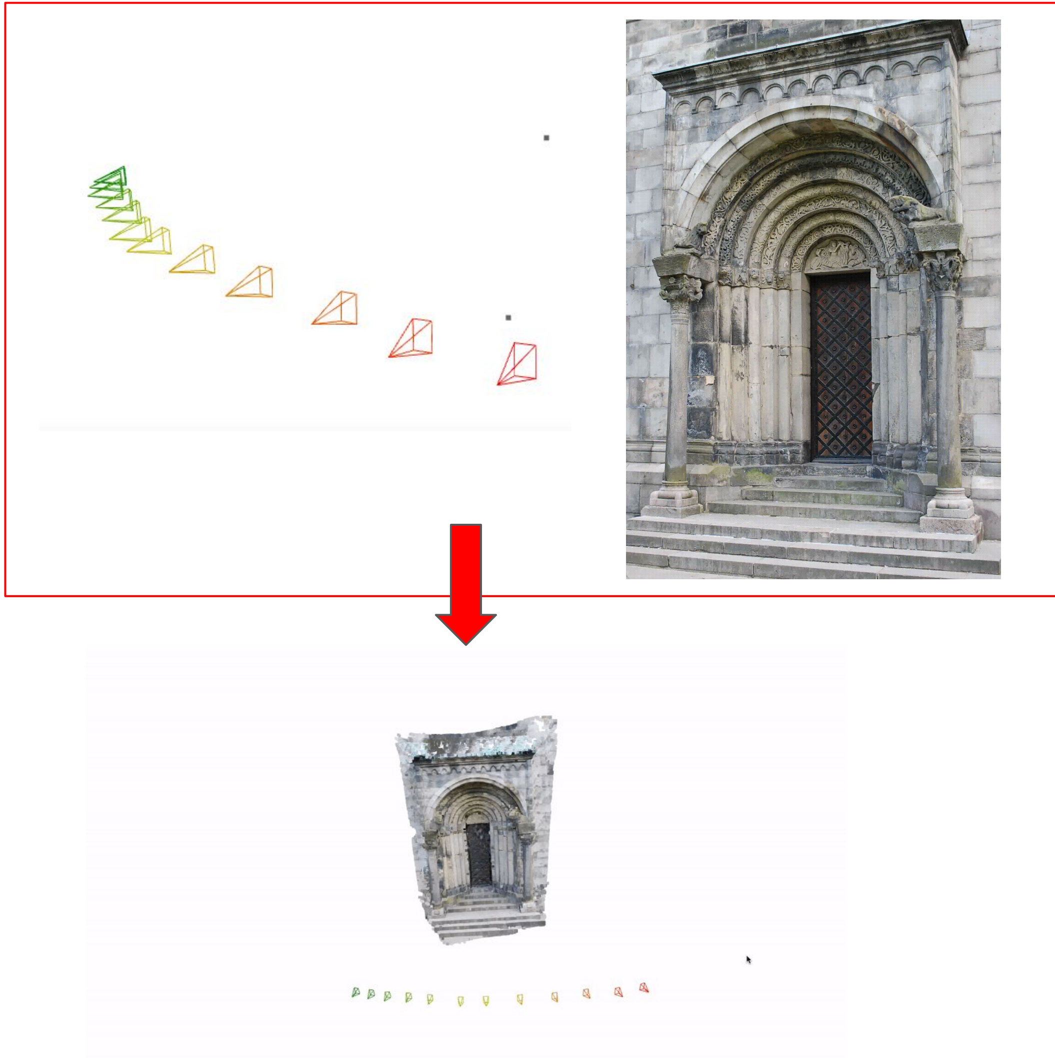




The structure now looks clean,
but is too sparse

Multi-View Stereo (MVS)

- Problem definition: *Given camera extrinsics and intrinsics for multiple cameras, and some possible range of depths, can we obtain dense structure?*



Multi-View Stereo (MVS)

- Problem definition: *Given camera extrinsics and intrinsics for multiple cameras, and some possible range of depths, can we obtain dense structure?*
- Can we use every pixel value, instead of only sparse keypoints?

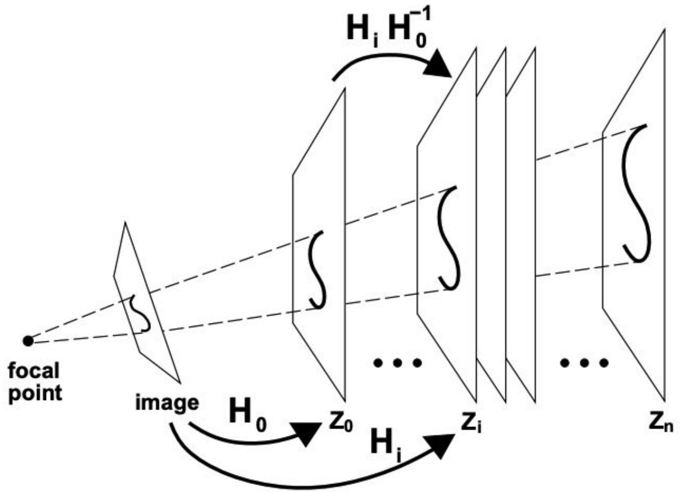


Figure 1: Illustration of the space-sweep method. Features from each image are backprojected onto successive positions $Z = z_i$ of a plane sweeping through space.

