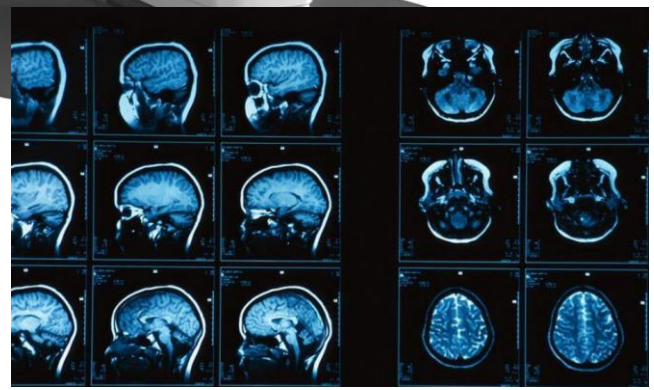
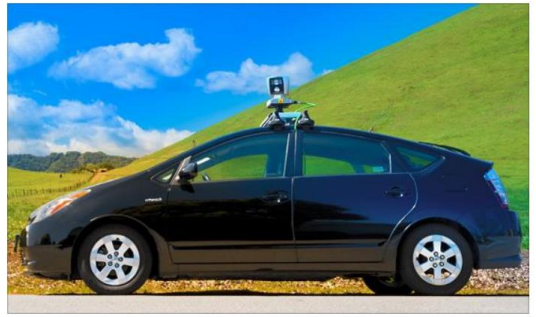
3D Scanning & Motion Capture

Overview of 3D reconstruction methods

Prof. Matthias Nießner

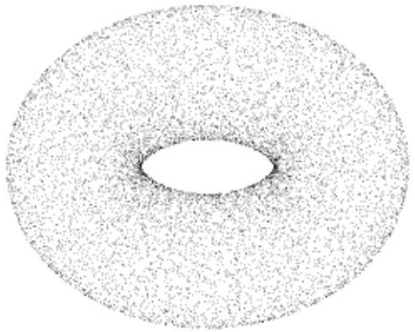
Last Lecture: How to obtain “3D”?



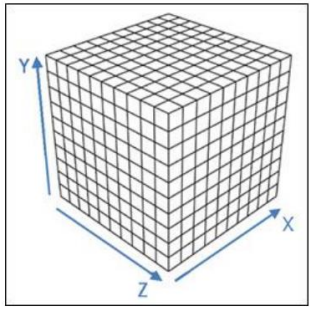


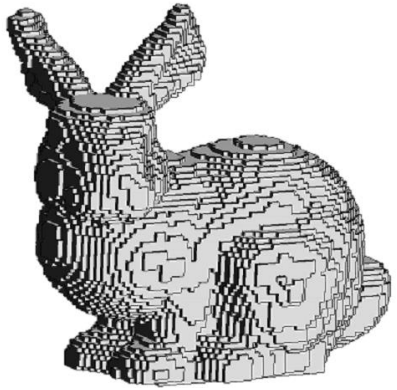
Last Lecture: Surface Representations

- Point Clouds

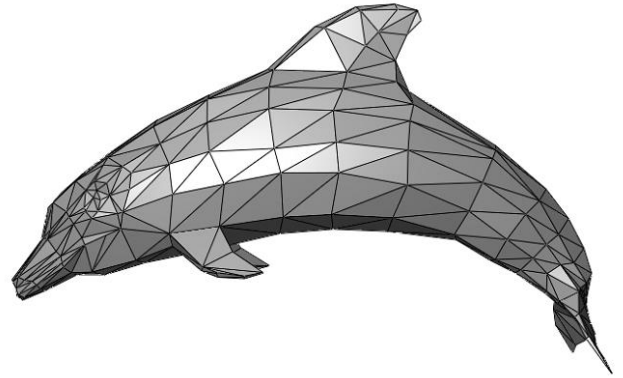


- Voxels

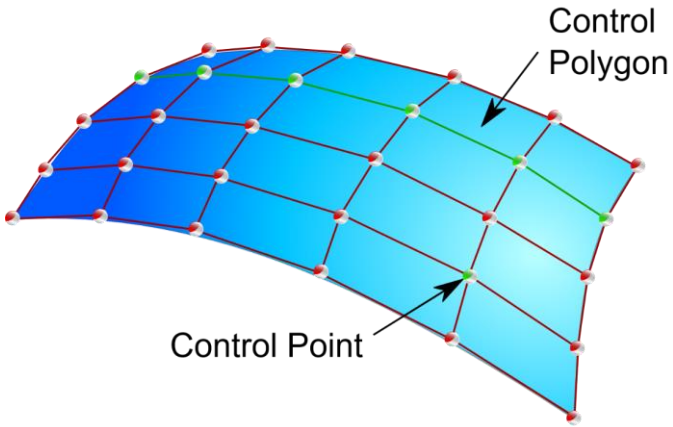




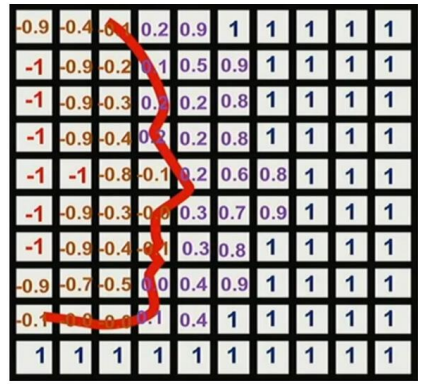
- Polygonal Meshes



- Parametric Surfaces



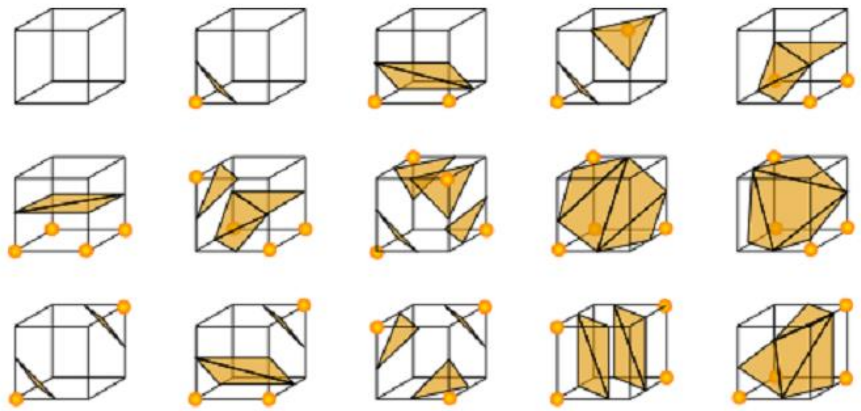
- Implicit Surfaces



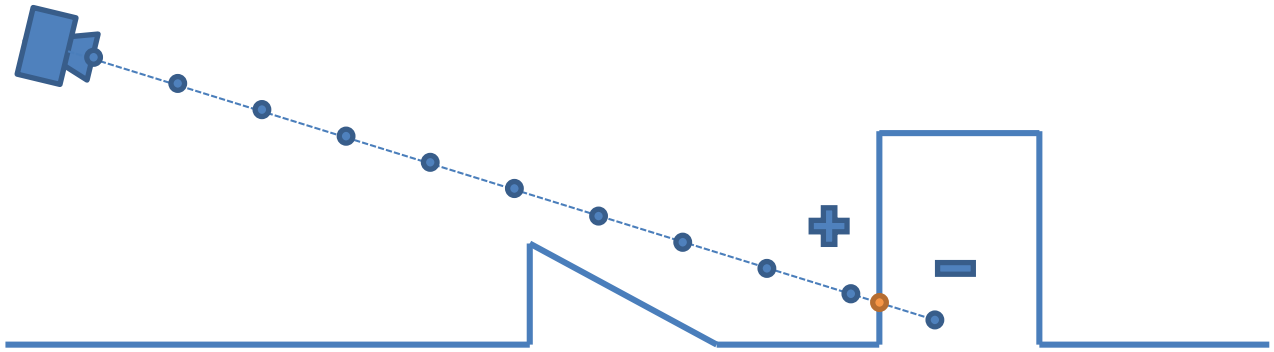
Truncated Signed Distance Field (TSDF)

Last Lecture: Surface Representations

- Important Algorithms
  - Marching Cubes
  - Ray cast



Marching Cubes table



Ray Casting

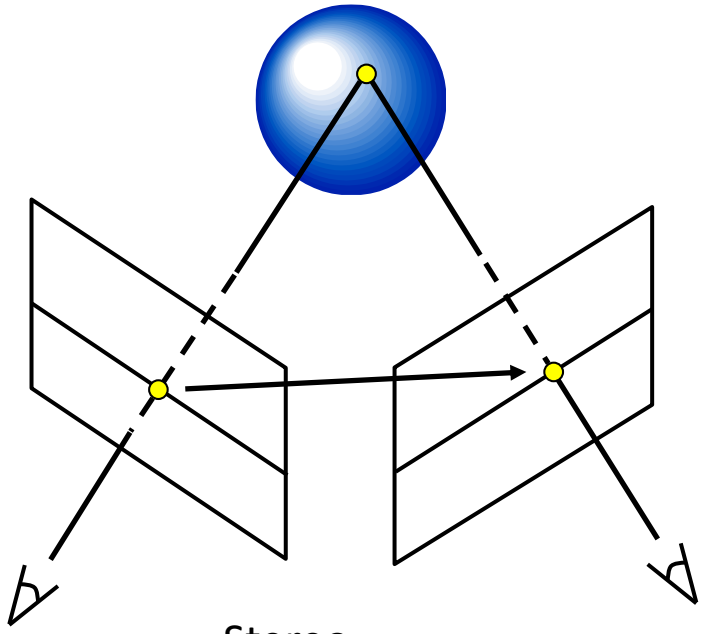
Today: Overview of 3D reconstruction methods

Overview: 3D Reconstruction Methods

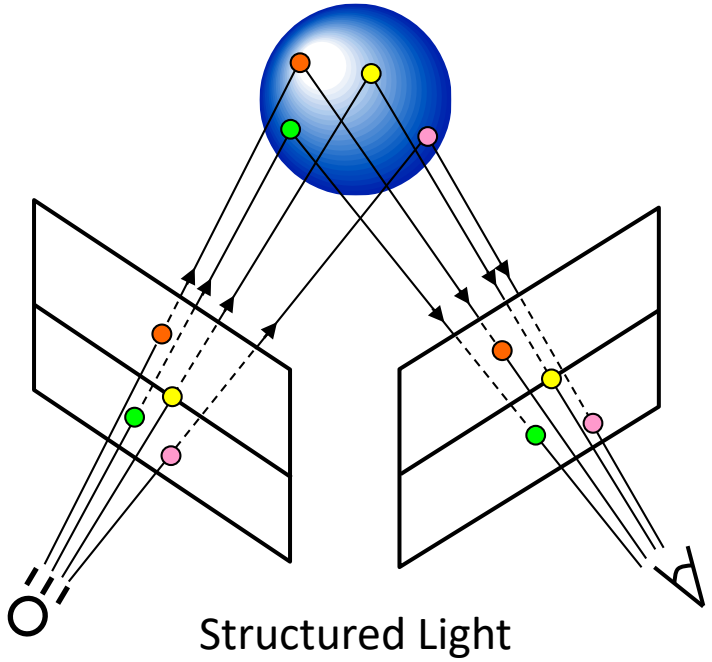
- Single Image (extremely ill-posed problem)
  - Prior-based (lots of learning methods)
  - Shape from Shading
- Multiple Images
  - Multiple camera setup
  - Video stream from one camera
- Depth Cameras
  - Active / passive
  - Real-time vs offline
  - Etc.

Overview: 3D Reconstruction Methods

- As we have seen for range sensing devices, we always need two optical centers (either two cameras, or one-proj-one-cam setups) in order to obtain actual depth data



Stereo

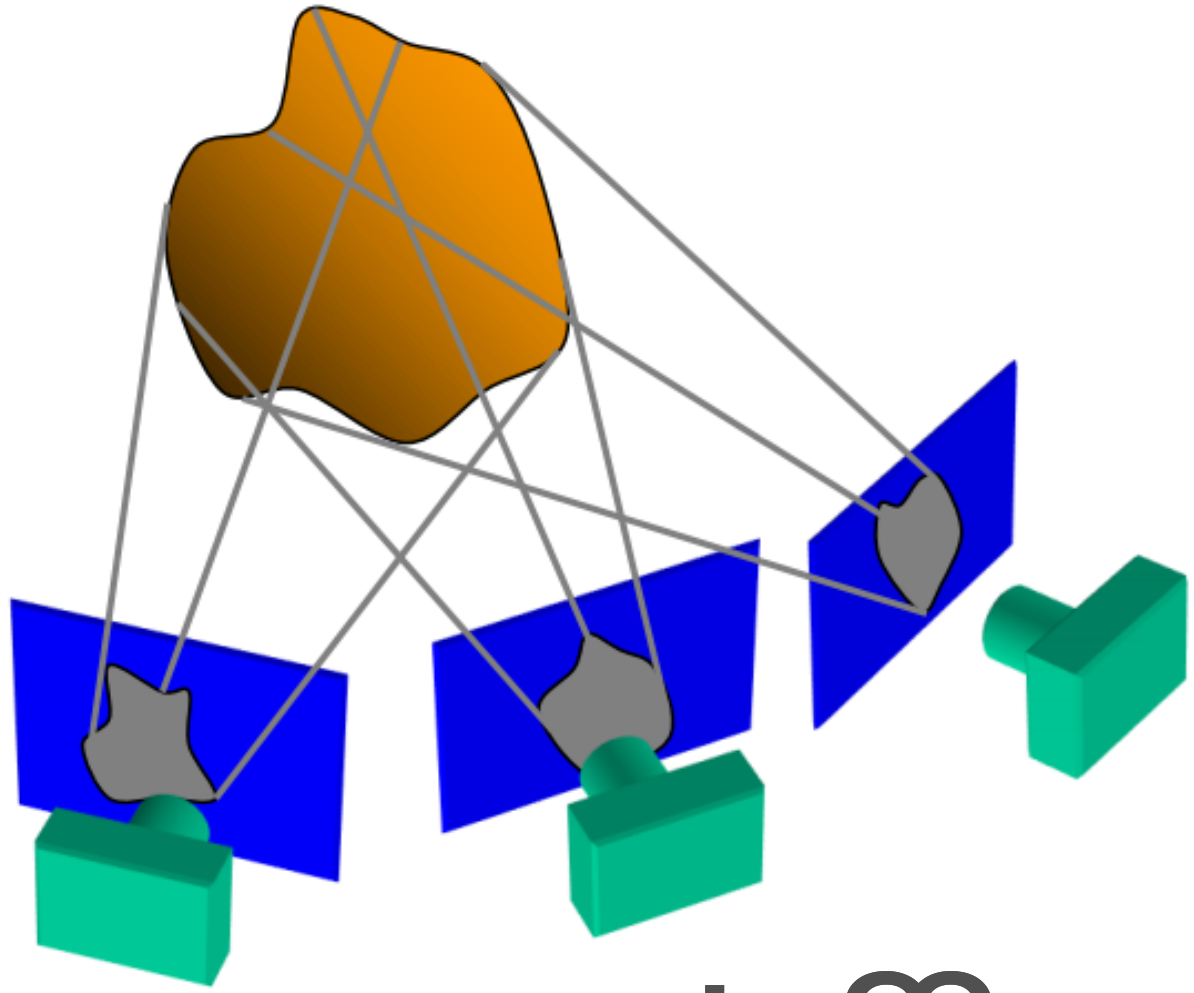


Structured Light

Visual Hull Carving

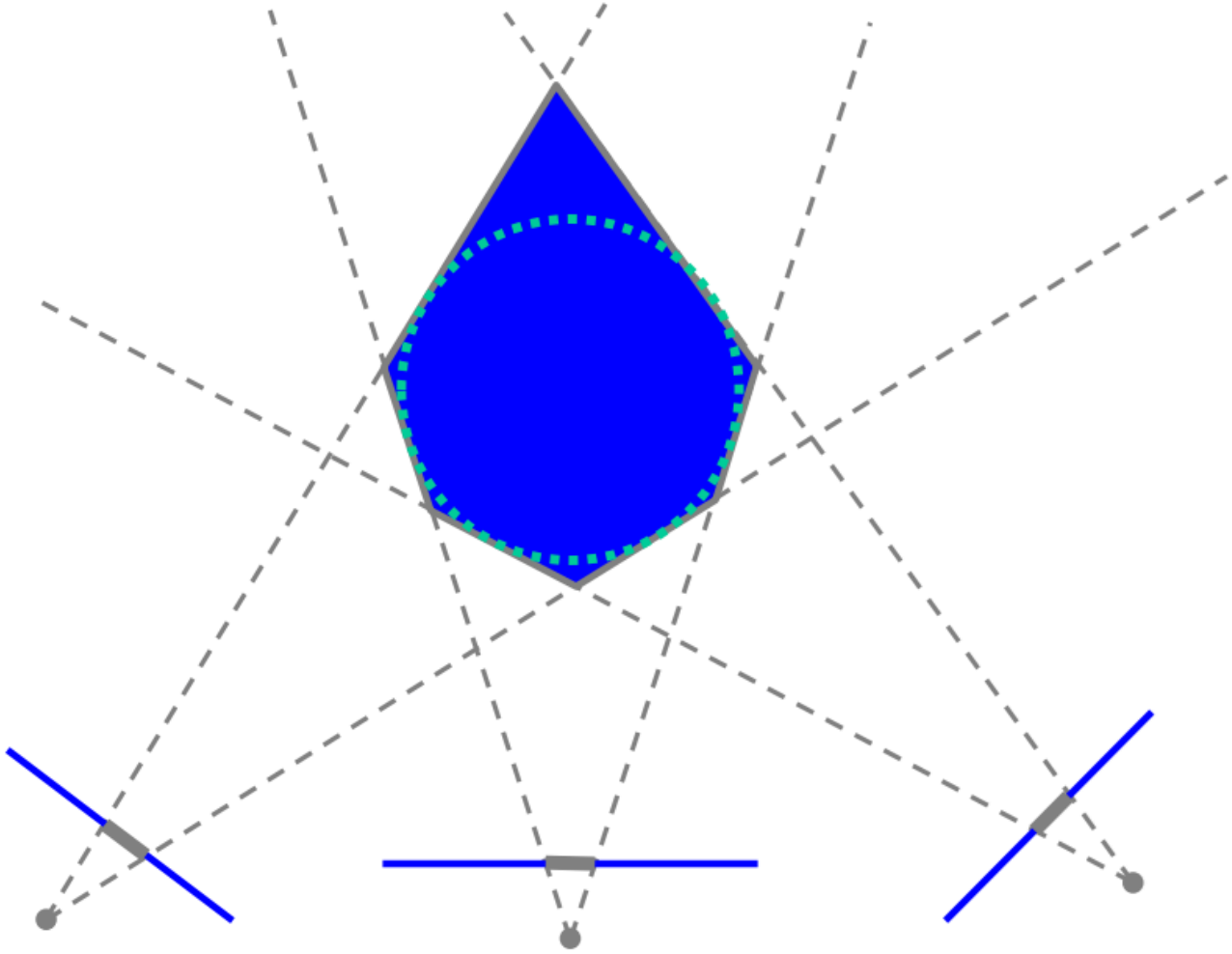
Visual Hull Carving

- Calibrated multi-view setup
  - Given intrinsics of each cam
  - Given extrinsics (between them)
- RGB stream / camera
- Need object segmentation



