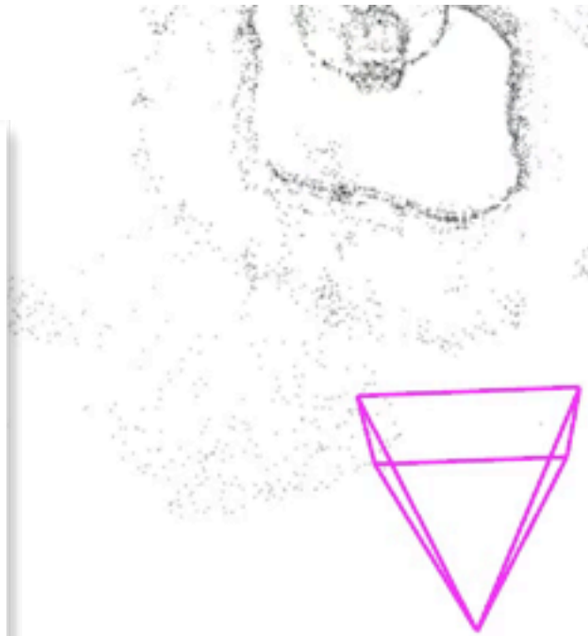
Image-Based Localization

Torsten Sattler
Computer Vision & Geometry Group, ETH Zurich

Image-Based Localization

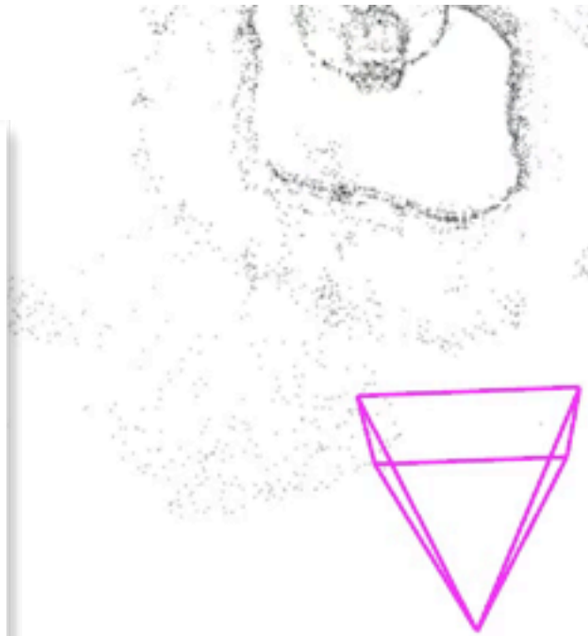




Compute **exact position and orientation** of query image relative to 3D scene model.

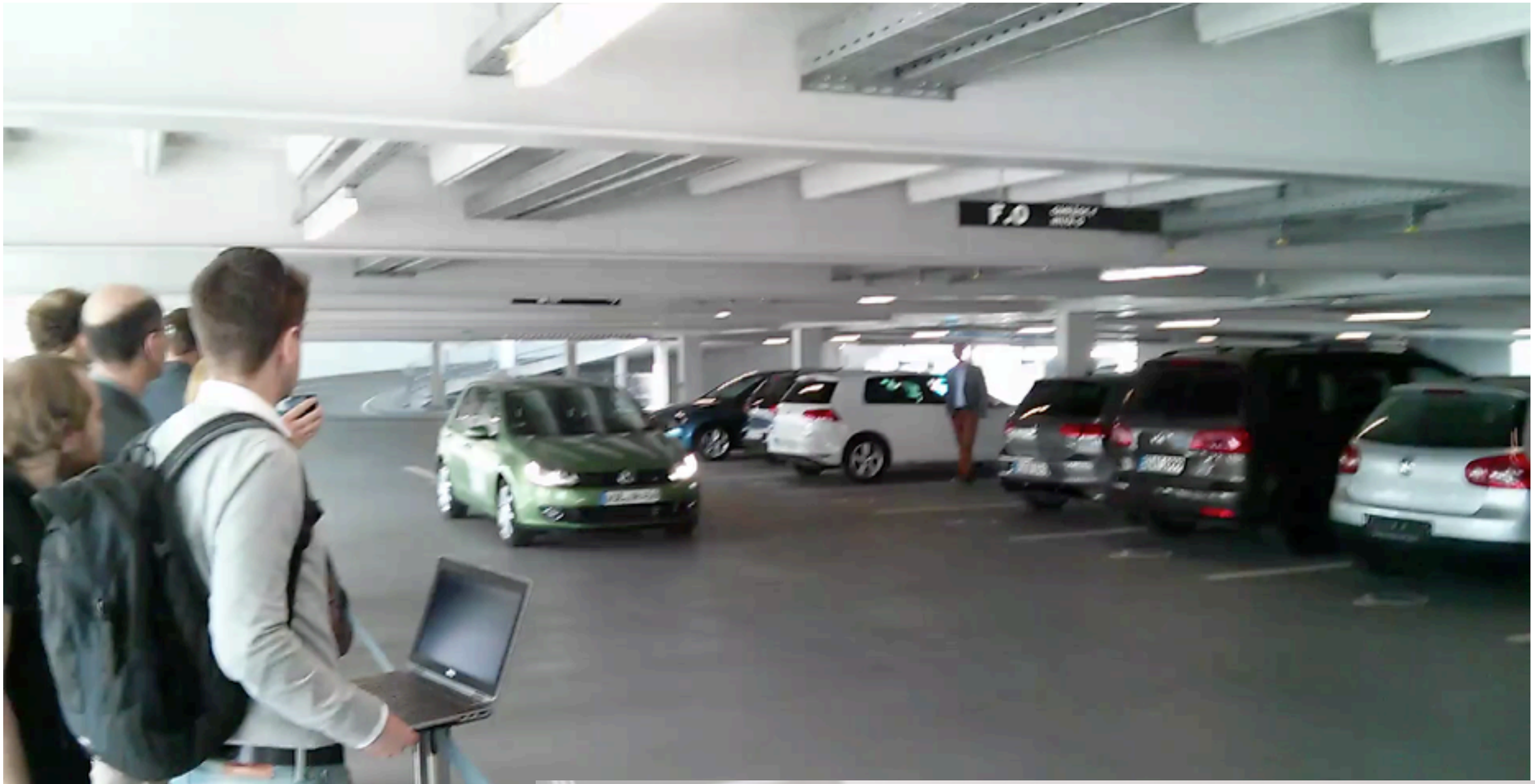
Image-Based Localization





Compute **exact position and orientation** of query image relative to 3D scene model.