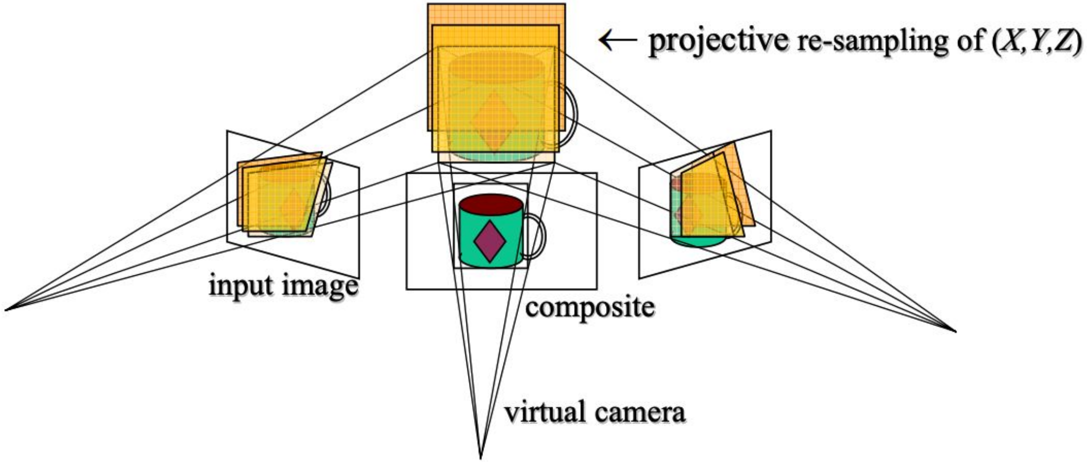
Robert T. Collins. *A Space-Sweep Approach to True Multi-Image Matching*. CVPR 1996

Multi-View Stereo (MVS)

- Problem definition: *Given camera extrinsics and intrinsics for multiple cameras, and some possible range of depths, can we obtain dense structure?*
- Can we use every pixel value, instead of only sparse keypoints?
- Predict depth at every pixel (depth map). Backproject into 3d space.

Plane Sweep Stereo

- Sweep family of planes through volume



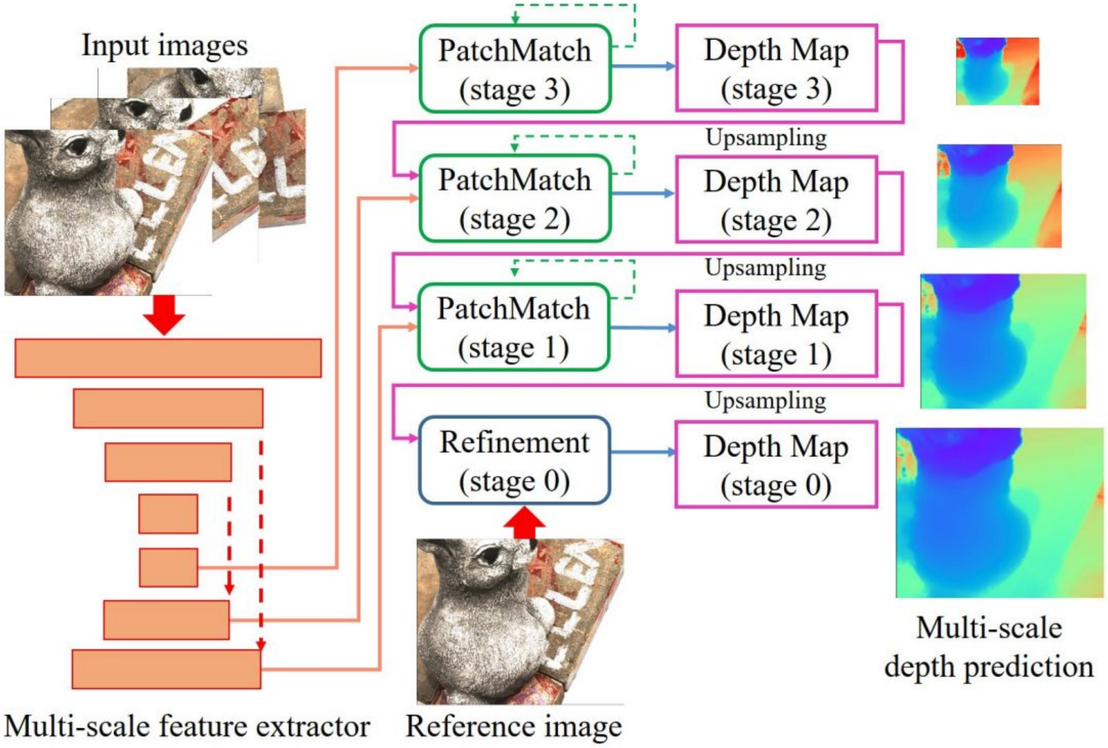
- each plane defines an image ⇒ composite homography

Given two cameras $P = K[I|0]$ and $P' = K'[R|t]$ and a plane $\pi = (n^T, d)^T$

The homography $x' = Hx$ is defined as $H = K'(R - tn^T/d)K^{-1}$

Figure source: [Dan Huttenlocher]