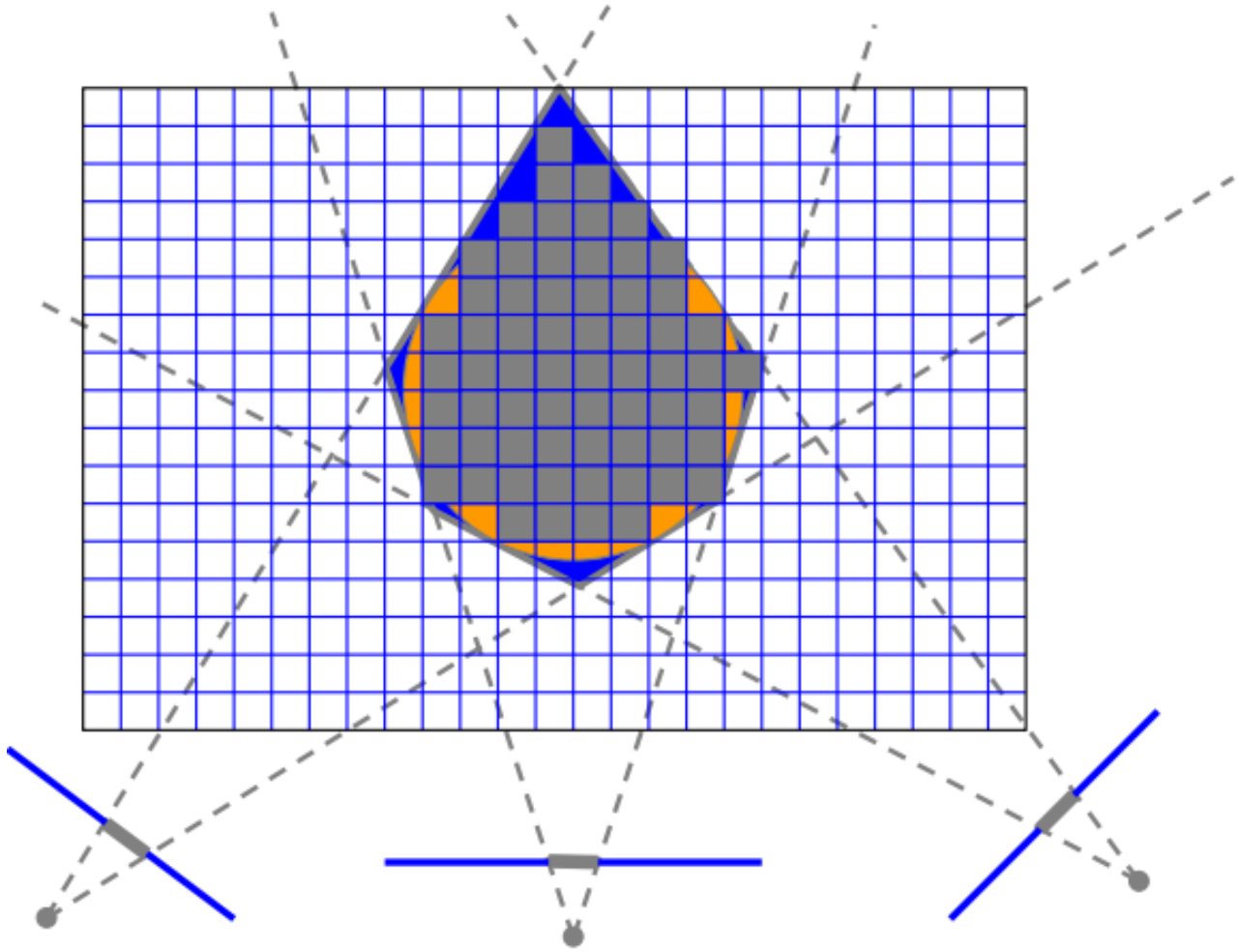
[Kutulakos and Seitz 00] A theory of shape by space carving

Visual Hull Carving



[Kutulakos and Seitz 00] A theory of shape by space carving

Visual Hull Carving



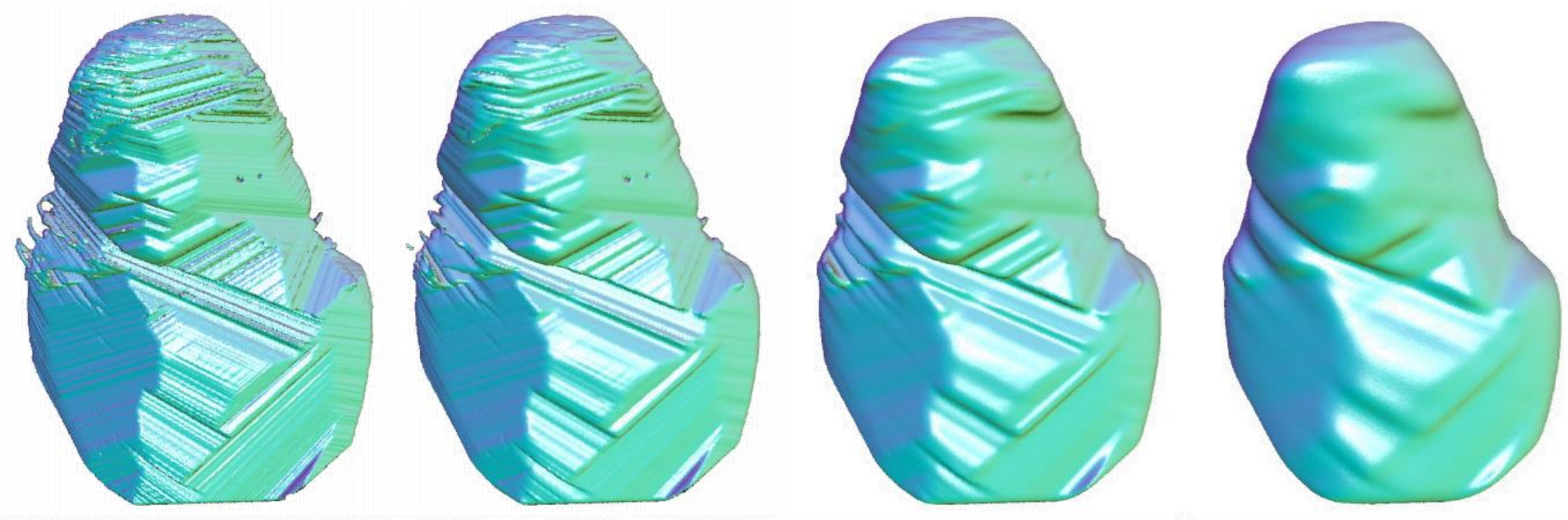
[Kutulakos and Seitz 00] A theory of shape by space carving

Visual Hull Carving

[Loop et al. 2013] Real-time high-resolution sparse…

Visual Hull Carving

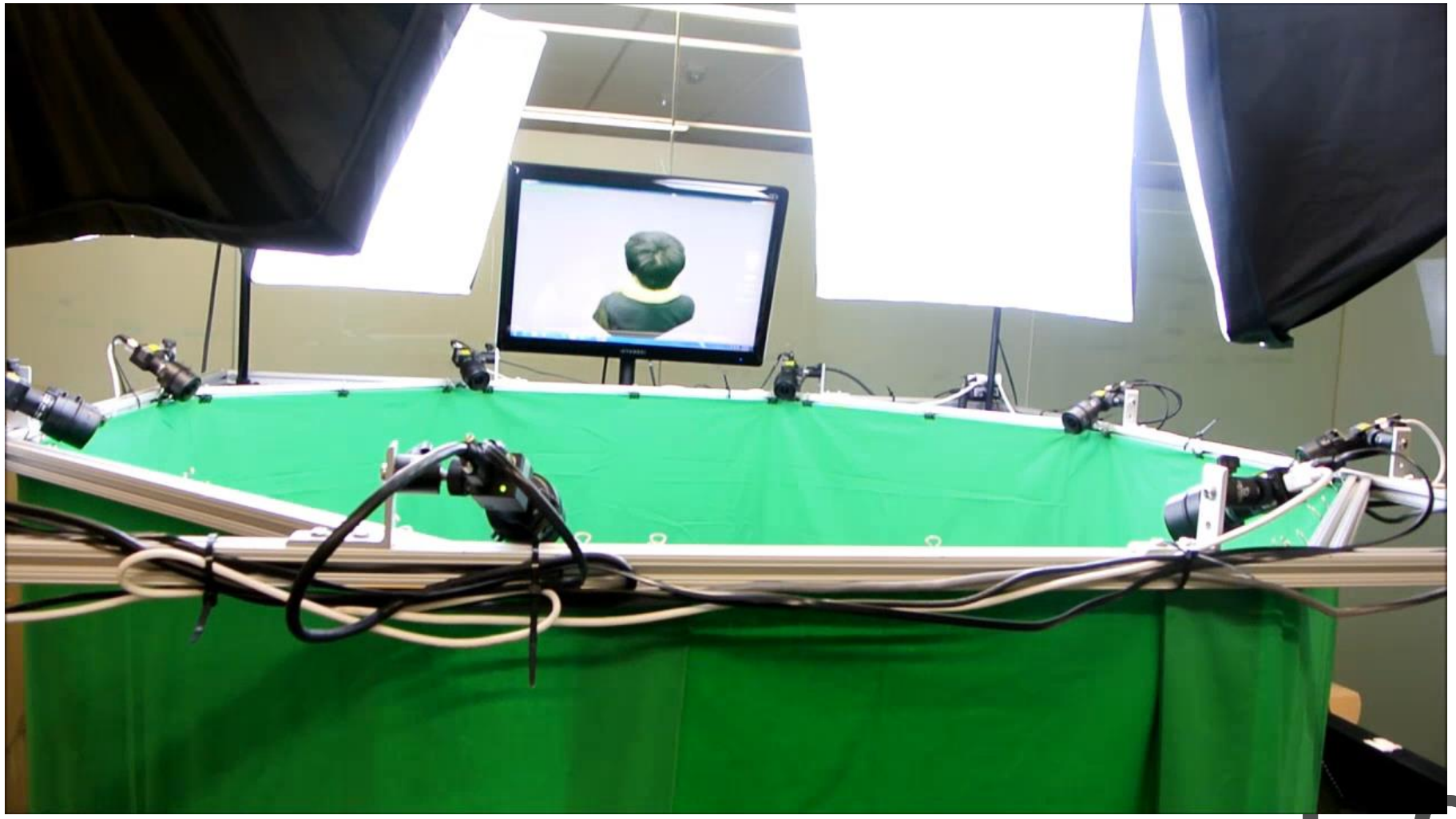
Can apply various smoothing kernels to improve quality



11 21 41 81

[Loop et al. 2013] Real-time high-resolution sparse…

Visual Hull Carving

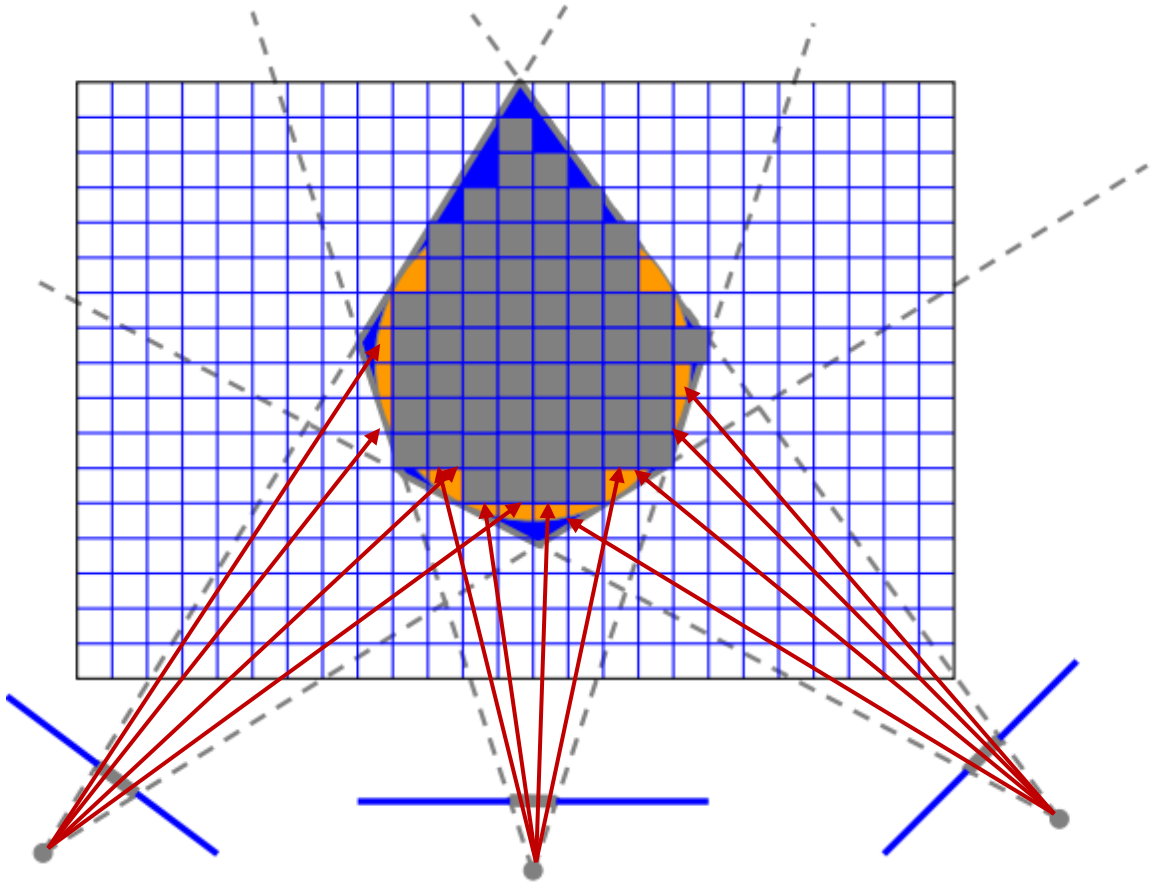


[Loop et al. 2013] Real-time high-resolution sparse…

Space Carving with Depth

- Could do with depth data as well

Need to ‘fuse’ data from views
Need to somehow handle noise
-> very challenging in practice



Visual Hull / Space Carving

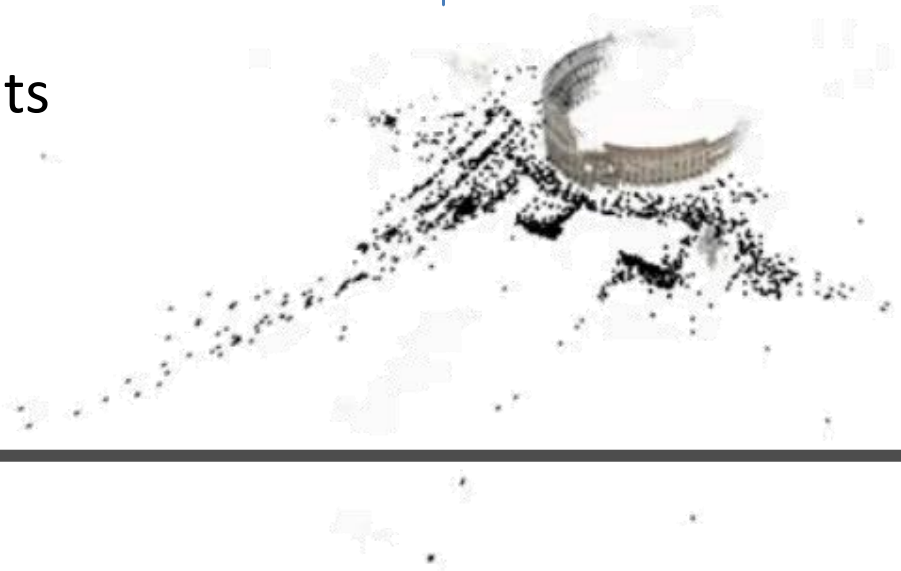
- Requires calibrated setups
- Background <-> foreground segmentation
- Not very robust to noise
- Good for conceptual understanding

Structure from Motion (SfM)

Structure from Motion (SfM)



SfM: Recover both 3D points and camera positions



Structure from Motion (SfM)

- Take set of images
- Detect key points in each image
- Find correspondences between images (i.e., match key points)