Autonomous Navigation

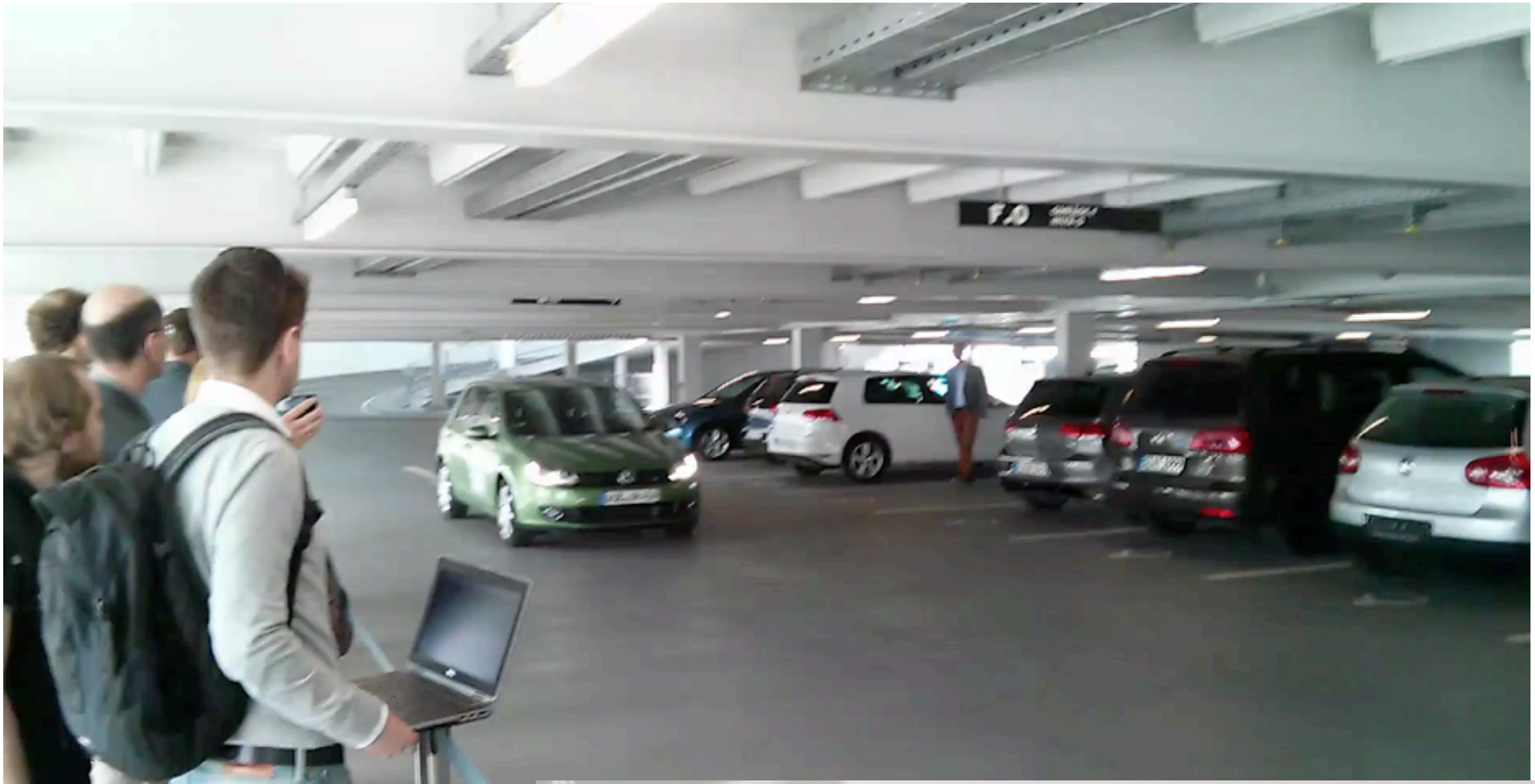


V-CHARGE

V-Charge

Automated Valet Parking and Charging for e-Mobility
Collaborative Project no. FP7-269916

Autonomous Navigation



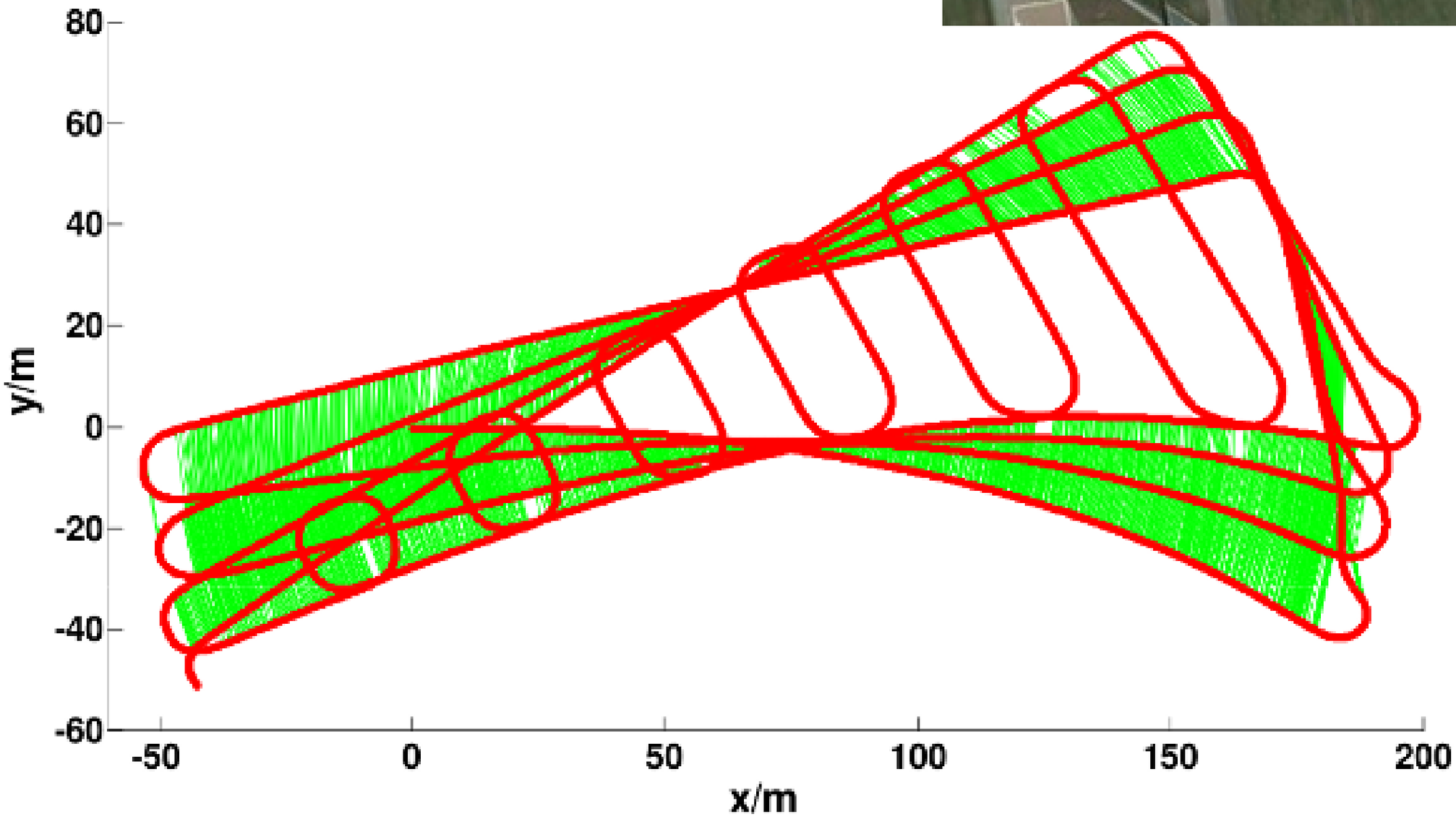
V-CHARGE

V-Charge

Automated Valet Parking and Charging for e-Mobility
Collaborative Project no. FP7-269916