MVS: PatchmatchNet



Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, Marc Pollefeys. PatchmatchNet: Learned Multi-View Patchmatch Stereo. CVPR 2021

Image 1

Image 6

Image 12

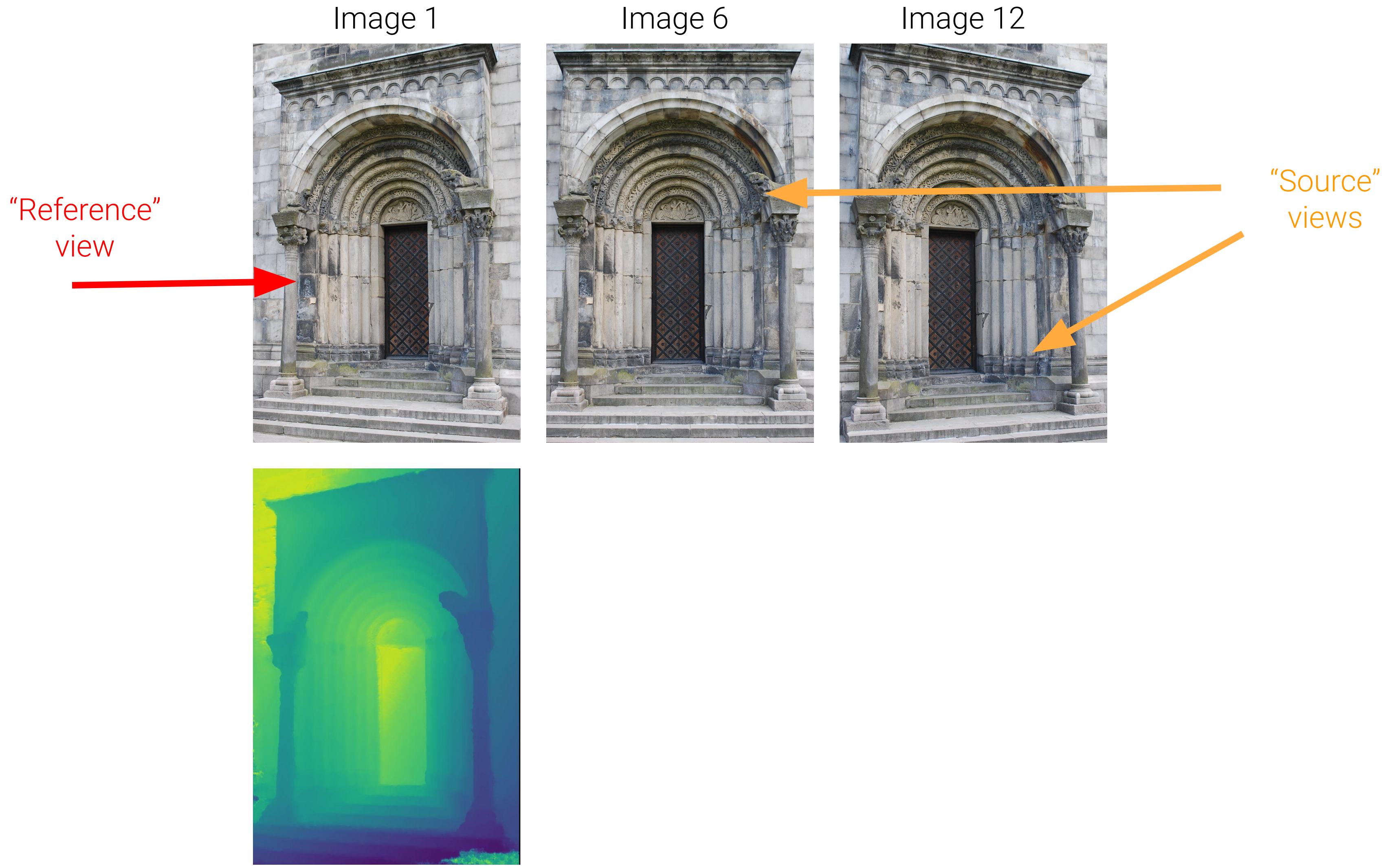


Image 1
Image 6
Image 12
“Reference” view
“Source” views

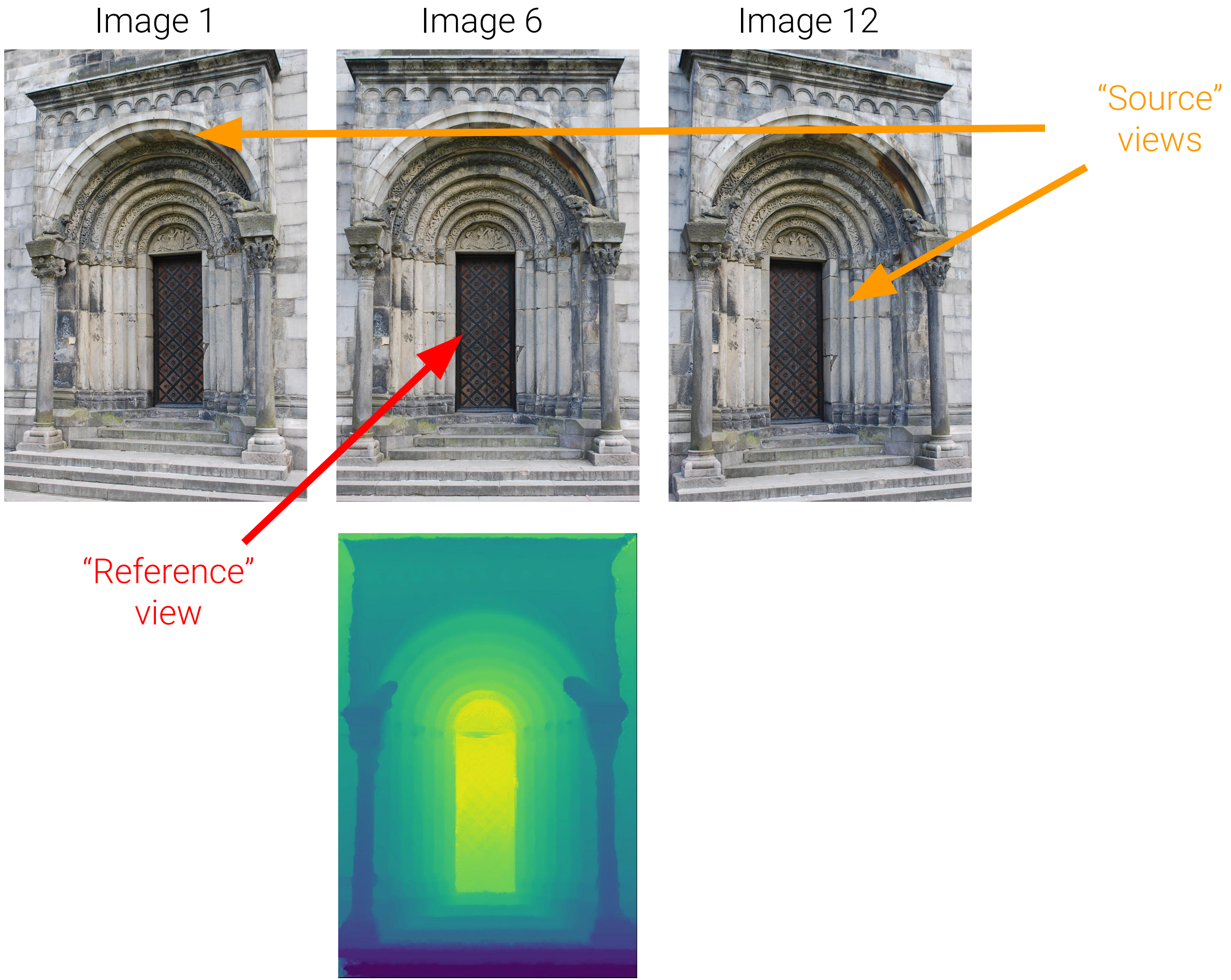