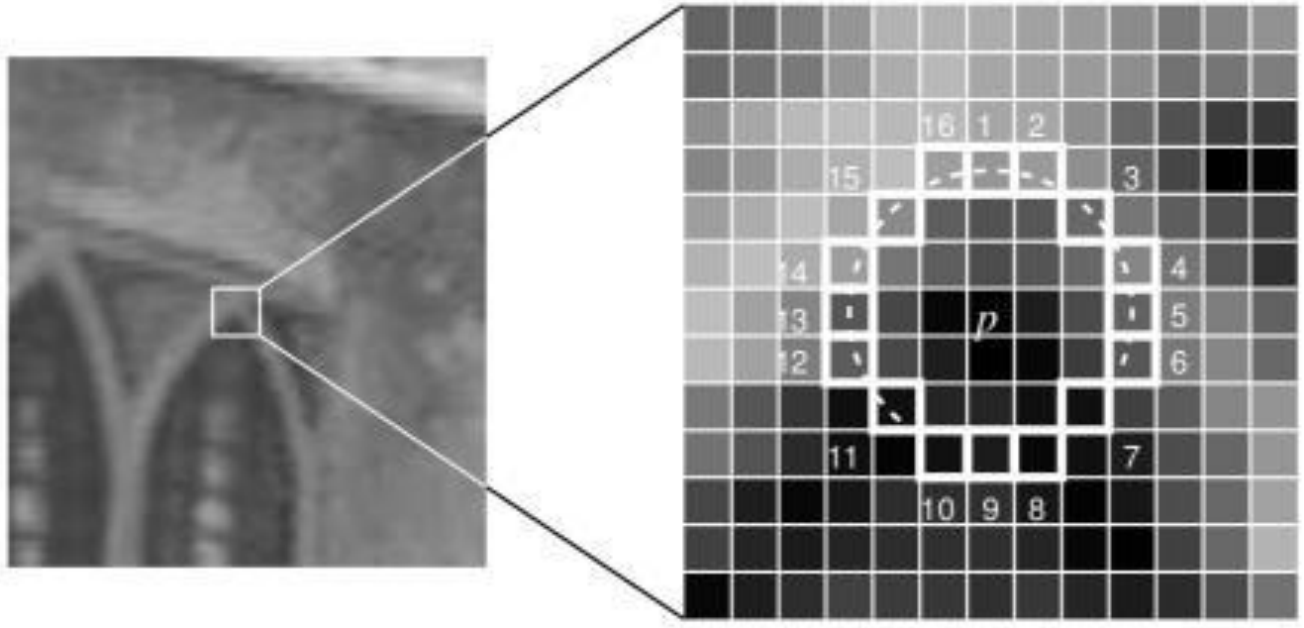
- Each image I_i has a transformation T_i
  - Given all matches, jointly solve for T_i and 3D location of points X_j

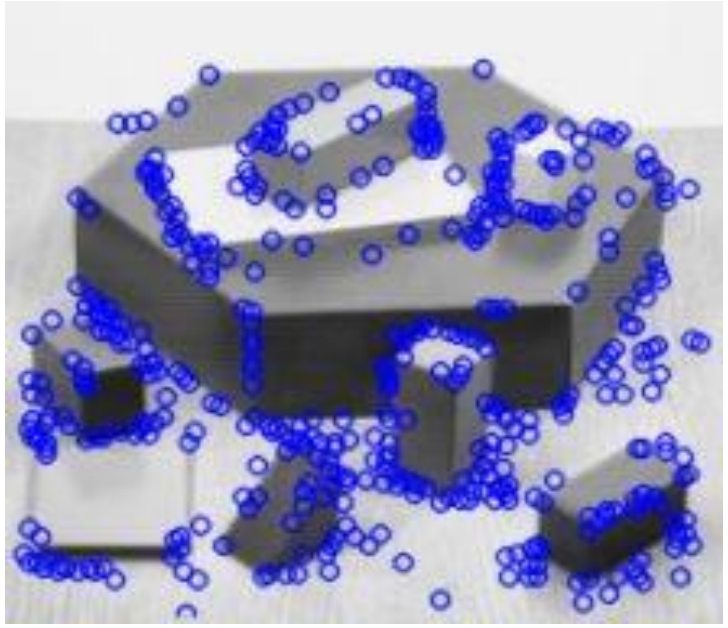
SfM: Keypoint Detection

- Harris corner detector
- FAST corner detector
- Shi-Tomasi corner detector
- SIFT Detector
- …

SfM: Keypoint Detection

- FAST keypoint detector





≥ 12 contiguous pixels brighter than p + thresh

SfM: Keypoint Detection

- Harris Corner Detector

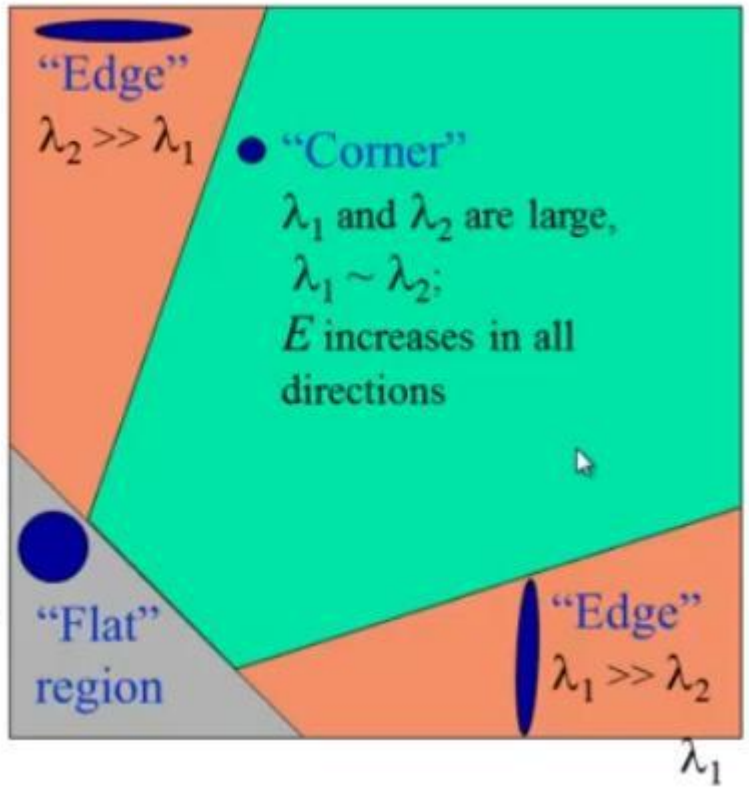
$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{window function}} \underbrace{[I(x+u,y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}]^2$$

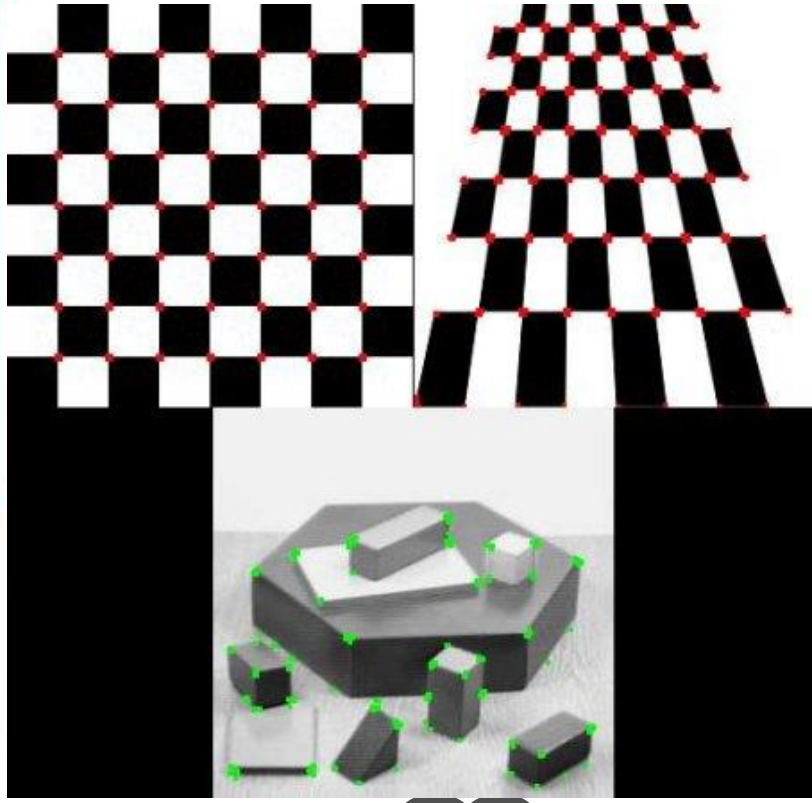
$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

$$R = det(M) - k(trace(M))^2$$

$|R|$ small -> region is flat (λ_1,λ_2 are small)
$R < 0$ -> region is edge ($\lambda_1 \gg \lambda_2$)
R is large -> corner (λ_1,λ_2 are both large)

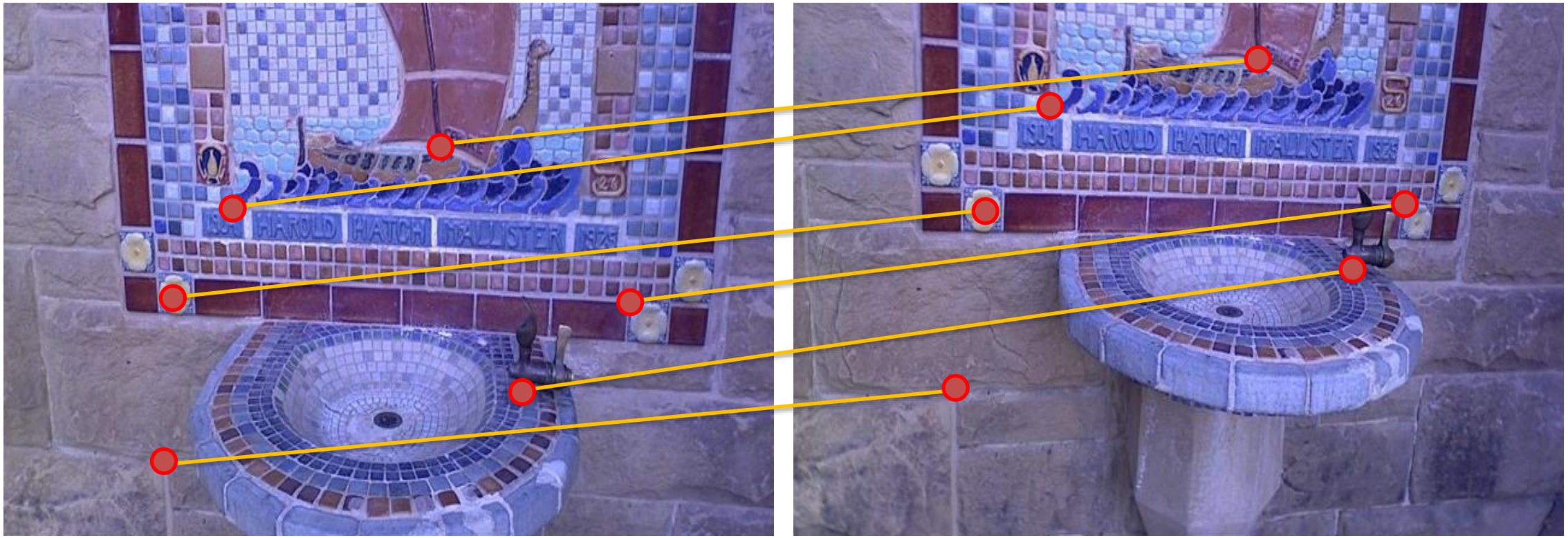




SfM: Correspondence Finding / Matching

SfM: Correspondence Finding / Matching

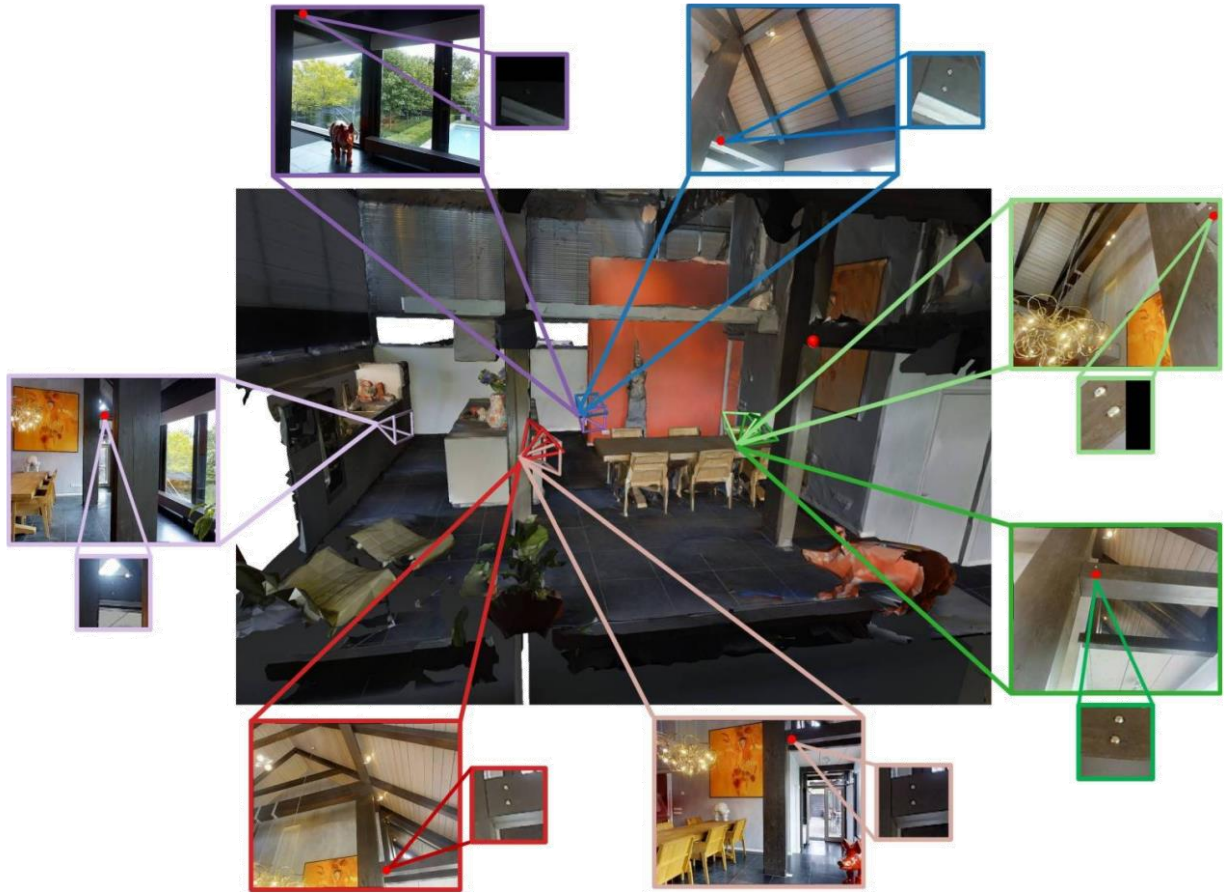
SfM: Correspondence Finding / Matching



SfM: Correspondence Finding / Matching

- Features establish matches

Same point seen
from many views

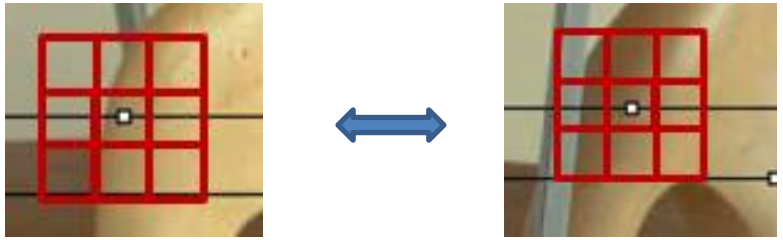


SfM: Correspondence Finding / Matching

- What are features?
  - Typically around key points
  - Maps high-dim space -> low(er)-dim
  - Easy comparison in feature space

Naïve Feature

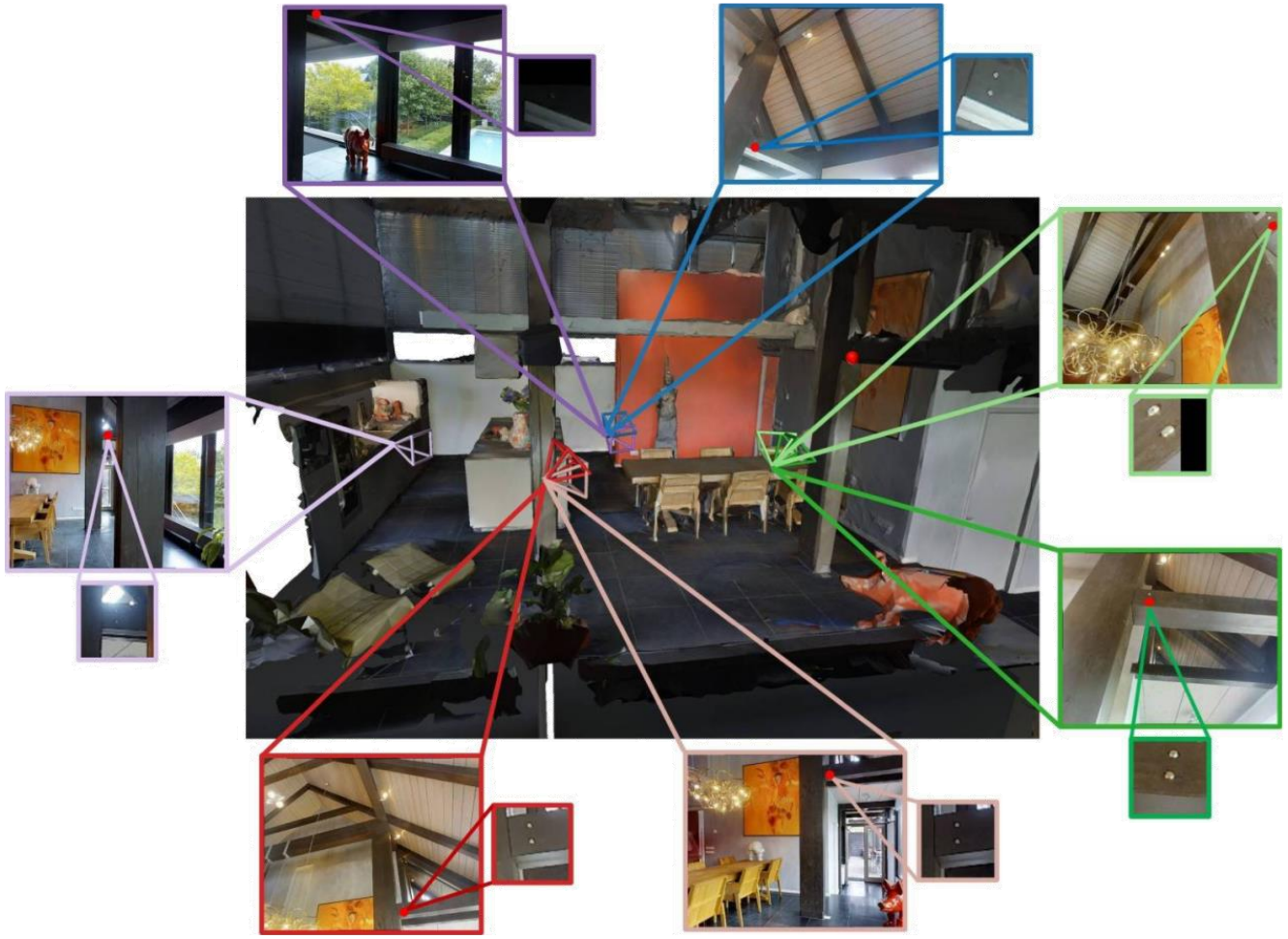
$$SSD(u,v) = \sum_{(u,v)} \left(I_{left}(i,j) - I_{right}(i,j) \right)^2$$