Loop Closure for SLAM





[Lee, Pollefeys. CVPR'14]

Accurate Geo-Tagging for Holiday Photos



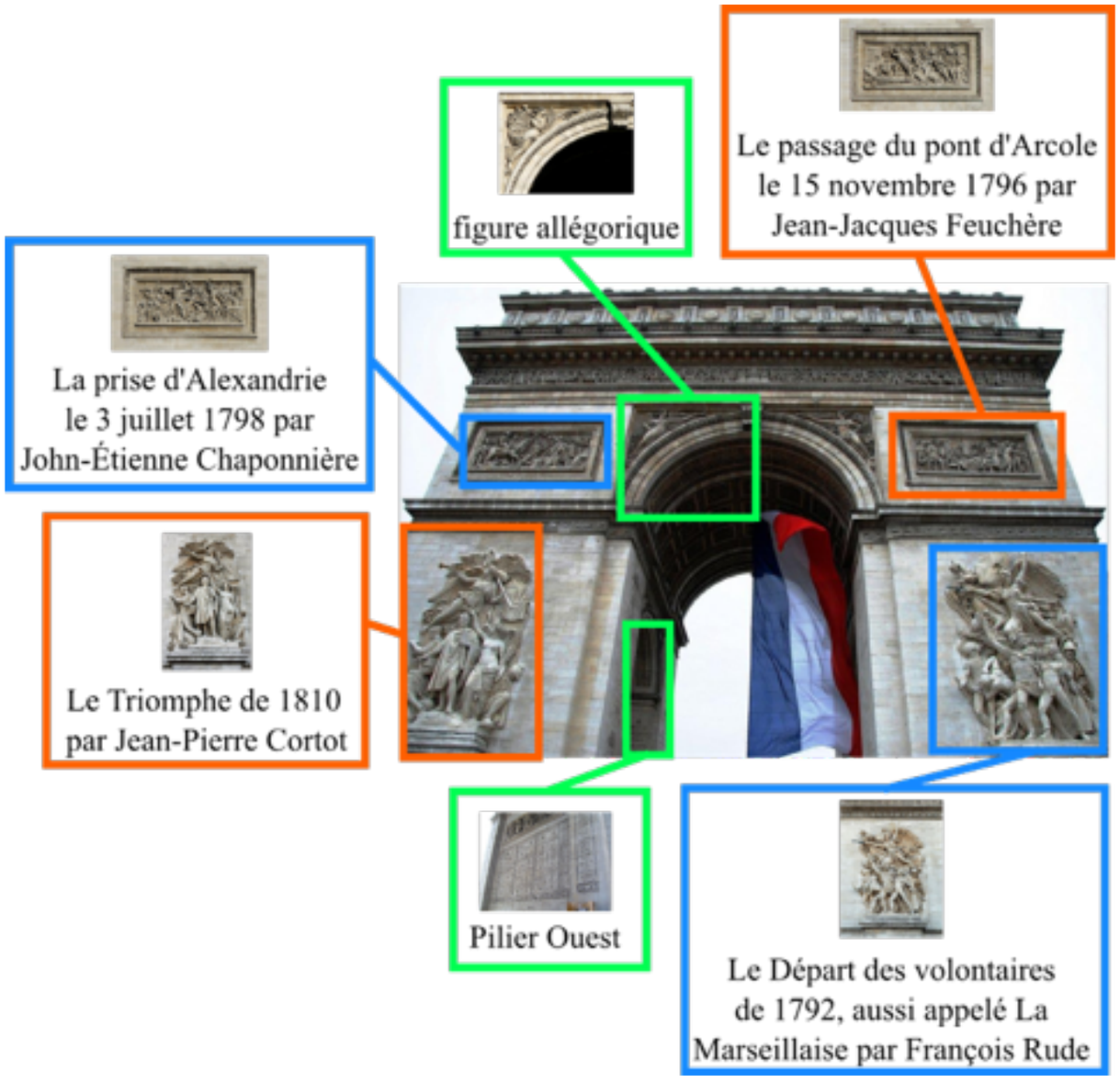
[Li et al., ECCV'12] http://landmark.cs.cornell.edu/

Accurate Geo-Tagging for Holiday Photos



[Li et al., ECCV'12] http://landmark.cs.cornell.edu/

Scene Detail Discovery for Touristic Applications





[Weyand, Leibe, ICCV'13]

Augmented Reality



[Middelberg et al., ECCV'14]

Augmented Reality



[Middelberg et al., ECCV'14]