Image 1
Image 6
Image 12
“Source” views
“Reference” view

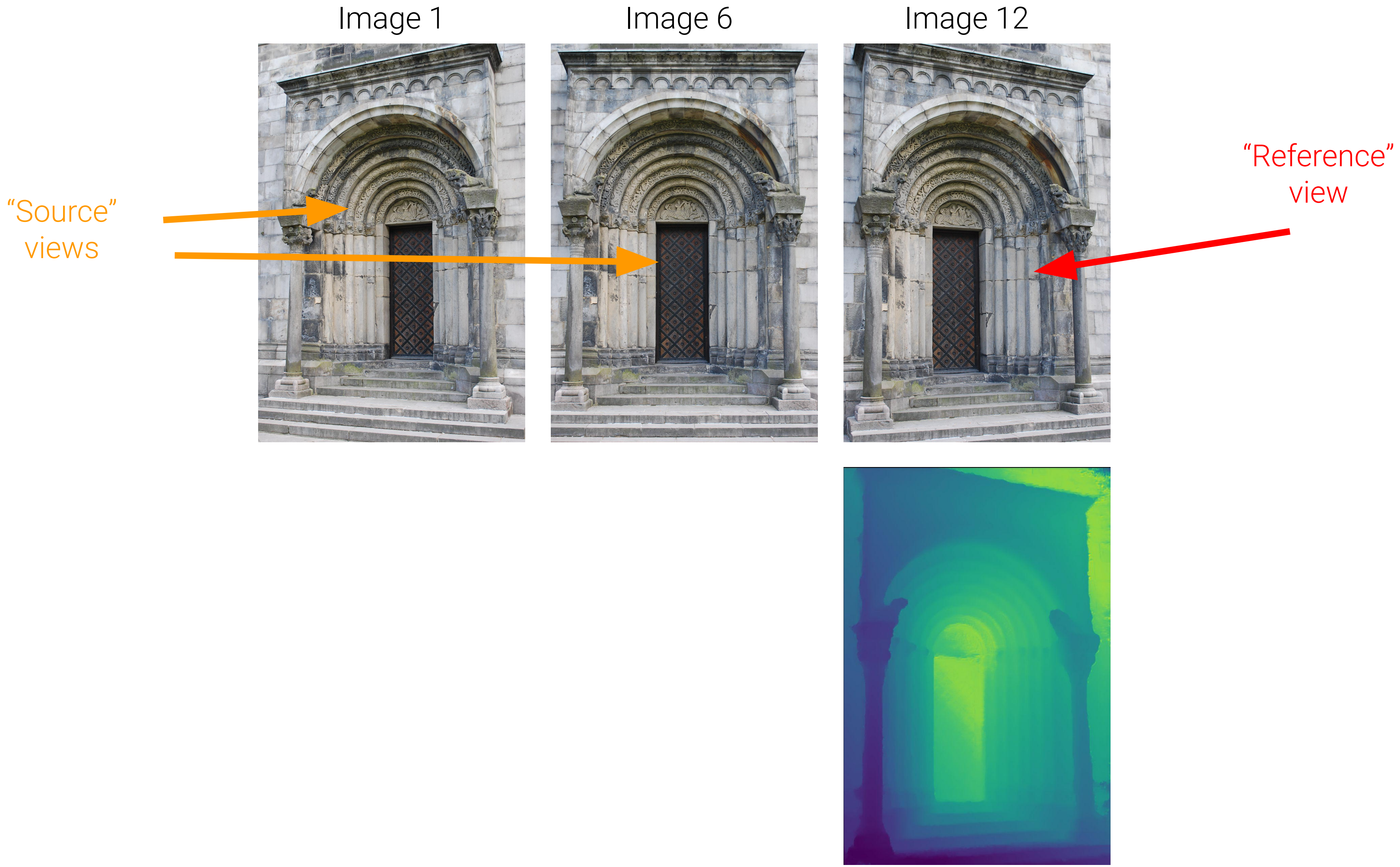
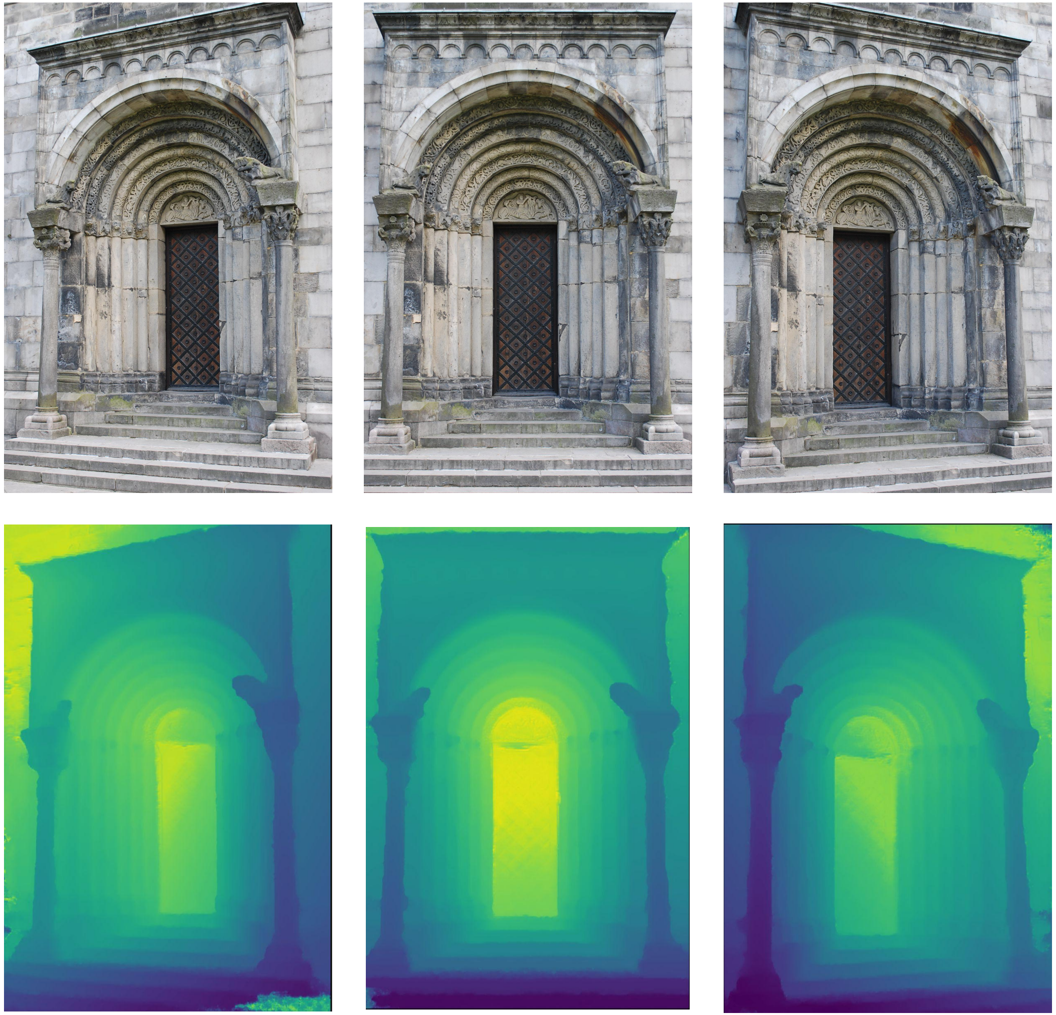