- E.g., take $f: 256^2 \rightarrow 128$
  - Distance between two features could be L_2: 128-dimensional dot product

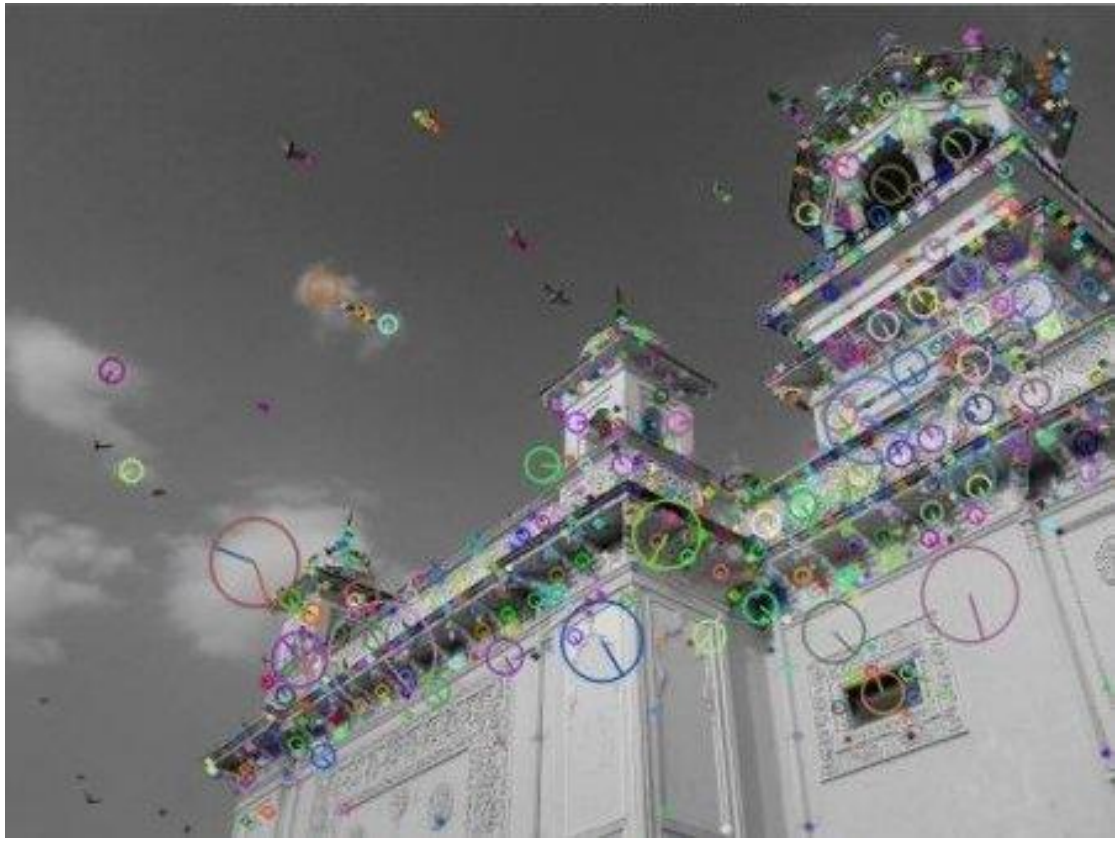
SfM: Correspondence Finding / Matching

- Properties of good features
  - Fast comparisons;
    e.g., L_2 in desc-space
    -> could build Kd-Tree
  - Scale invariant
  - View invariant
  - Lighting invariant



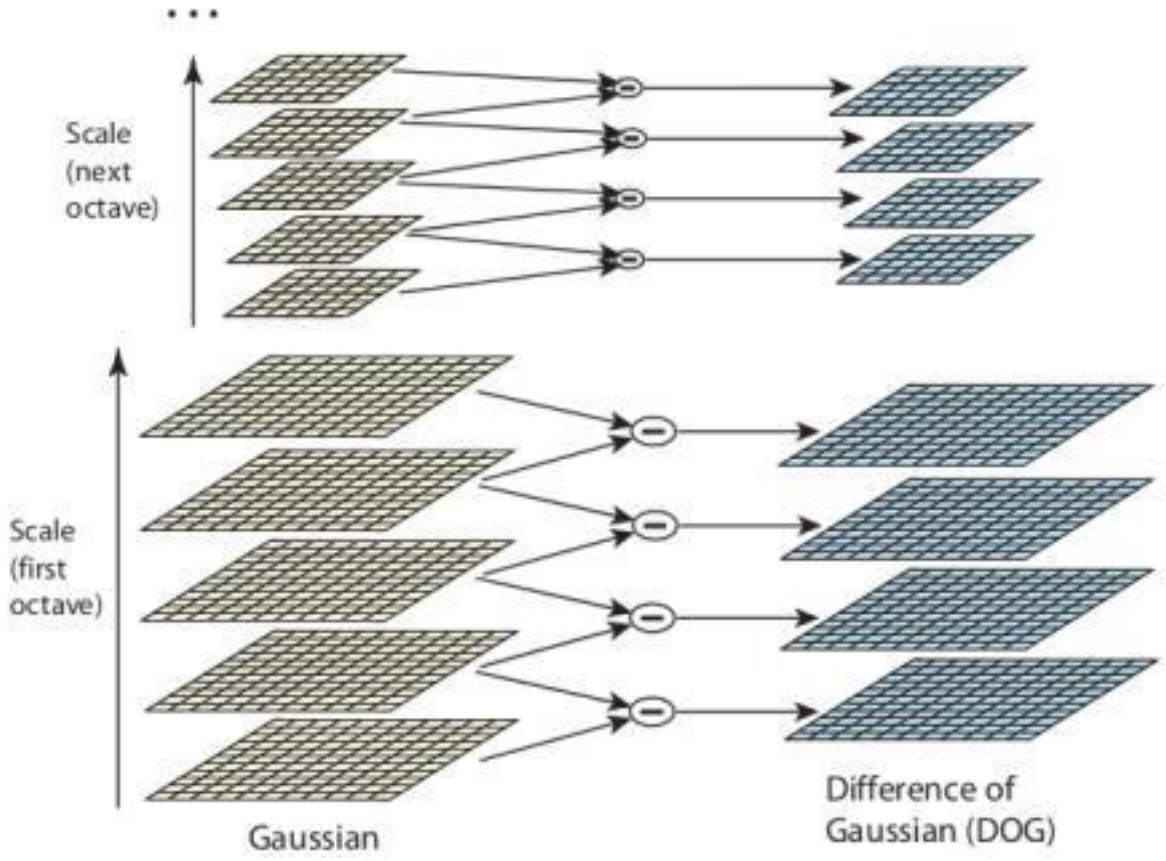
SfM: Correspondence Finding / Matching

- Feature descriptors for matching:
  - SIFT
  - SURF
  - ORB
  - FREAK
  - …
  - Use machine learning ☺
    - MatchNet, LIFT, or 3DMatch (in 3D).



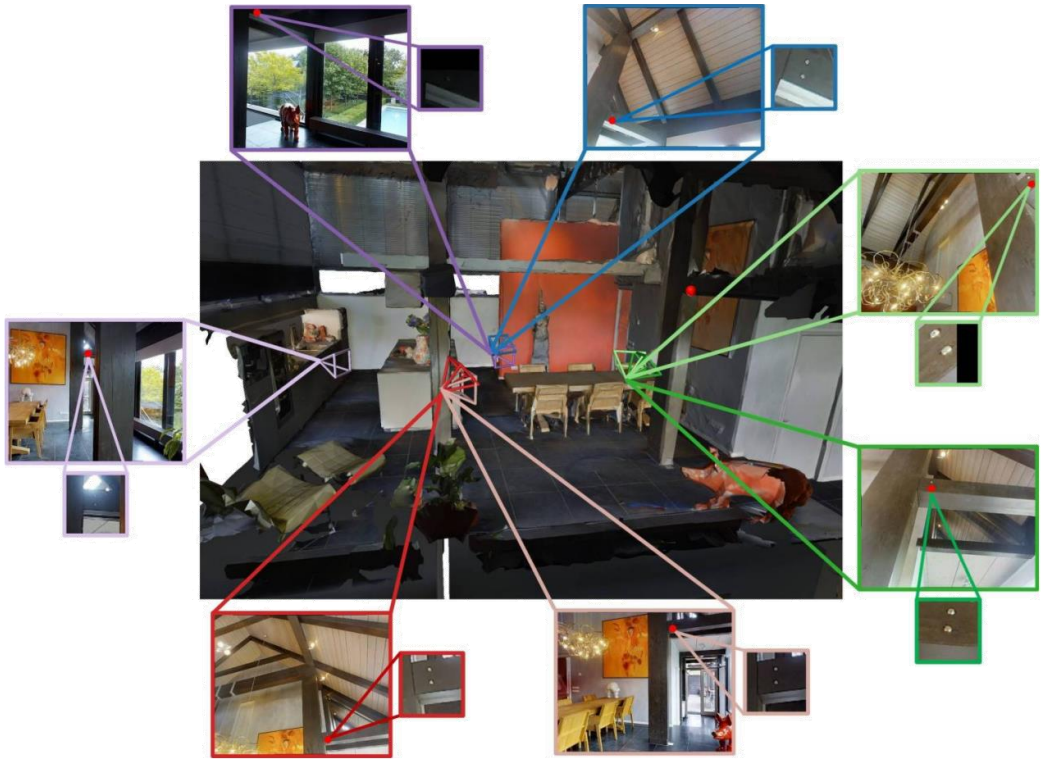
SfM: Correspondence Finding / Matching

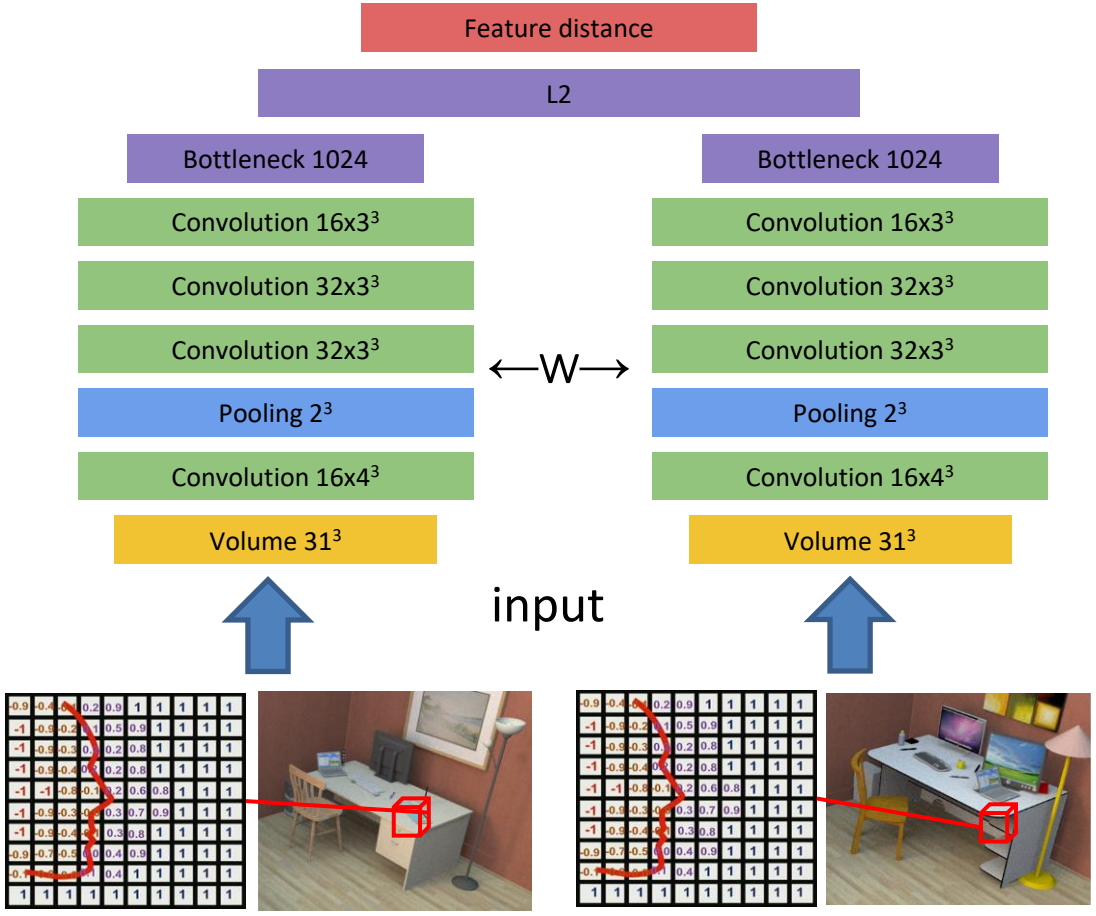
- SIFT (Scale-Invariant Feature Transform)
  - Difference of Gaussians
  - Neighborhood is divided in 16 sub-blocks
  - 8-bin orientation histograms
  - 128-dimensional
  - Matching: dot product between feature vectors
  - Quite costly to compute…



SfM: Correspondence Finding / Matching

- Learning Features with DeepLearning (ConvNets)





3DMatch [Zhang et al. 17]

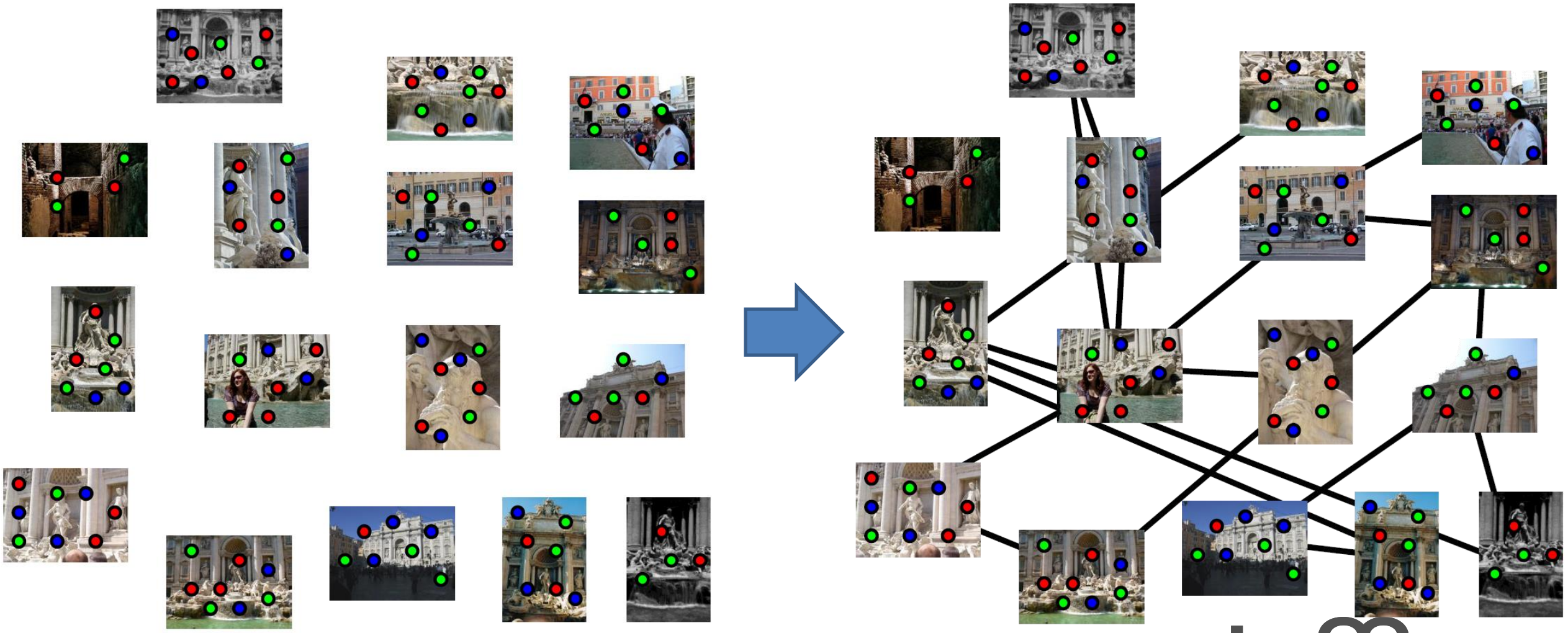
SfM: Correspondence Finding / Matching

- Learning Features with DeepLearning (ConvNets)

SURF	46.8%
SIFT	37.8%
ResNet-50 w/ Matterport3D	10.6%
ResNet-50 w/ SUN3D	10.5%
ResNet-50 w/ Matterport3D + SUN3D	9.2%