Image-Based Localization Pipeline





Image-Based Localization Pipeline



Extract Local Features



Image-Based Localization Pipeline



Extract Local Features

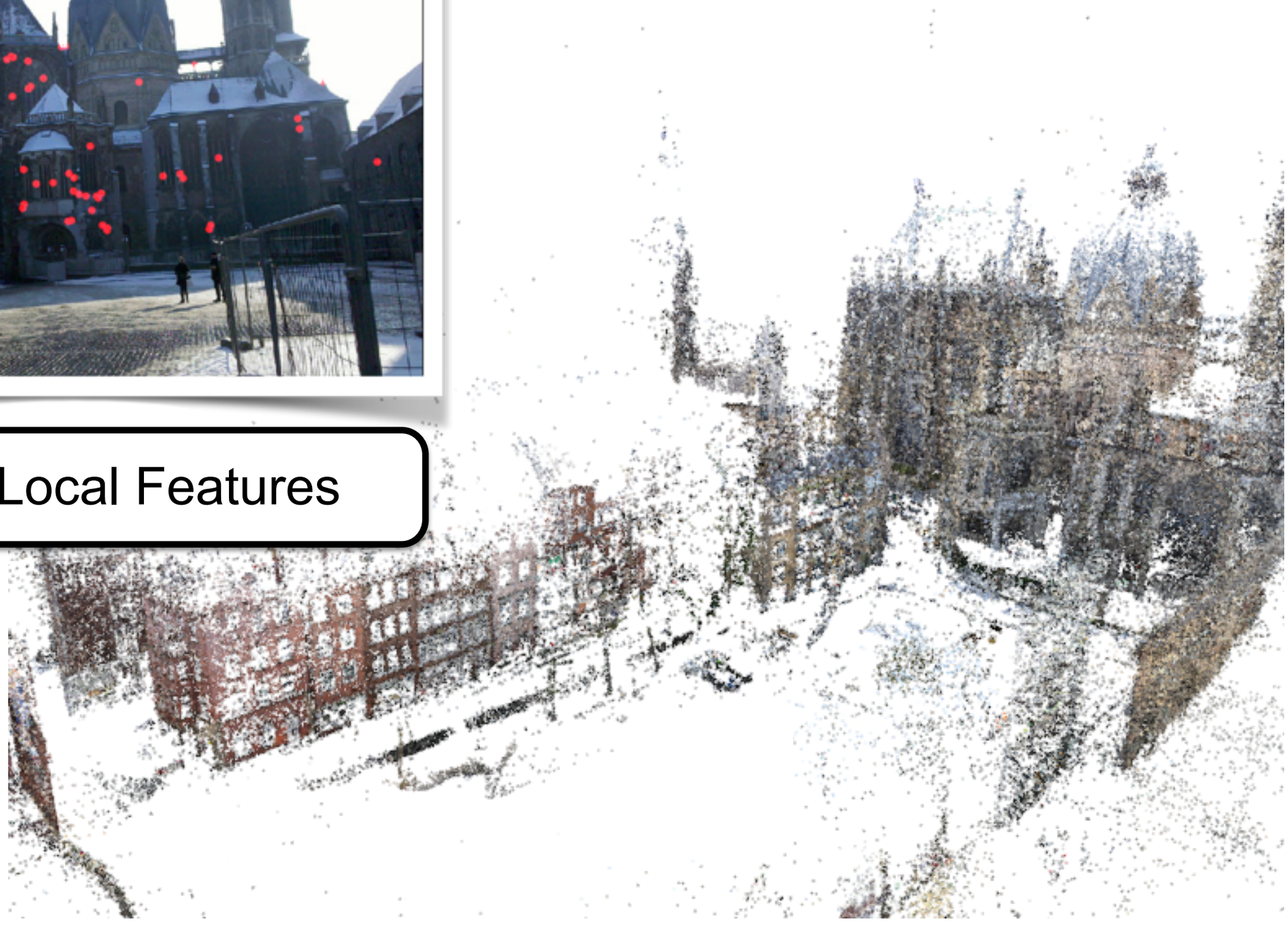


Image-Based Localization Pipeline



Extract Local Features

Establish 2D-3D Matches

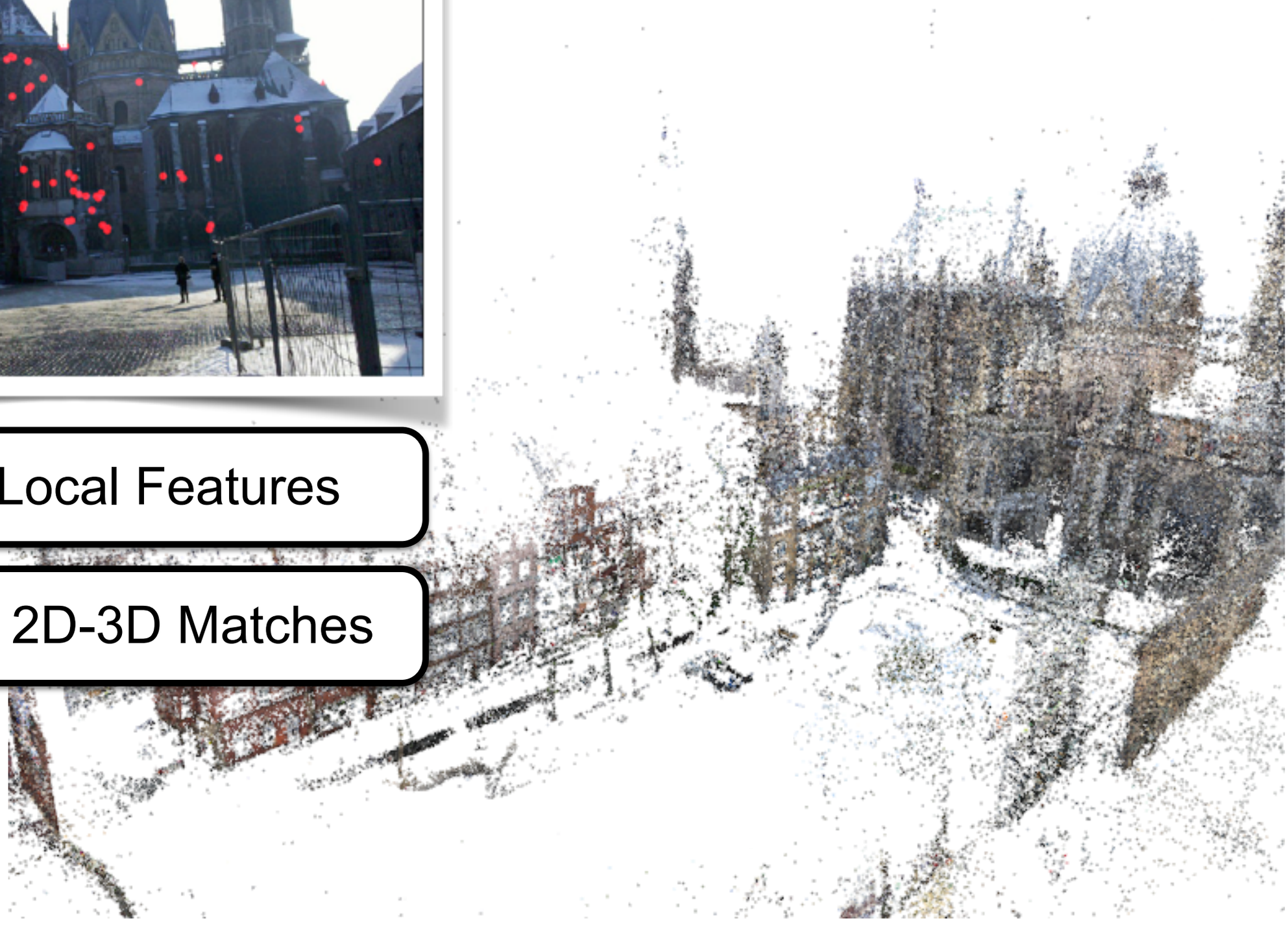


Image-Based Localization Pipeline



Extract Local Features

Establish 2D-3D Matches

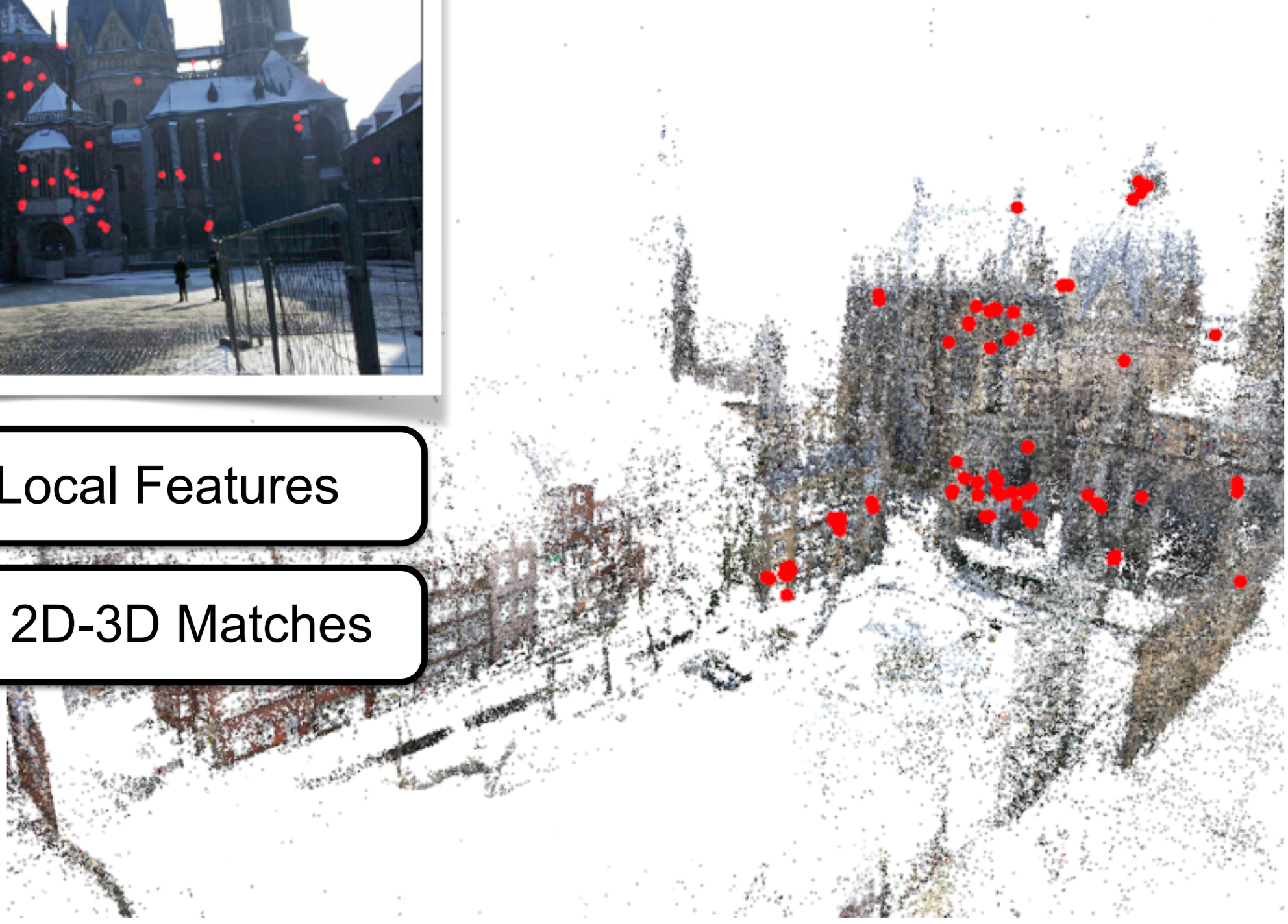


Image-Based Localization Pipeline



Extract Local Features

Establish 2D-3D Matches

Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

Image-Based Localization Pipeline



Extract Local Features

Establish 2D-3D Matches

Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

Challenges

- **Efficiency**: Quickly localize query images

Challenges

- **Efficiency**: Quickly localize query images
- **Effectiveness**: Localize all query images

Challenges

- **Efficiency**: Quickly localize query images

- **Effectiveness**: Localize all query images

- **Accuracy**: Accurately recover camera pose

Overview

- Basics
- Localization using Prioritized Matching
- Scalable Image-Based Localization
- Conclusion / Open Challenges

Overview

- Basics
- Localization using Prioritized Matching
- Scalable Image-Based Localization
- Conclusion / Open Challenges

Fundamentals

Structure-from-Motion

Establish 2D-3D Matches

Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

(This is meant as a short overview, no in-depth discussion)