Image 1
Image 6
Image 12
“Source” views
“Reference” view

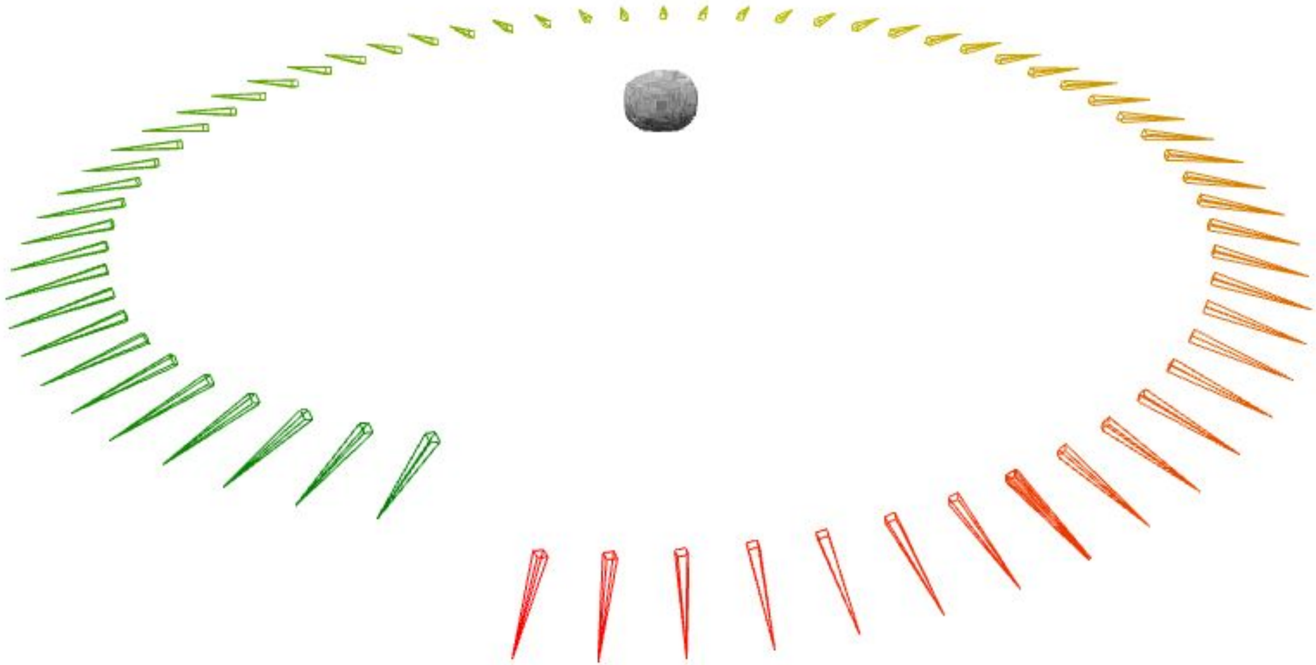
Image 1 Image 6 Image 12



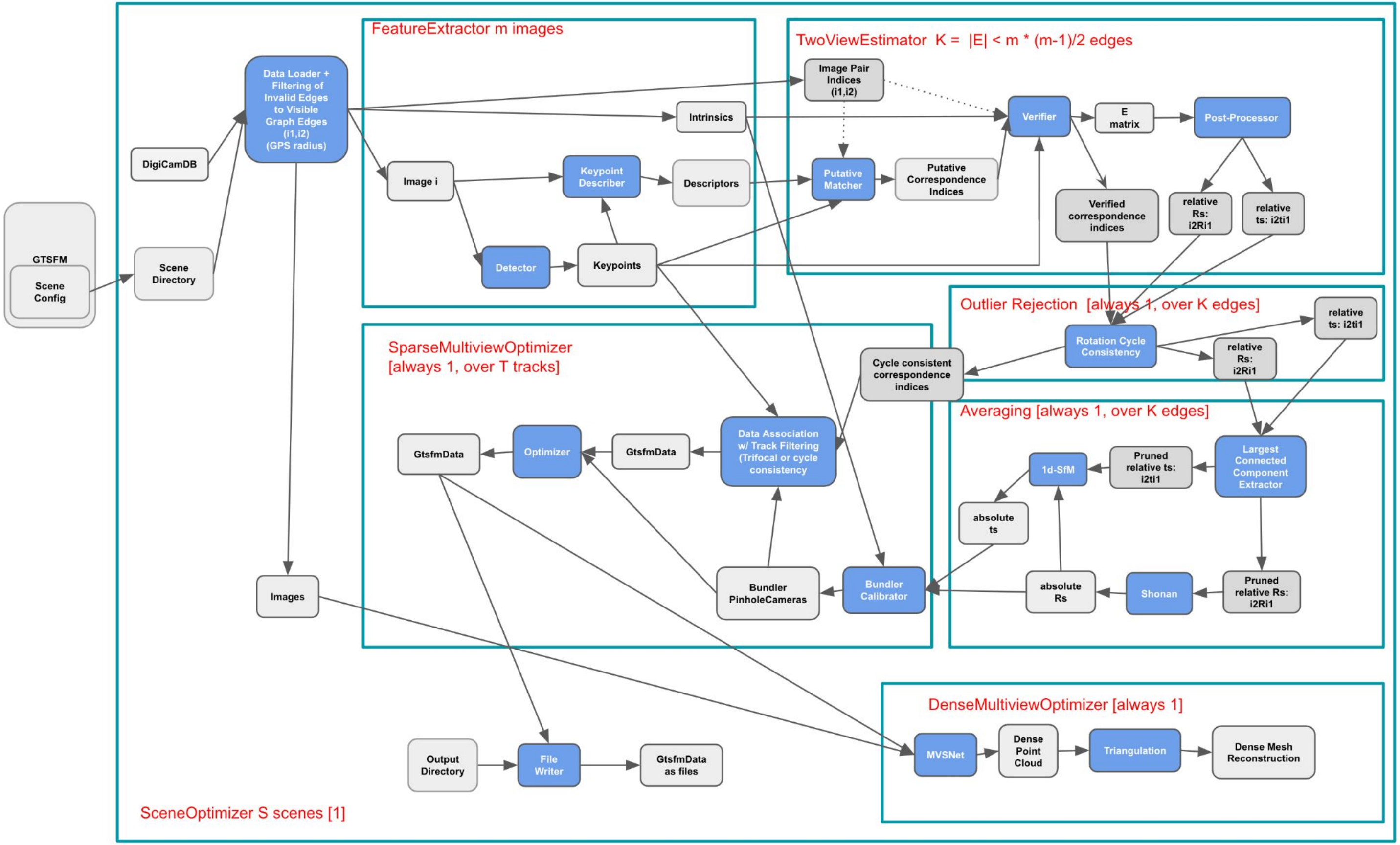
$$\begin{bmatrix} u \cdot d \\ v \cdot d \\ d \end{bmatrix} = K_{ref} * p_c$$