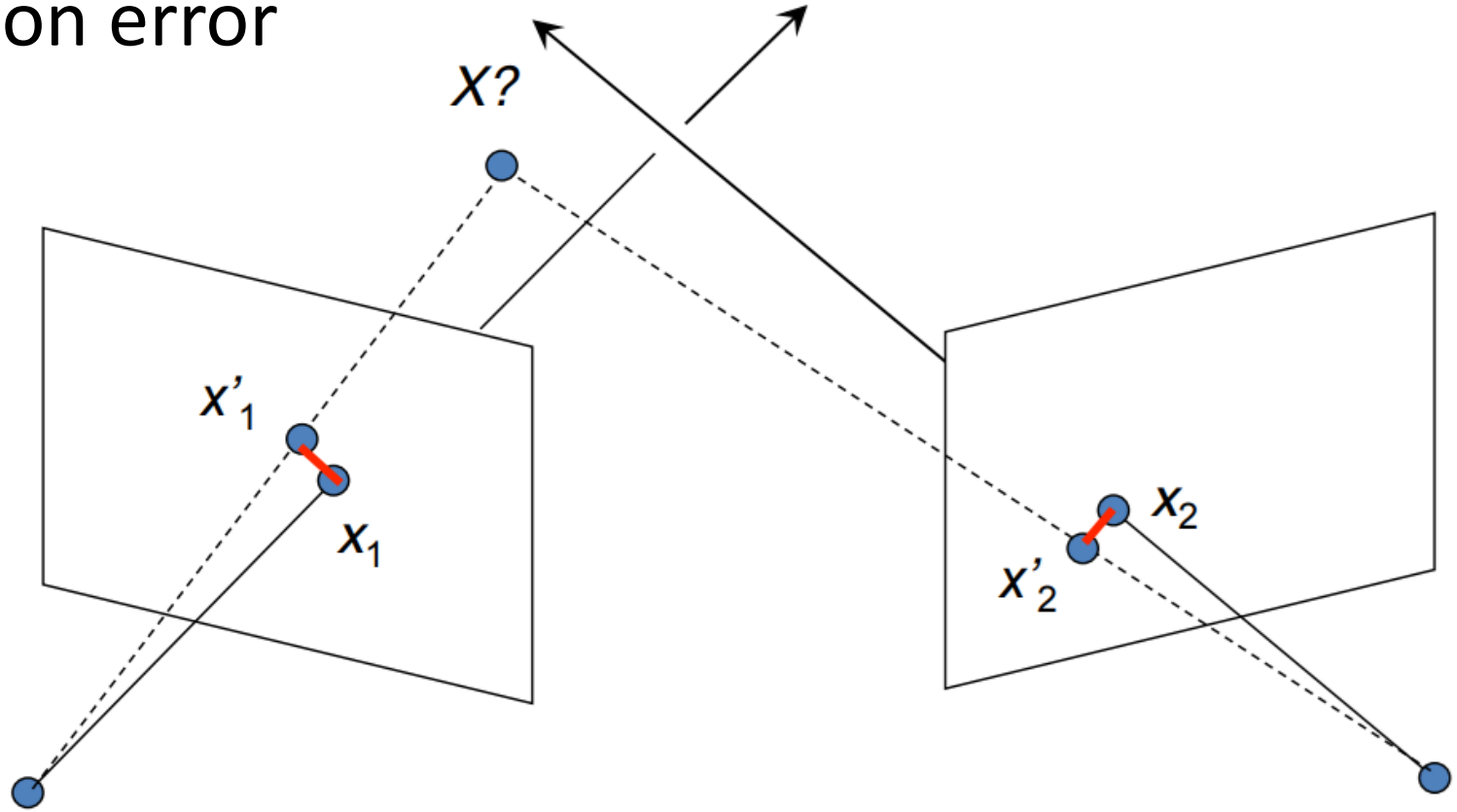
Table 1: **Keypoint matching results.** Error (%) at 95% recall on ground truth correspondences from the SUN3D testing scenes. We see an improvement in performance from pretraining on Matterport3D.

SfM: Correspondence Finding / Matching



SfM: Bundle Adjustment

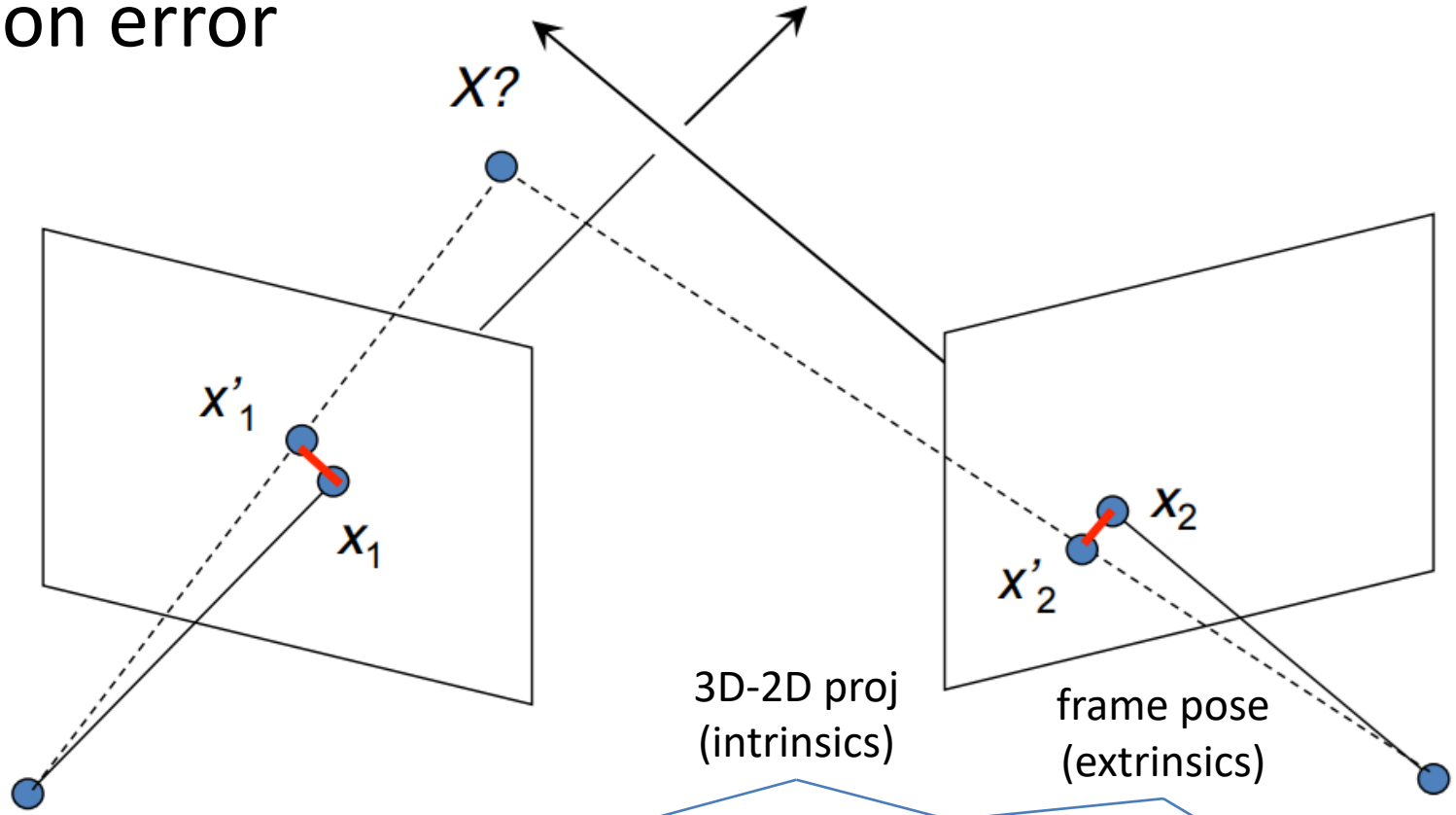
- Re-projection error



$$E(T_{left}, T_{right}, X) = \left|\left|x_1 - \pi_{left}(T_{left} \cdot X)\right|\right|_2^2 + \left|\left|x_2 - \pi_{right}(T_{right} \cdot X)\right|\right|_2^2$$

SfM: Bundle Adjustment

- Re-projection error



3D-2D proj (intrinsics)

frame pose (extrinsics)

$$E(T_{left}, T_{right}, X) = \left|\left|x_1 - \pi_{left}(T_{left} \cdot X)\right|\right|_2^2 + \left|\left|x_2 - \pi_{right}(T_{right} \cdot X)\right|\right|_2^2$$

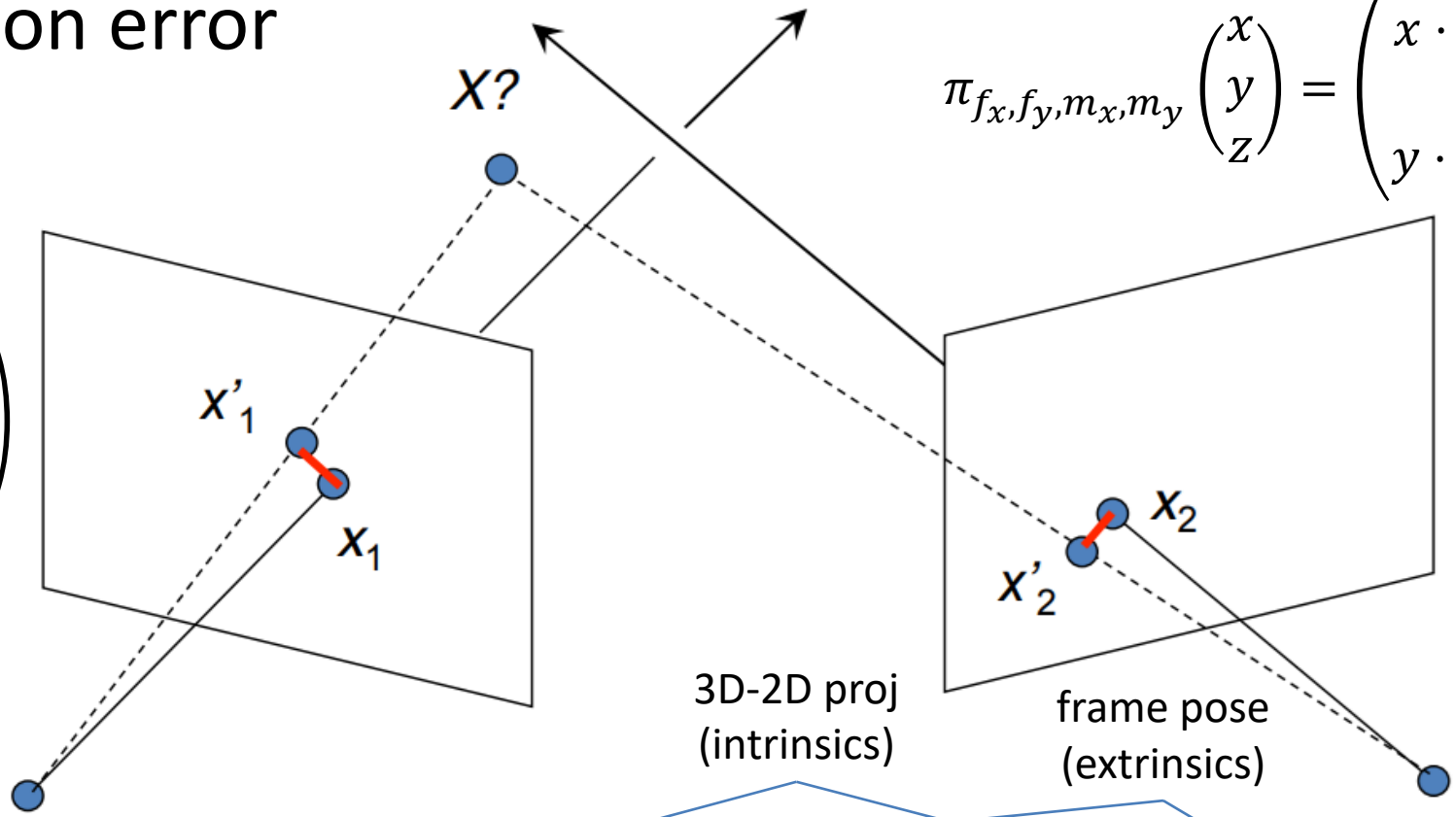
SfM: Bundle Adjustment

- Re-projection error

$$\pi_{f_x,f_y,m_x,m_y}\begin{pmatrix}x\\y\\z\end{pmatrix}=\begin{pmatrix}x\cdot\frac{f_x}{z}+m_x\\ y\cdot\frac{f_y}{z}+m_y\end{pmatrix}\rightarrow\begin{pmatrix}u\\v\end{pmatrix}$$

$$T(\alpha,\beta,\gamma,t_x,t_y,t_z)=$$

$$\begin{pmatrix}R_{00} & R_{01} & R_{02} & t_x\\ R_{10} & R_{11} & R_{12} & t_y\\ R_{20} & R_{21} & R_{22} & t_z\\ 0 & 0 & 0 & 1\end{pmatrix}$$



3D-2D proj (intrinsics)

frame pose (extrinsics)

$$E(T_{left},T_{right},X)=\left|\left|x_1-\pi_{left}(T_{left}\cdot X)\right|\right|_2^2+\left|\left|x_2-\pi_{right}(T_{right}\cdot X)\right|\right|_2^2$$

SfM: Bundle Adjustment

- m images
- n points in 3d

- $E_{re-proj}(\boldsymbol{T}, \boldsymbol{X}) =$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left|\left| x_{ij} - \pi_i(T_i \cdot X_j) \right|\right|_2^2$$

over images

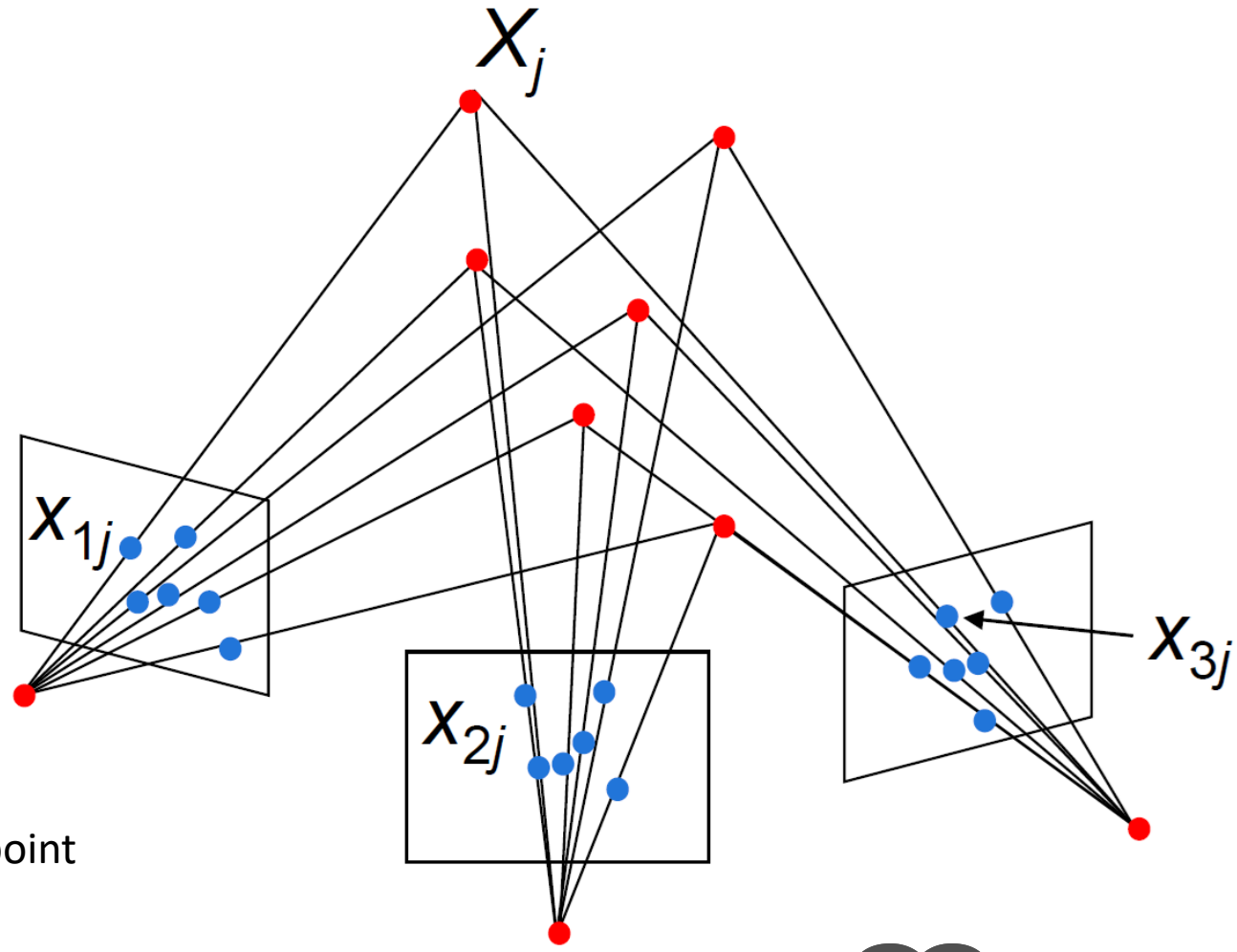
over 3d points

2d keypoint locations

3D-2D proj (intrinsics)

frame pose (extrinsics)

3D point



SfM: Bundle Adjustment

- m images n points in 3d
- Number of unknowns
  - $6 \cdot (m-1)$ for poses; $(T_1 = ID)$
  - $3 \cdot n$ for points
  - Possibly intrinsics (per camera or global)
- Number of constraints:
  - $2 \cdot m \cdot n$ (constraints are in 2D)

$$E_{re-proj}(\boldsymbol{T}, \boldsymbol{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \left|\left| x_{ij} - \pi_i(T_i \cdot X_j) \right|\right|_2^2$$

Between two (intnr. calibrated) images:
Need at least 6 correspondences
(also must not be linearly dependent)

- Under-constrained vs over-constrained?
  - Make sure $2 \cdot m \cdot n \geq 6 \cdot (m-1) + 3 \cdot n + (\text{'intrinsics'})$

Structure from Motion (SfM)