Fundamentals

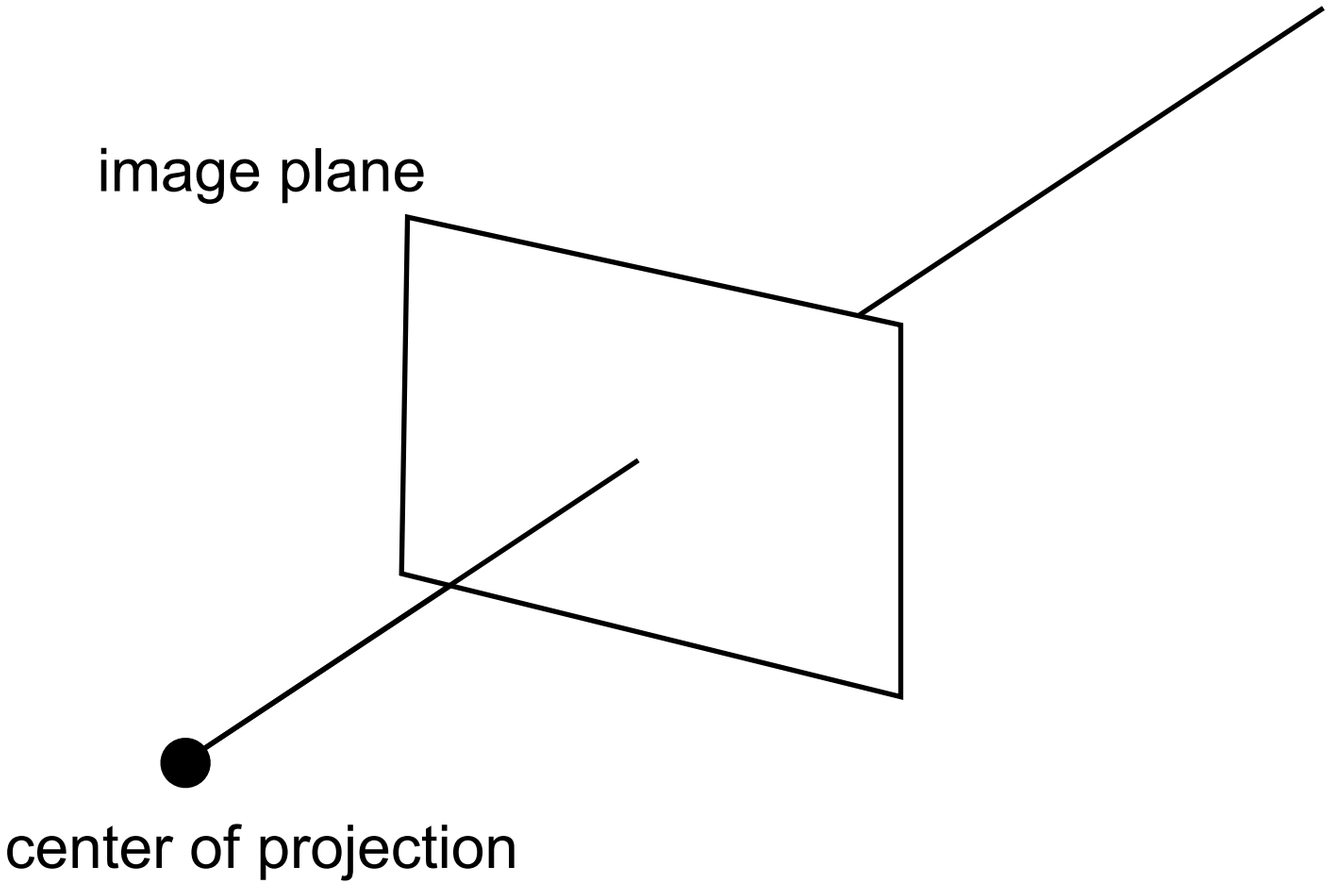
Structure-from-Motion

Establish 2D-3D Matches

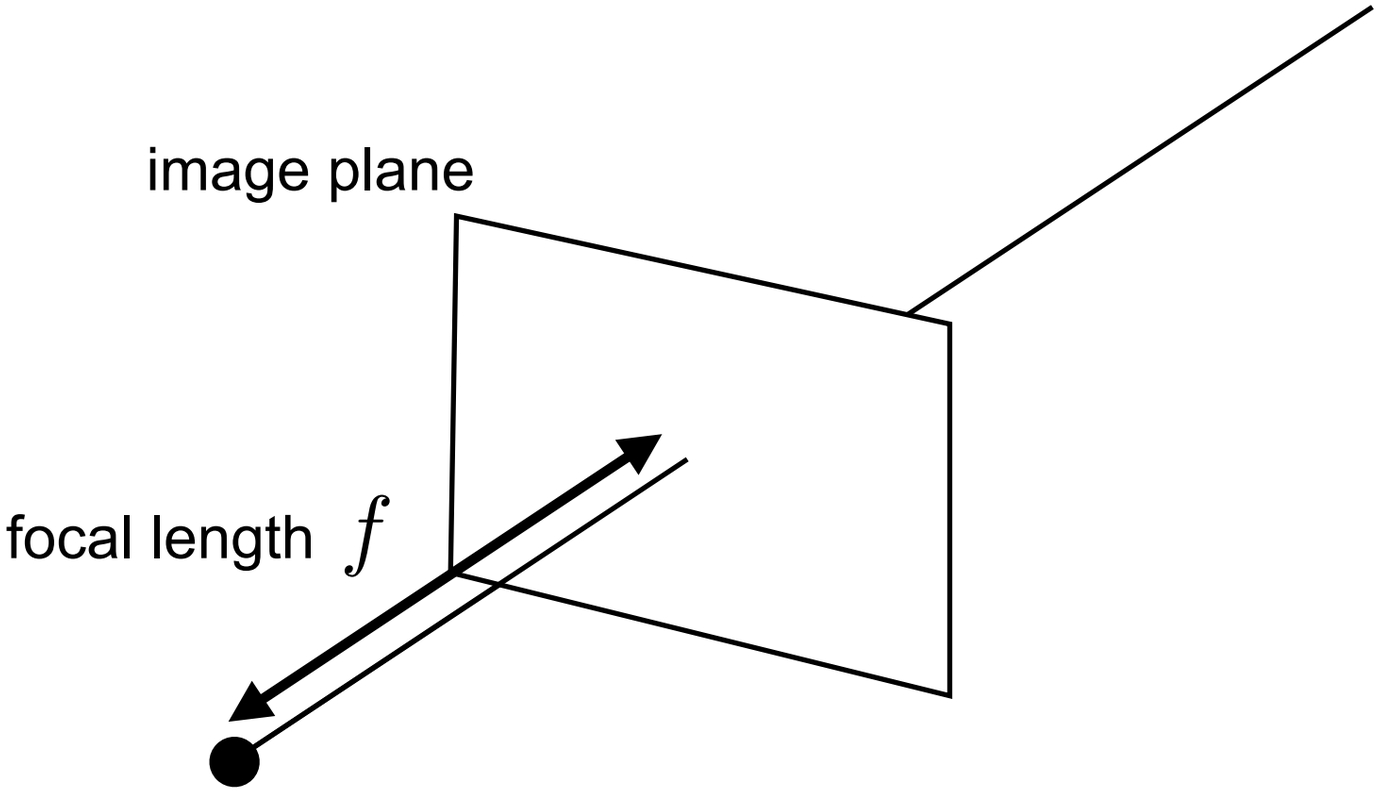
Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

(This is meant as a short overview, no in-depth discussion)