$$p_c = K_{ref}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \cdot d$$

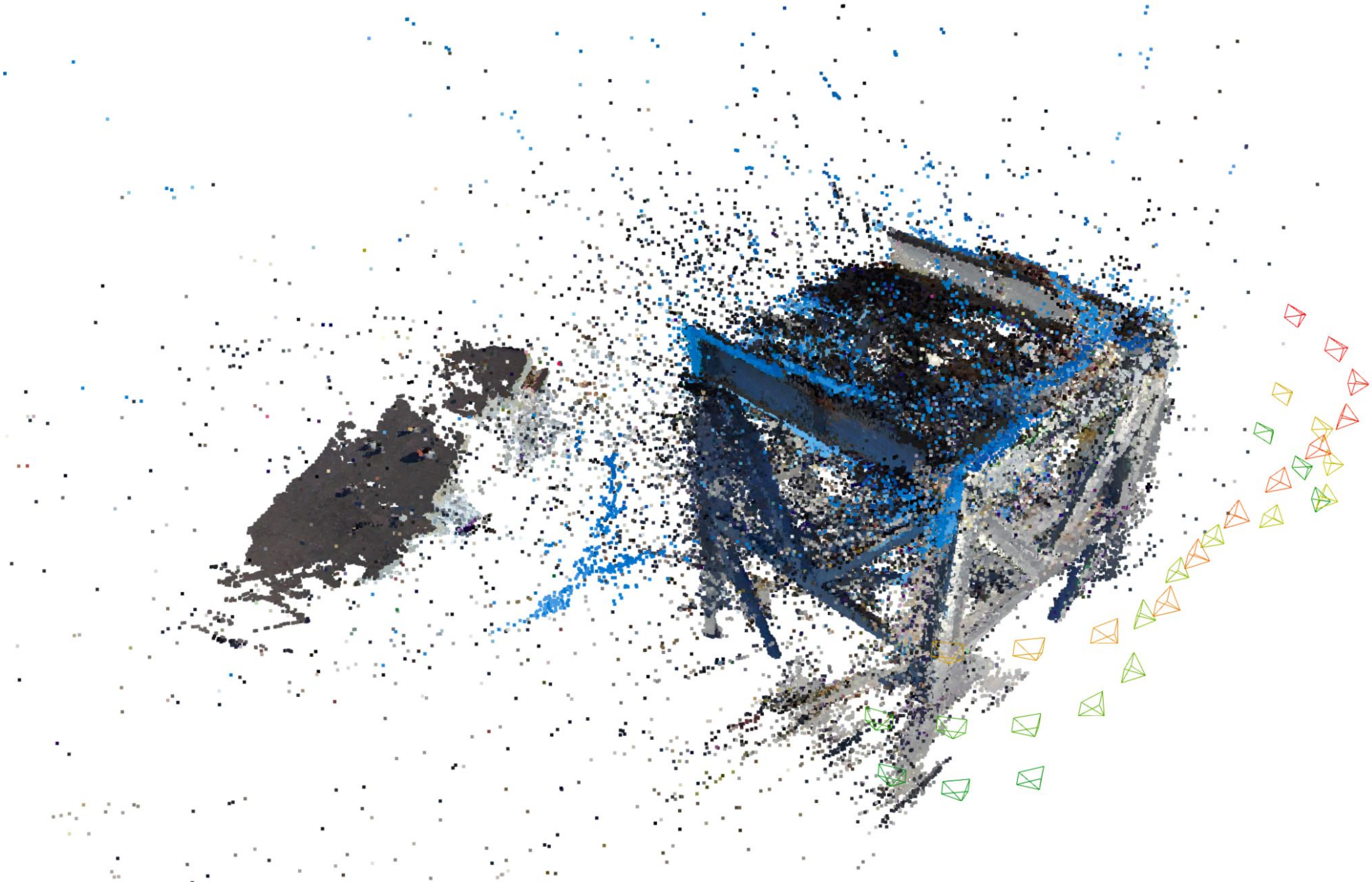
$$p_w = {}^{w}T_c * p_c = {}^{w}T_c * \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