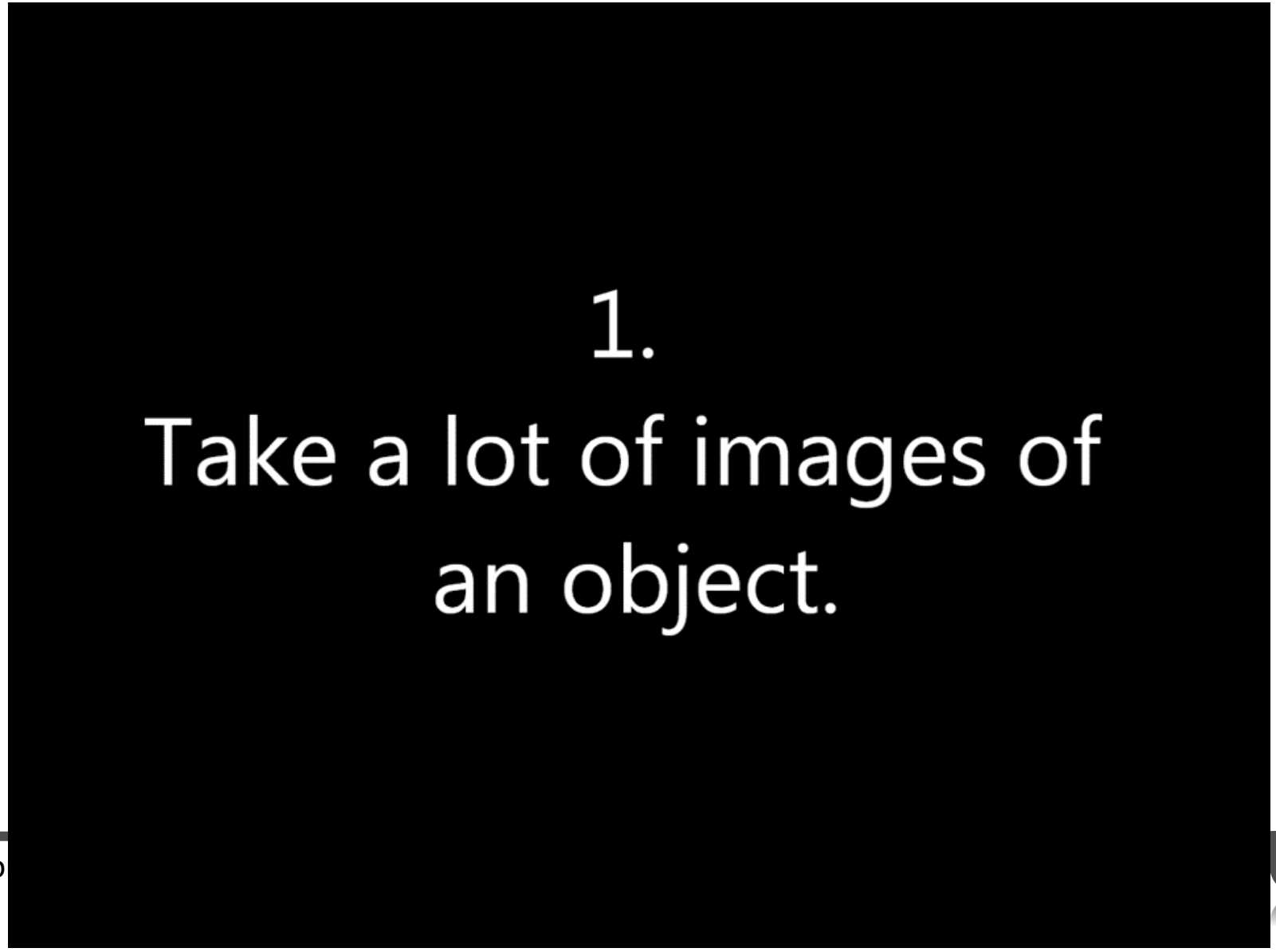
- Bottom line… we only obtain a sparse point cloud
- It's also slow ☺

- Key challenges:
  - Find good feature matches
  - Find good initialization for optimization

SfM: Bundler

[Snavely et al. 10] http://www.cs.cornell.edu/~snavely/bundler/

SfM Applications [Enquvist et al. 11]



SfM Applications: Building Rome in a Day

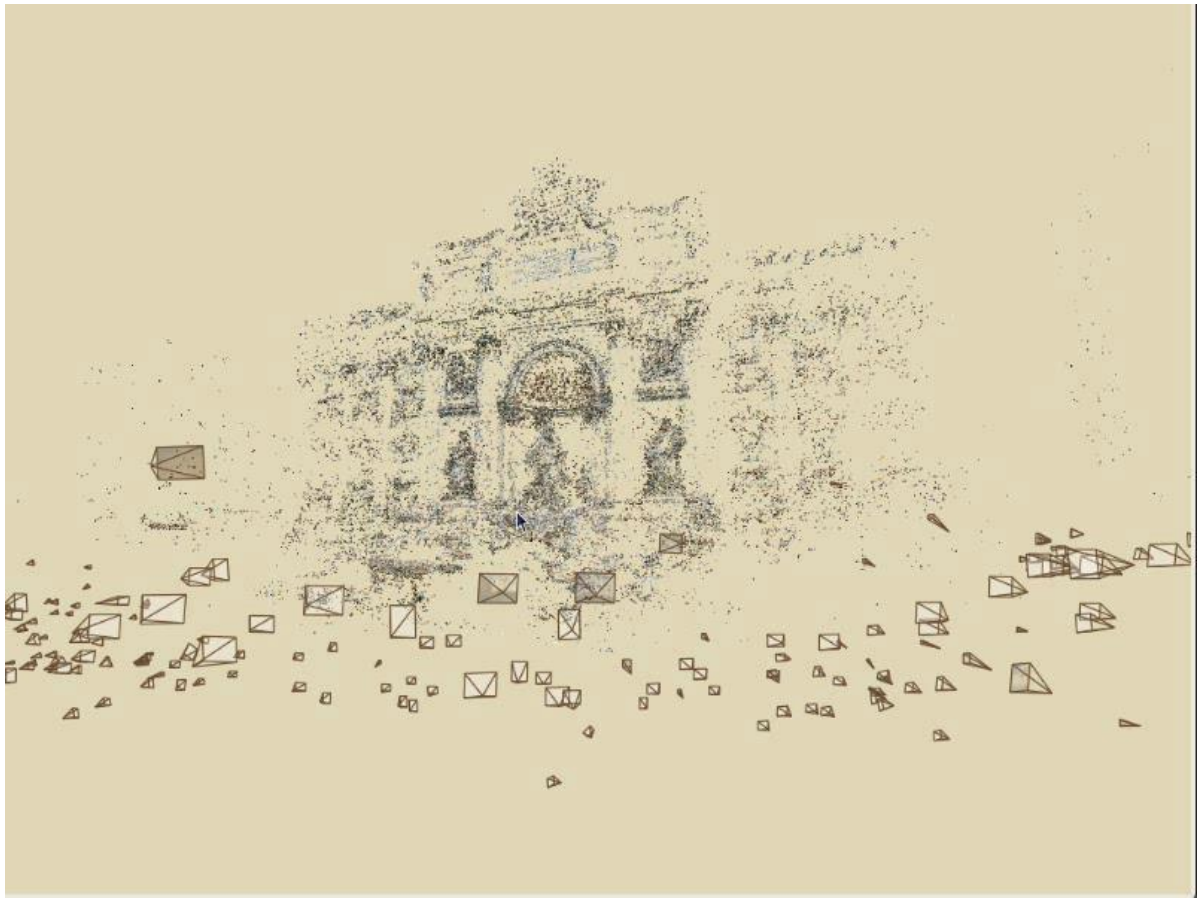
[Snavely et al. 11] Building Rome in a Day

SfM Applications: Building Rome in a Day

Dubrovnik, Croatia. (aka King's Landing)
4,619 images (out of an initial 57,845).
Total reconstruction time: 23 hours
Number of cores: 352

[Snavely et al. 11] Building Rome in a Day

SfM Applications: Photo Tourism



[Snavely et al. 06] Photo Tourism

SfM Applications: Hyperlapse

[Kopf et al. 14] First-person Hyperlapse

Multi-view Stereo (MVS)

- Take results of SfM
  - 6DoF Poses of images
  - Estimated 3D points (i.e., sparse point cloud)

- Aim to estimate dense surface (i.e., not only at keypoints)
- Many variations with semi-dense, etc.
- Often name ambiguity: Dense Photometric (multi-view) Stereo

Multi-view Stereo (MVS)

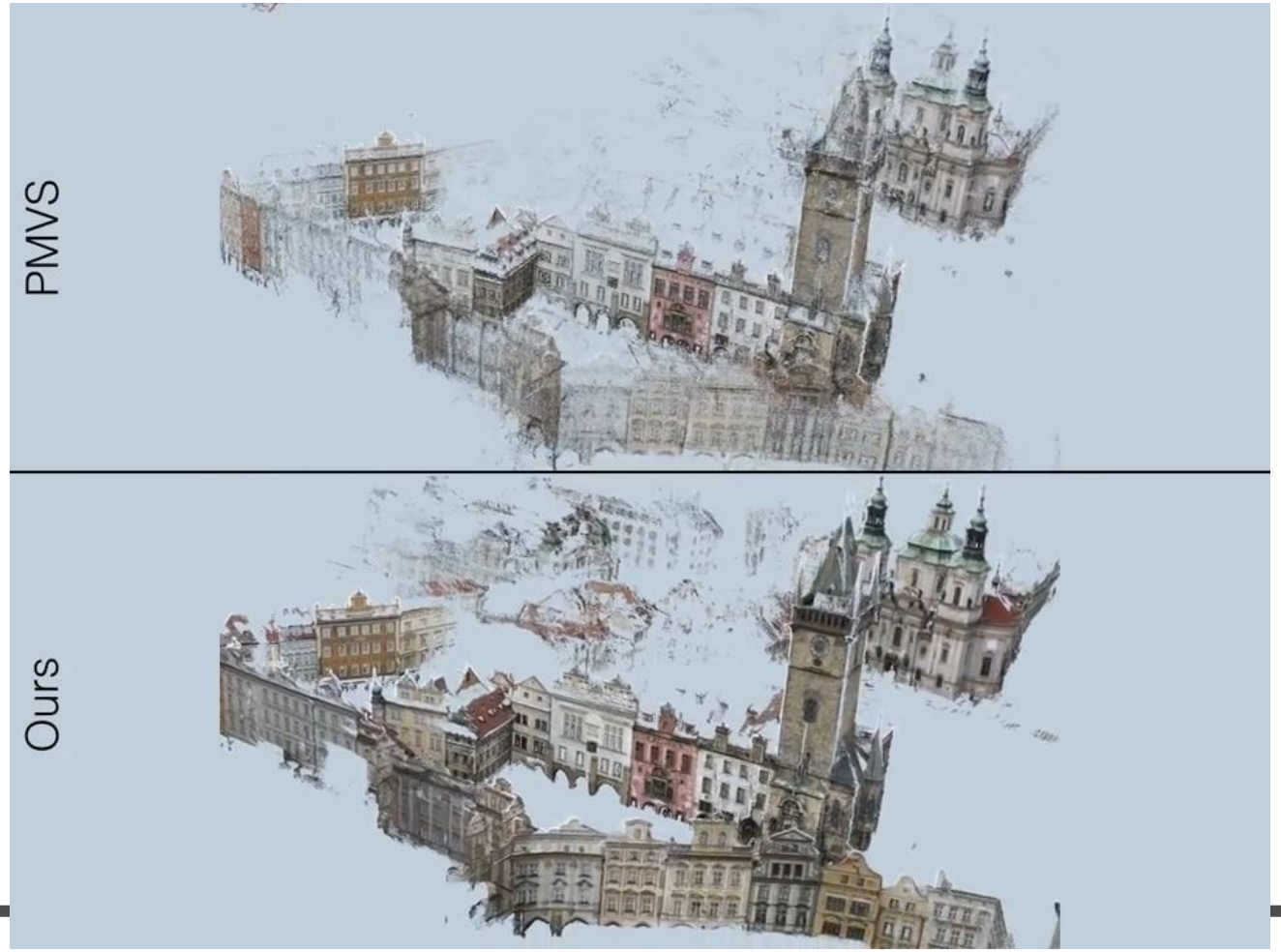
- Simple idea: just do dense stereo matching between every frame pair (we have the alignment + intrinsic)
  - Somehow ‘vote’ for results

- Global optimization:
  - Solve for all disparities simultaneously
  - Can be mapped to a Markov Random Field problem (MRF)
    - Matching cost term and pairwise terms

- Semi-global Matching (SGM) [Hirschmuller 08]
  - Approximate for tractable runtime

Multi-view Stereo (MVS)

- Goal is to create an as-dense-as-possible point cloud



Pixelwise view Selection for Unstructured Multi-view Stereo[Schoenberger et al. 16]

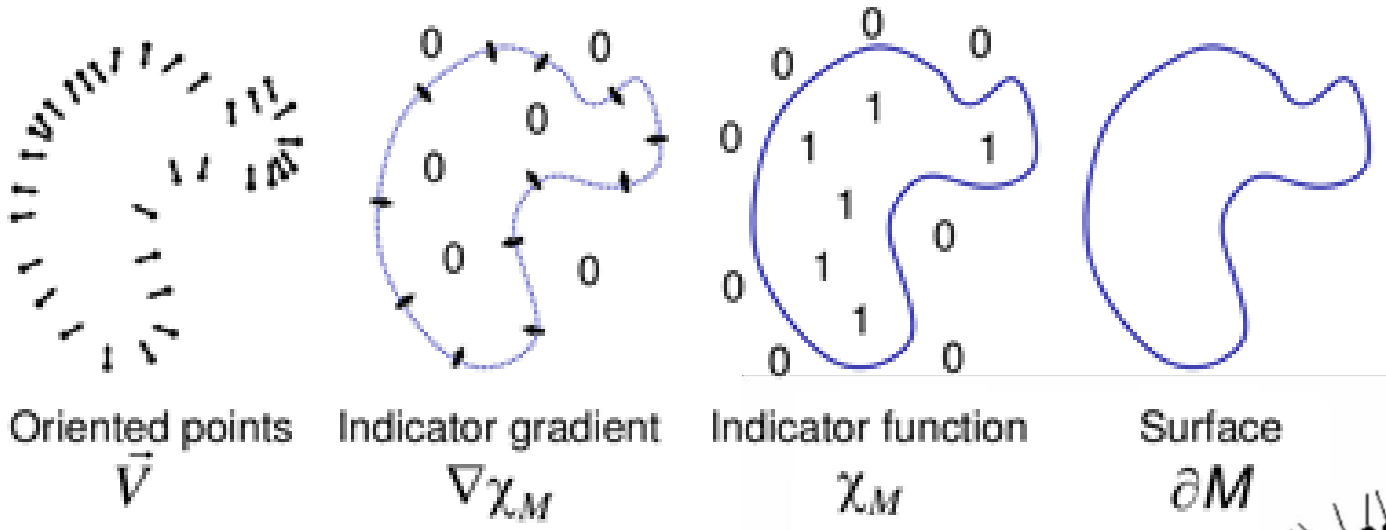
Final Reconstruction

- Use (dense) point cloud as input
  - Compute normals (e.g., using PCA)

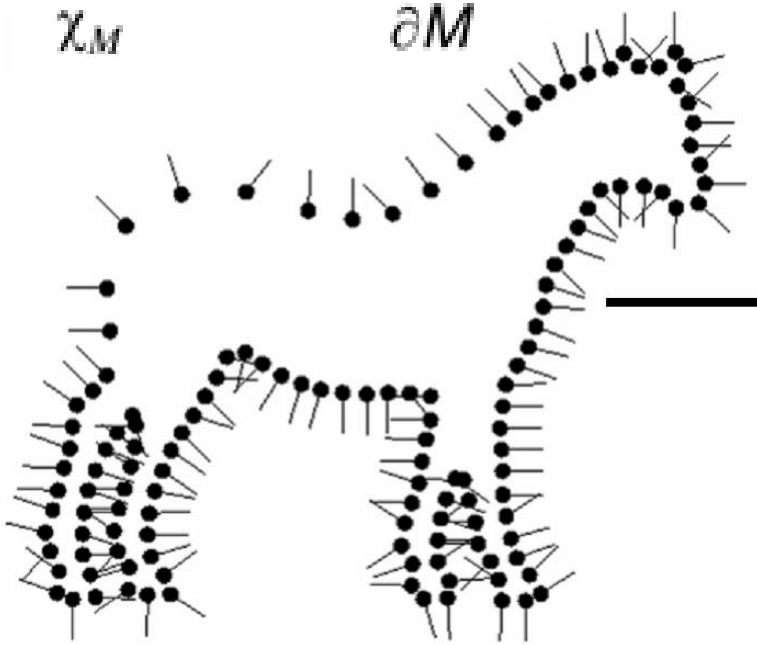
- Feed into some surface fitting / surface optimization
  - E.g., Poisson Surface Reconstruction [Kazhdan et al. 06/13]

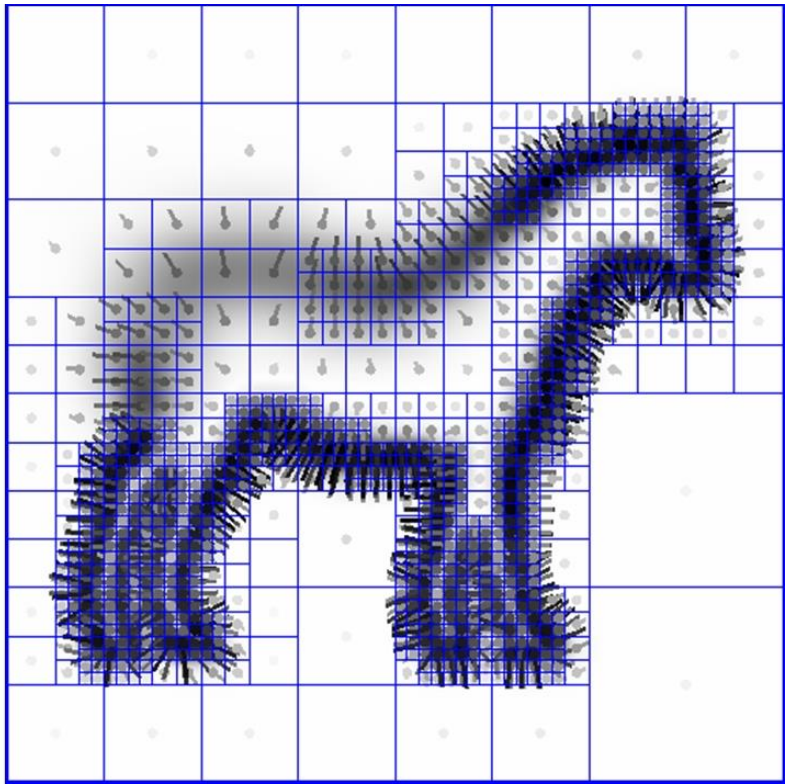
- Texture the resulting surface (e.g., optimize for texture warp on top of surface)