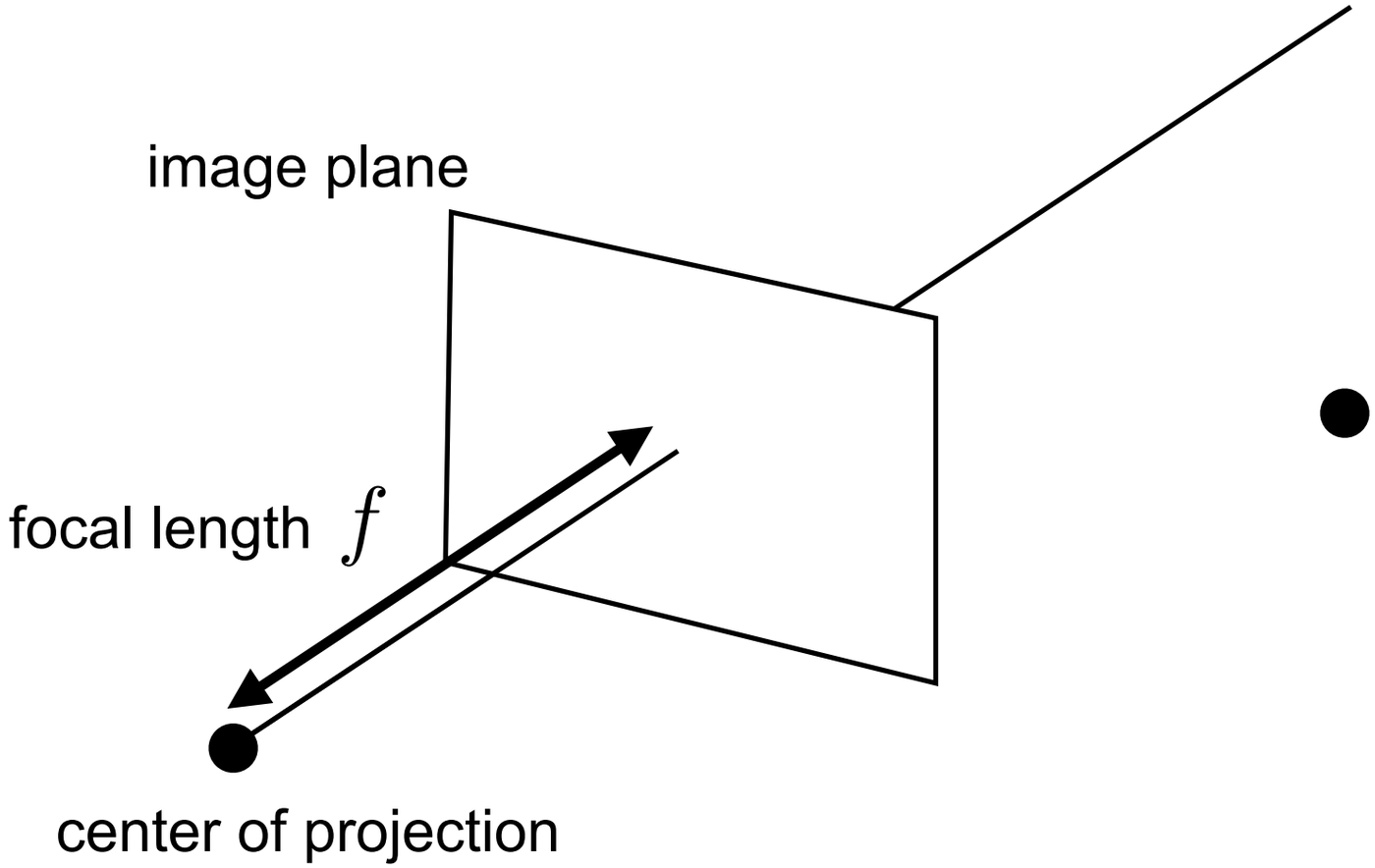
Pinhole Camera Model



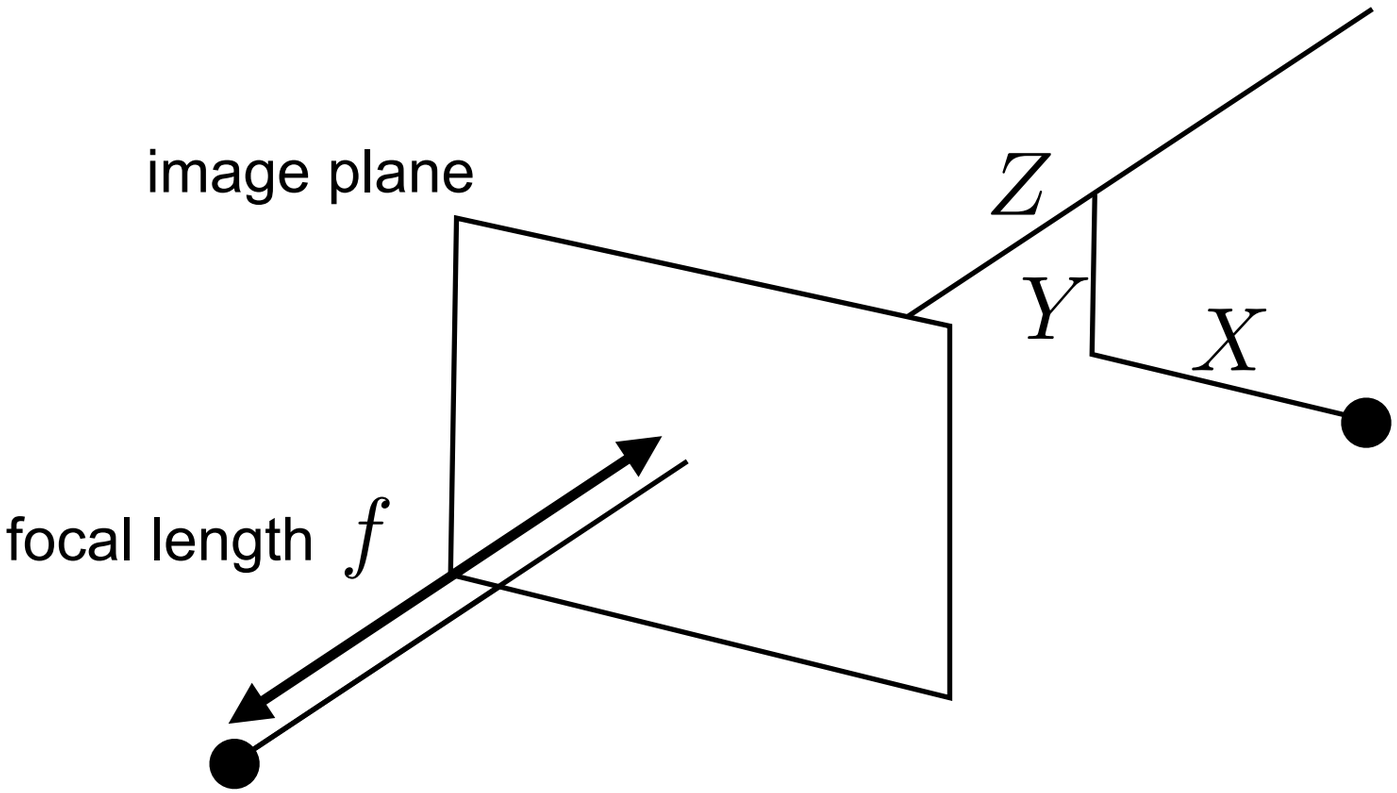
Pinhole Camera Model



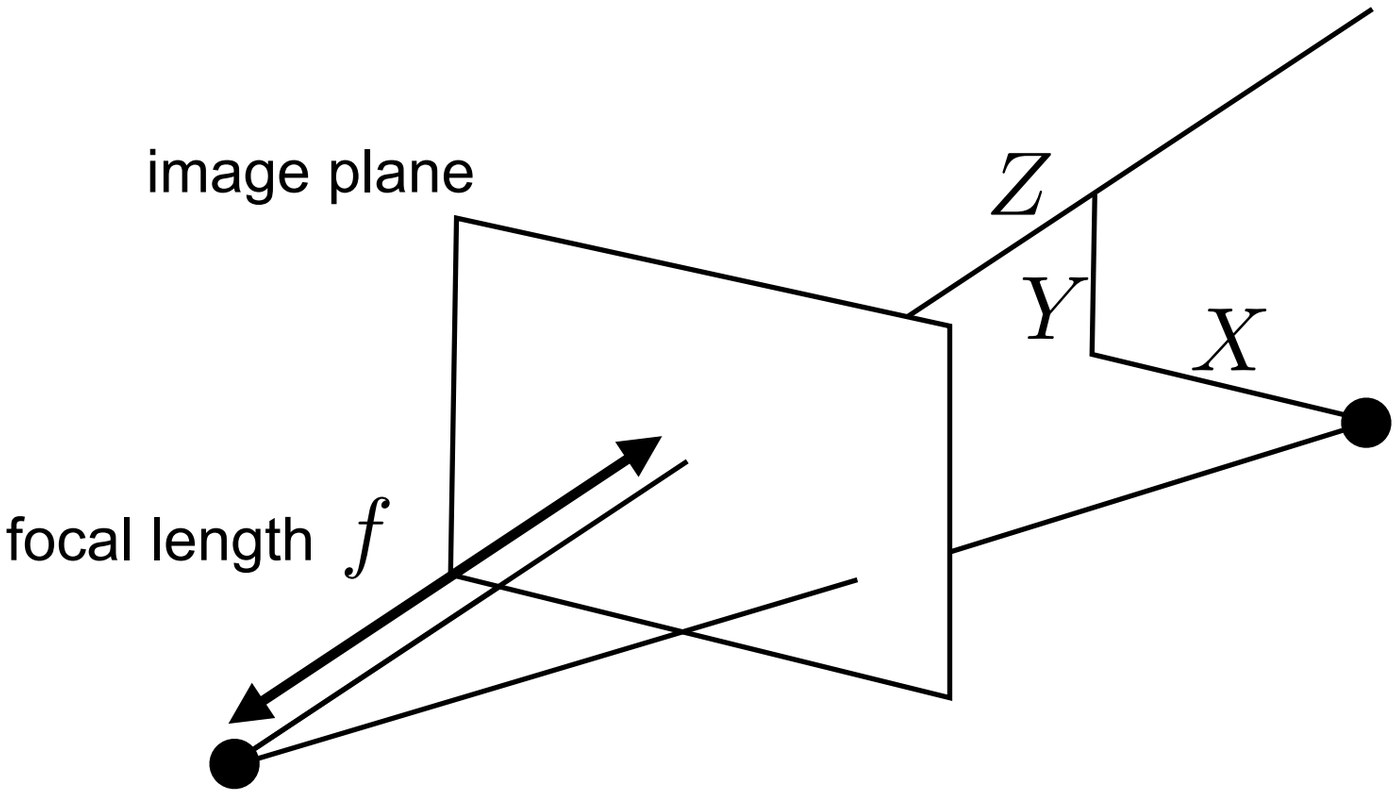