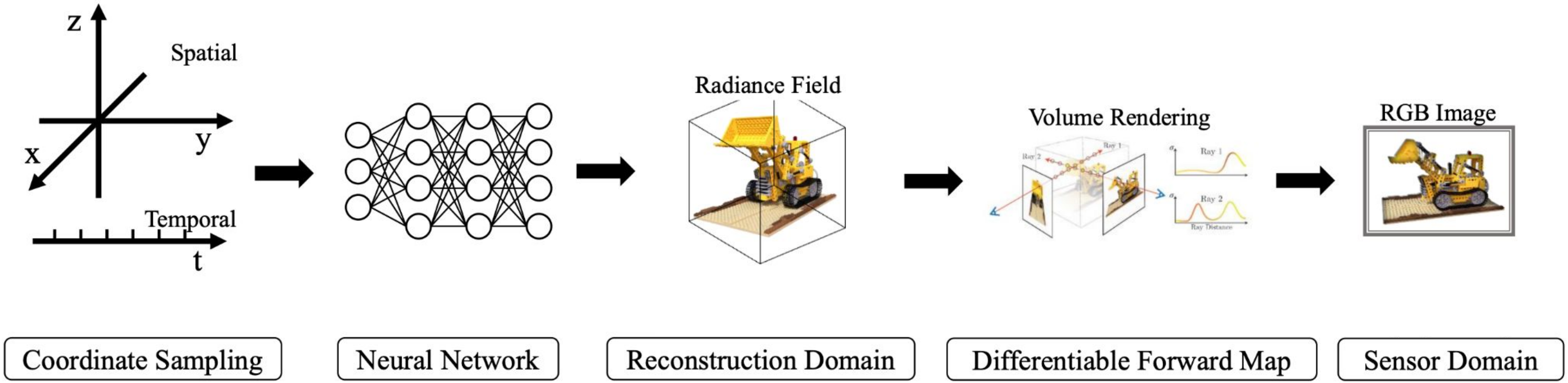
Global SfM Revisited



Challenges: occlusion and large depth ranges



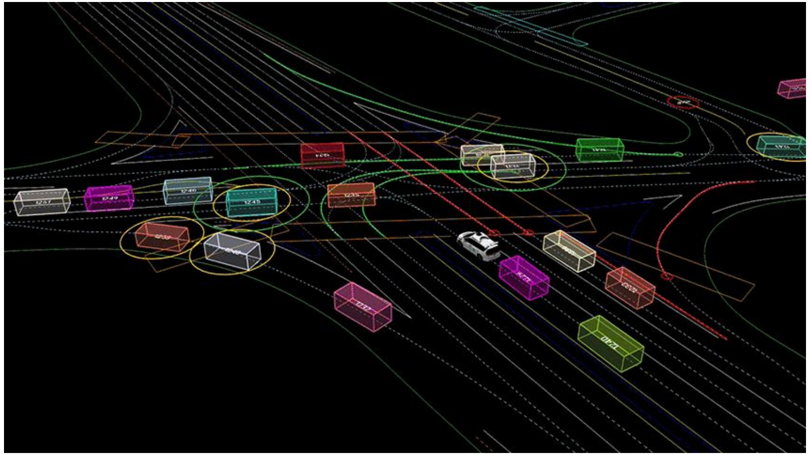
NeRF (Neural Radiance Fields)

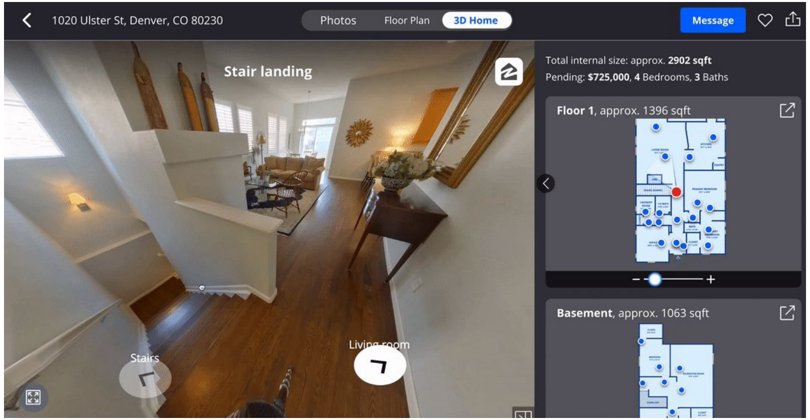



BLOCK-NERF
RESULTS
WAYMO
Google

The future is bright for spatial AI

Spatial AI will revolutionize the way we move and interact with the world.





Collaborators

github.com/borglab/gtsfm