Poisson Surface Reconstruction [Kazhdan 06/13]

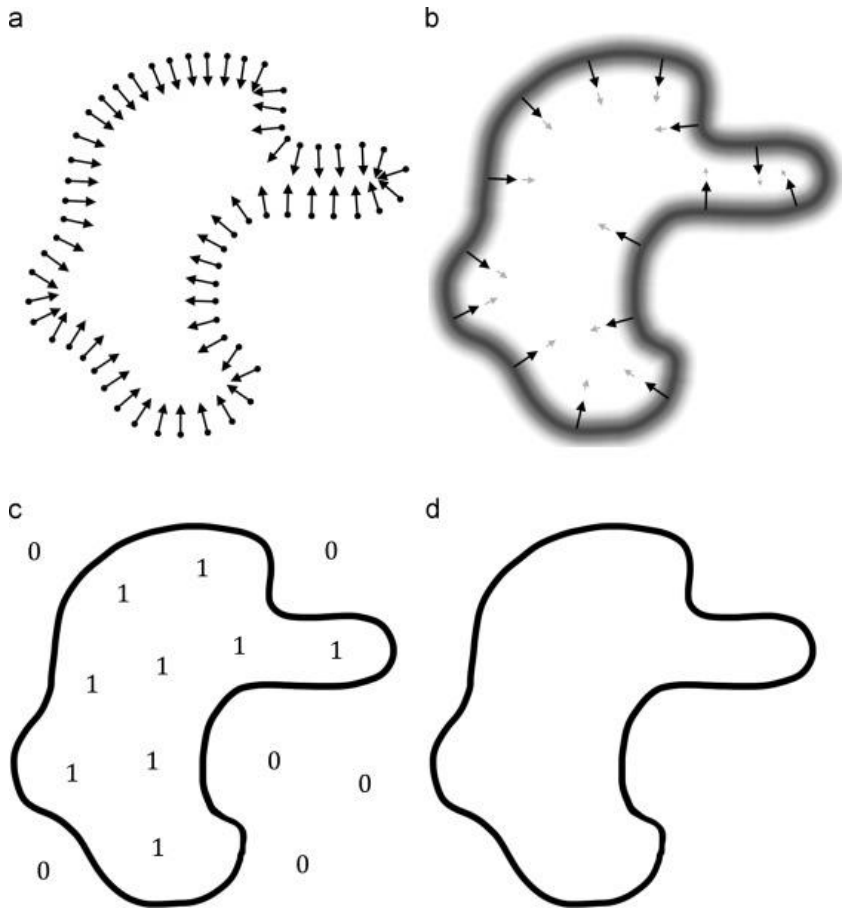


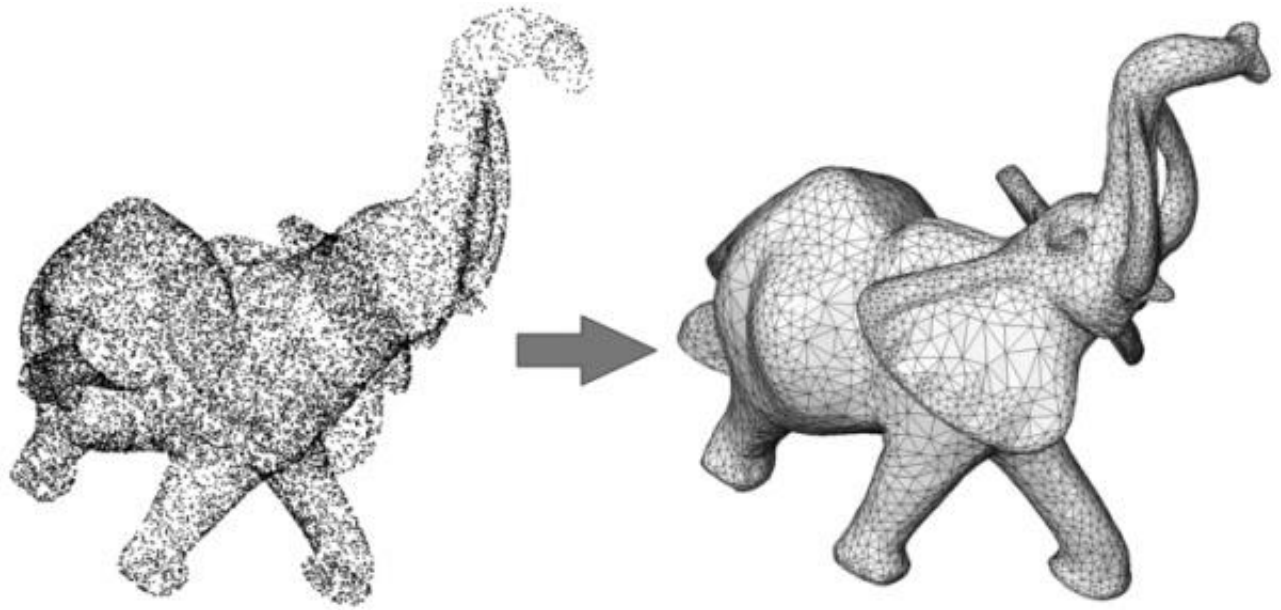
Input: point cloud with normals
Output: implicit function
-> mesh





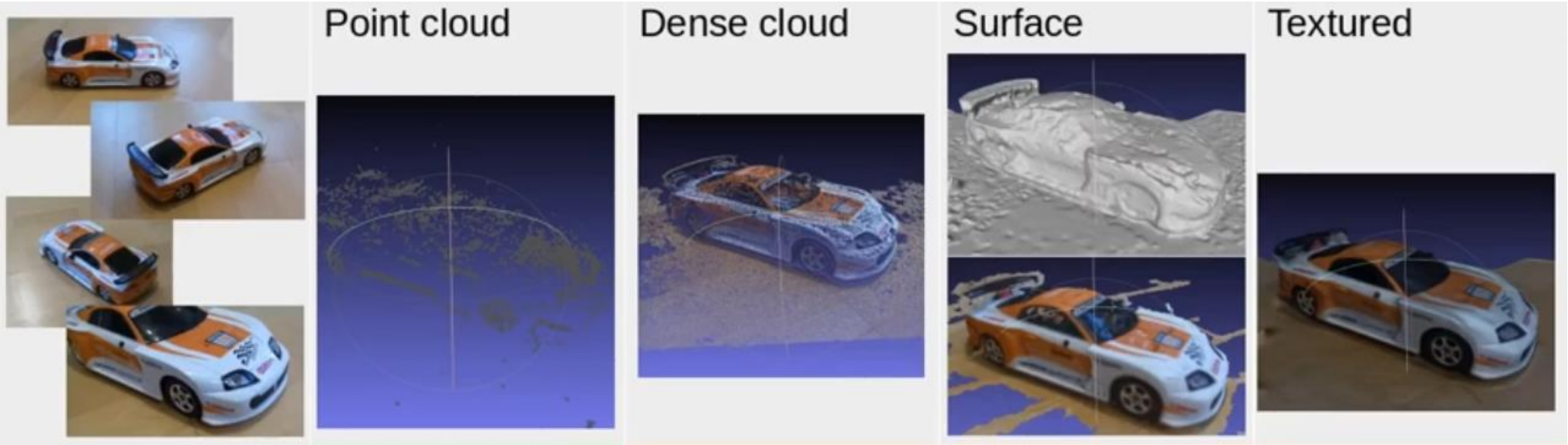
Poisson Surface Reconstruction [Kazhdan 06/13]





Typical Pipelines: SfM + MVS + Recon

- Pix4D (commercial)
- Agisoft PhotoScan (commercial)
- COLMAP (Schoenberger et al.; open source)
- …

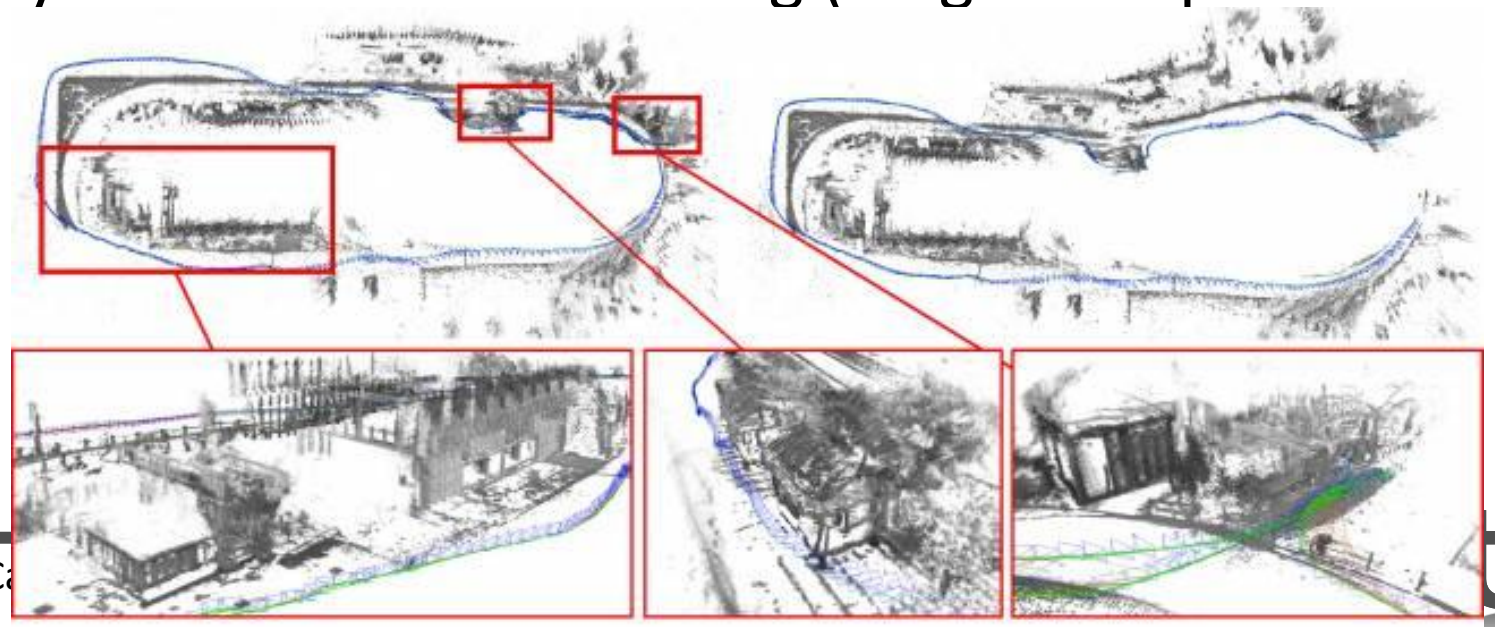


Typical Pipelines: SfM + MVS + Recon

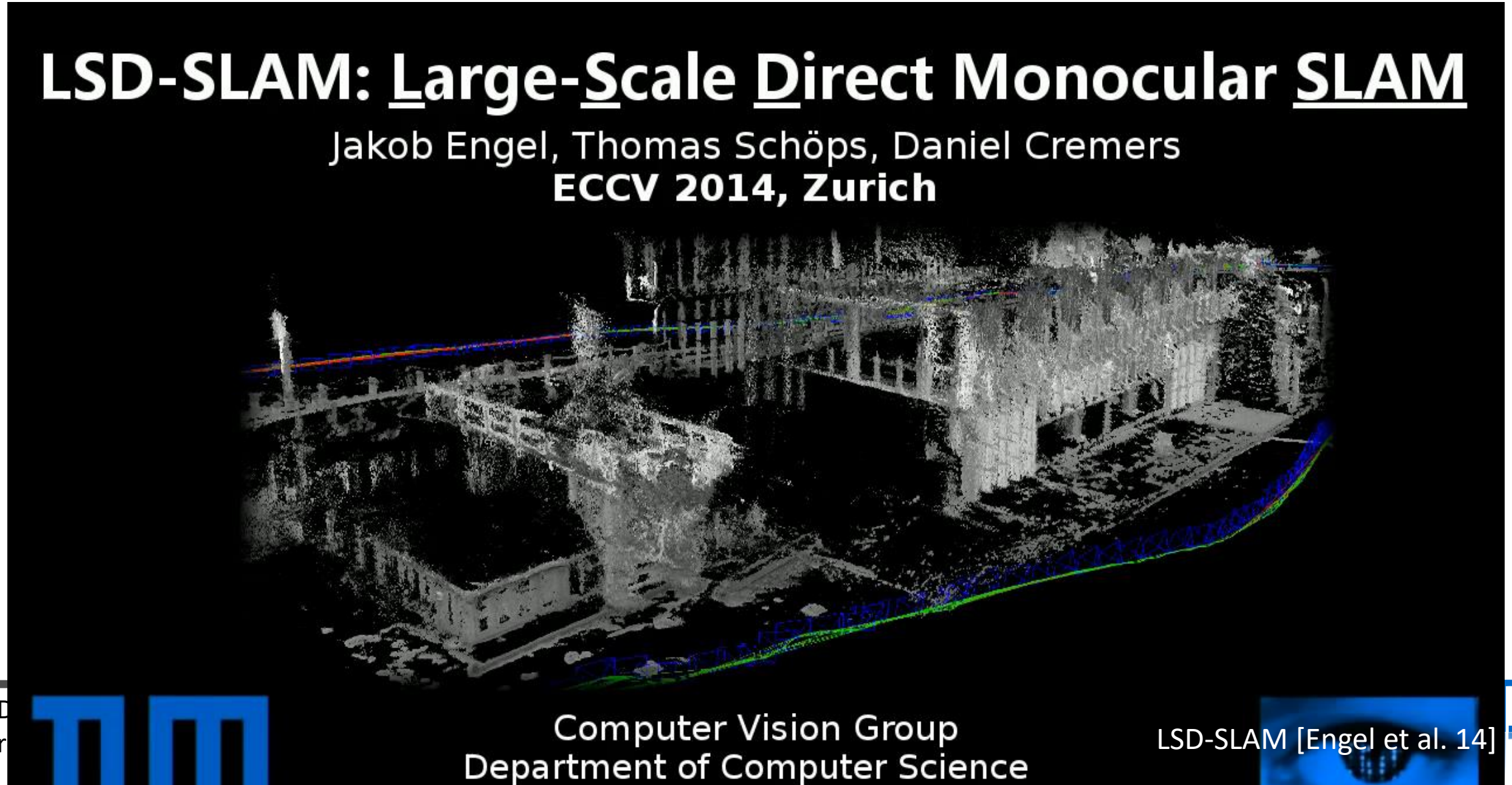
Simultaneous Localization and Mapping (SLAM)

Simultaneous Localization and Mapping (SLAM)

- Similar idea than SfM + MVS
  - But online; i.e., 'model' is built up incrementally
    - Takes real-time RGB stream (e.g., webcam)
  - Typically smaller compute budget
  - Often only frame-to-frame tracking (no global optimization)

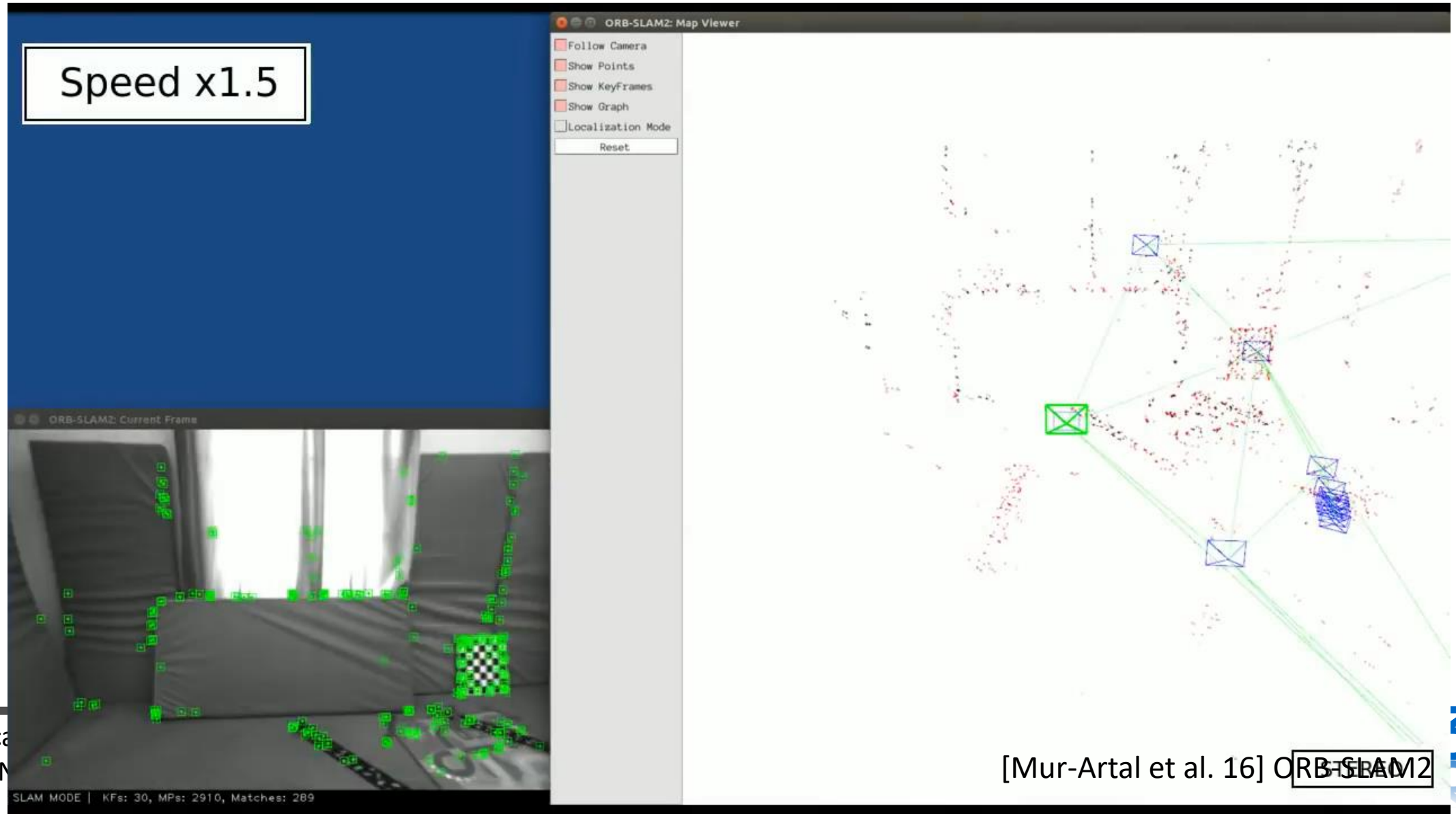


Simultaneous Localization and Mapping (SLAM)



LSD-SLAM [Engel et al. 14]

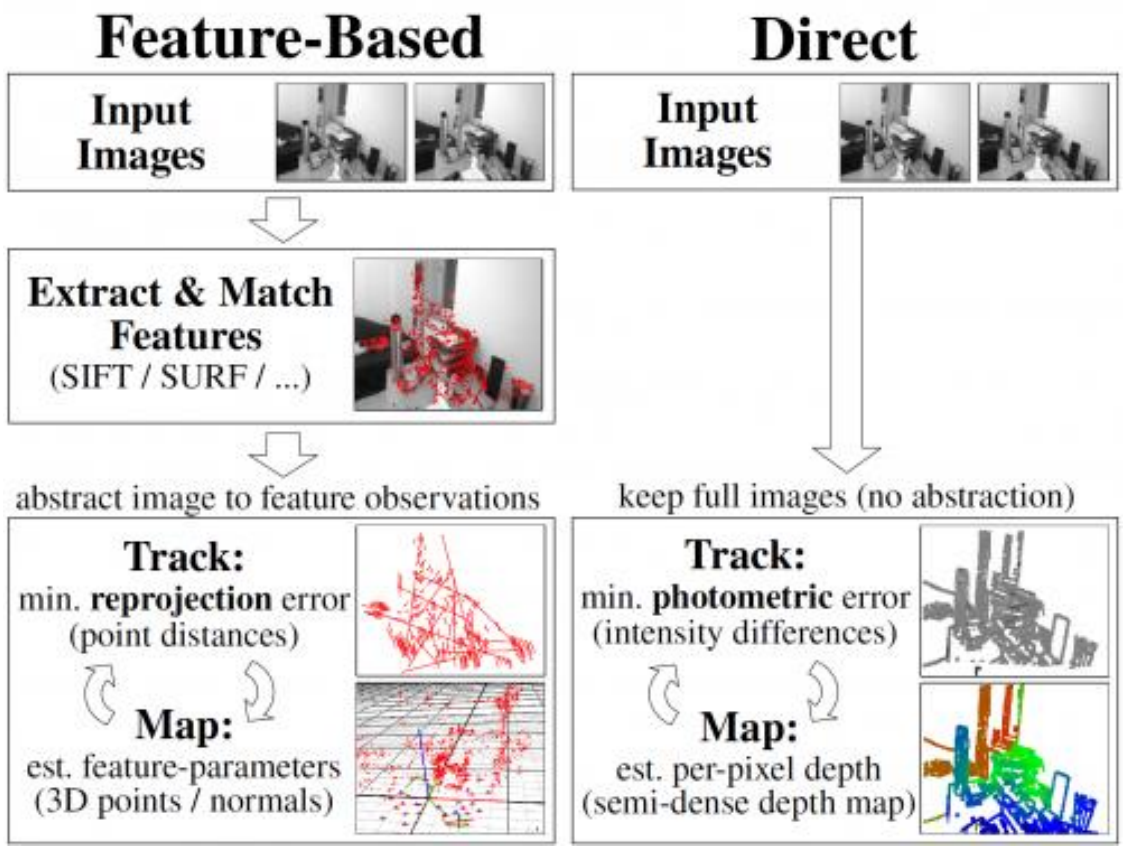
Simultaneous Localization and Mapping (SLAM)



[Mur-Artal et al. 16] ORB-SLAM2

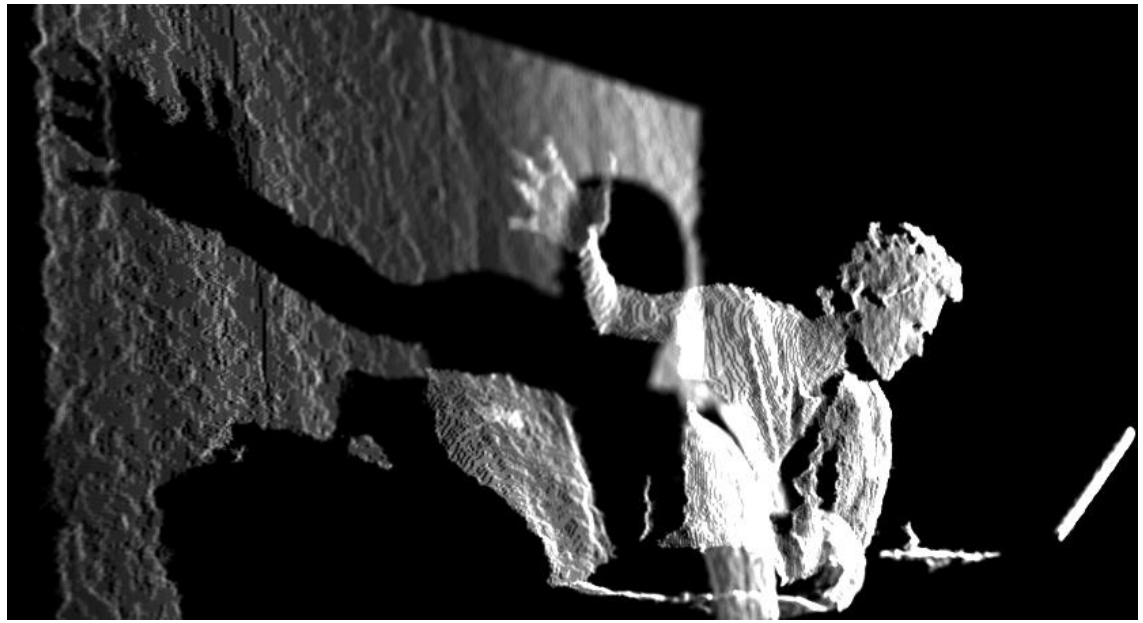
Simultaneous Localization and Mapping (SLAM)

- State-of-the-art SLAM
  - Similar to Bundle Adjustment
  - Extract keyframes (dynamically)
  - BA on local keyframe window
  - Typically ORB features (faster)
- Output
  - Poses + sparse point cloud
  - Various ‘semi-dense methods’



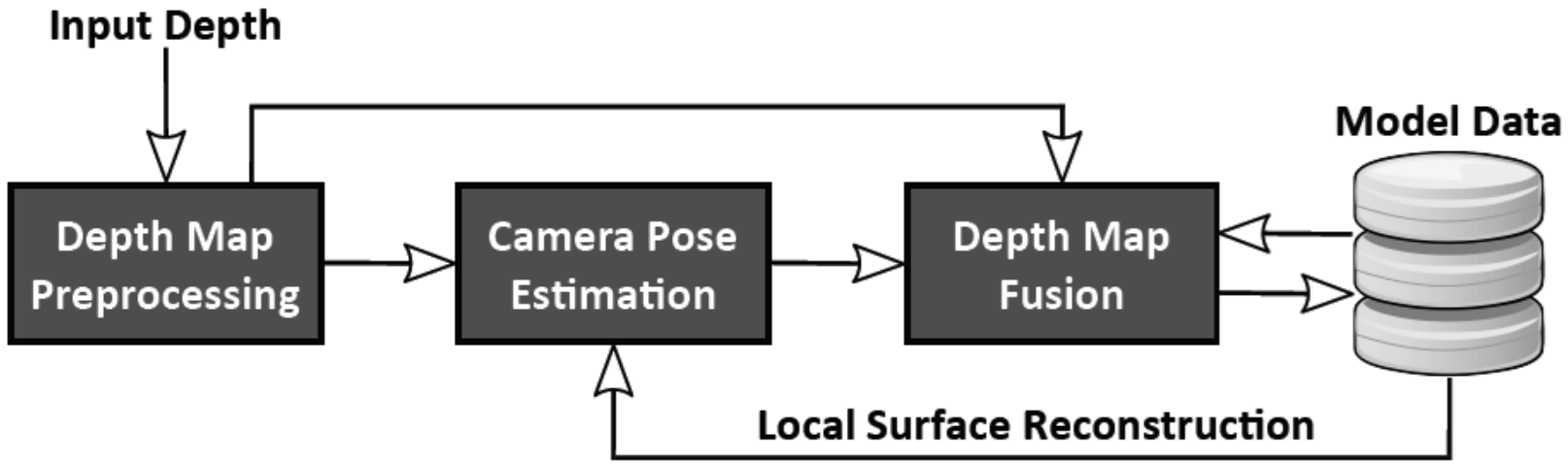
LSD-SLAM [Engel et al. 14]

RGB-D Reconstruction



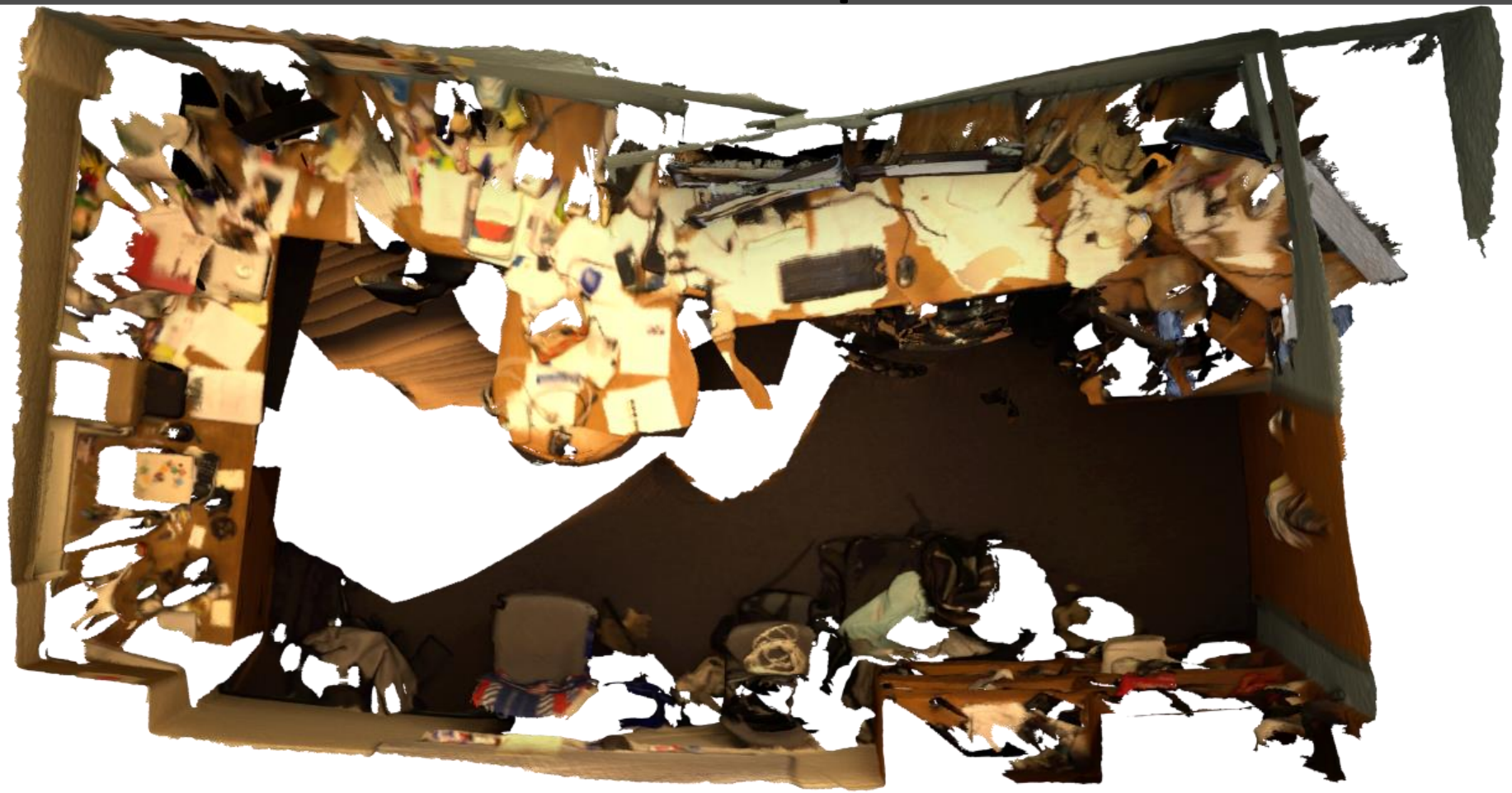
RGB-D Reconstruction

- We already have depth data (in real time)!
- Typical approach:
  - Frame-to-frame tracking (or frame-to-model)
  - Accumulate depth data in ‘model’ (implicit surface rep.)



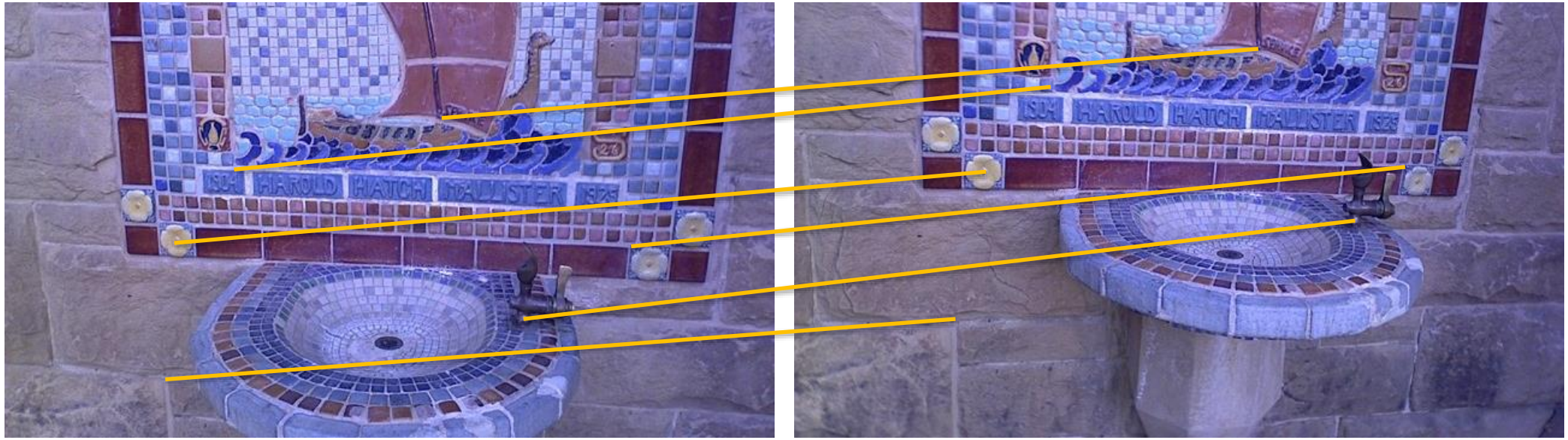
RGB-D Reconstruction: KinectFusion

RGB-D Reconstruction: Loop Closure



RGB-D Reconstruction: Loop Closure

RGB-D “Bundling”



$$E_{bundle}(T) = \sum_{i,j}^{\#frames} \sum_{k}^{\#corresp.} \left\| T_i p_{ik} - T_j p_{jk} \right\|_2^2$$

RGB-D “Bundling”

$$E_{bundle}(T) = \sum_{i,j}^{\#frames} \sum_{k}^{\#corresp.} \left\| T_i p_{ik} - T_j p_{jk} \right\|_2^2$$

$$E_{depth}(T) = \sum_{i,j}^{\#\text{frames}} \sum_{k}^{\#\text{pixels}} \left\| \left(p_k - T_i^{-1} T_j \pi_d^{-1} \left(D_j \left(\pi_d \left(T_j^{-1} T_i p_k \right) \right) \right) \right) \cdot n_k \right\|_2^2$$

$$E_{color}(T) = \sum_{i,j}^{\#\text{frames}} \sum_{k}^{\#\text{pixels}} \left\| \nabla I \left(\pi_c (p_k) \right) - \nabla I \left(\pi_c \left(T_j^{-1} T_i p_k \right) \right) \right\|_2^2$$

Next Lecture: Optimization

- How do we solve these non-linear terms?

$$E_{re-proj}(\boldsymbol{T}, \boldsymbol{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \left|\left| x_{ij} - \pi_i(T_i \cdot X_j) \right|\right|_2^2$$

$$E_{keypoint}(T) = \sum_{i,j}^{\#frames} \sum_{k}^{\#corresp.} \left\| T_i p_{ik} - T_j p_{jk} \right\|_2^2$$

Some Material

- Excellent Reading list on 3DReconstruction (by Pierre Moulon)
  https://github.com/openMVG/awesome_3DReconstruction_list

Administrative

- Reading Homework:
  - Structure-from-Motion Revisited [Schonberger et al. 16]
  - Run Colmap on toy dataset: https://colmap.github.io/install.html
  - LSD-SLAM: Large-Scale Direct Monocular SLAM https://jakobengel.github.io/pdf/engel14eccv.pdf

- Next week:
  - Optimization Methods for 3D Reconstruction

Administrative

See you next week!