Pinhole Camera Model



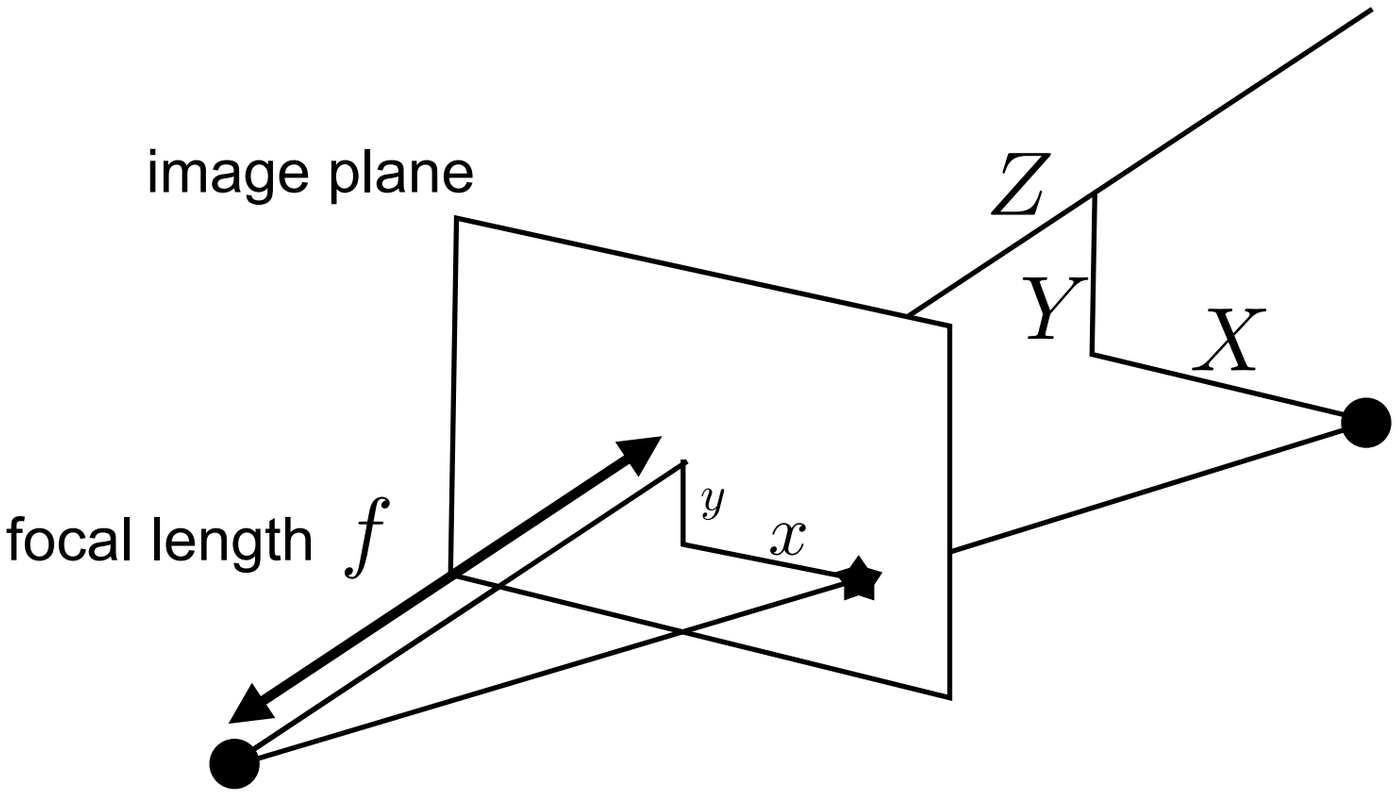
Pinhole Camera Model



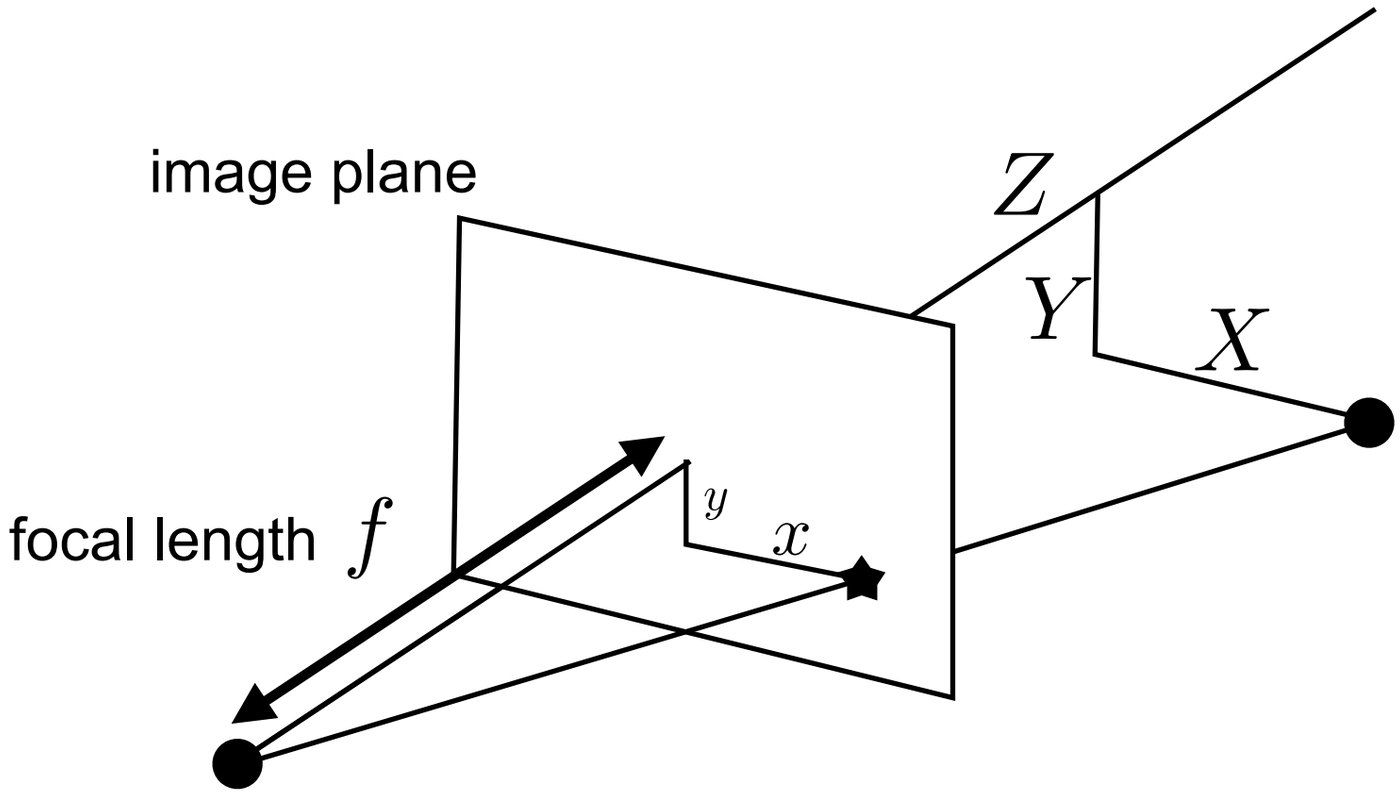
Pinhole Camera Model



Pinhole Camera Model



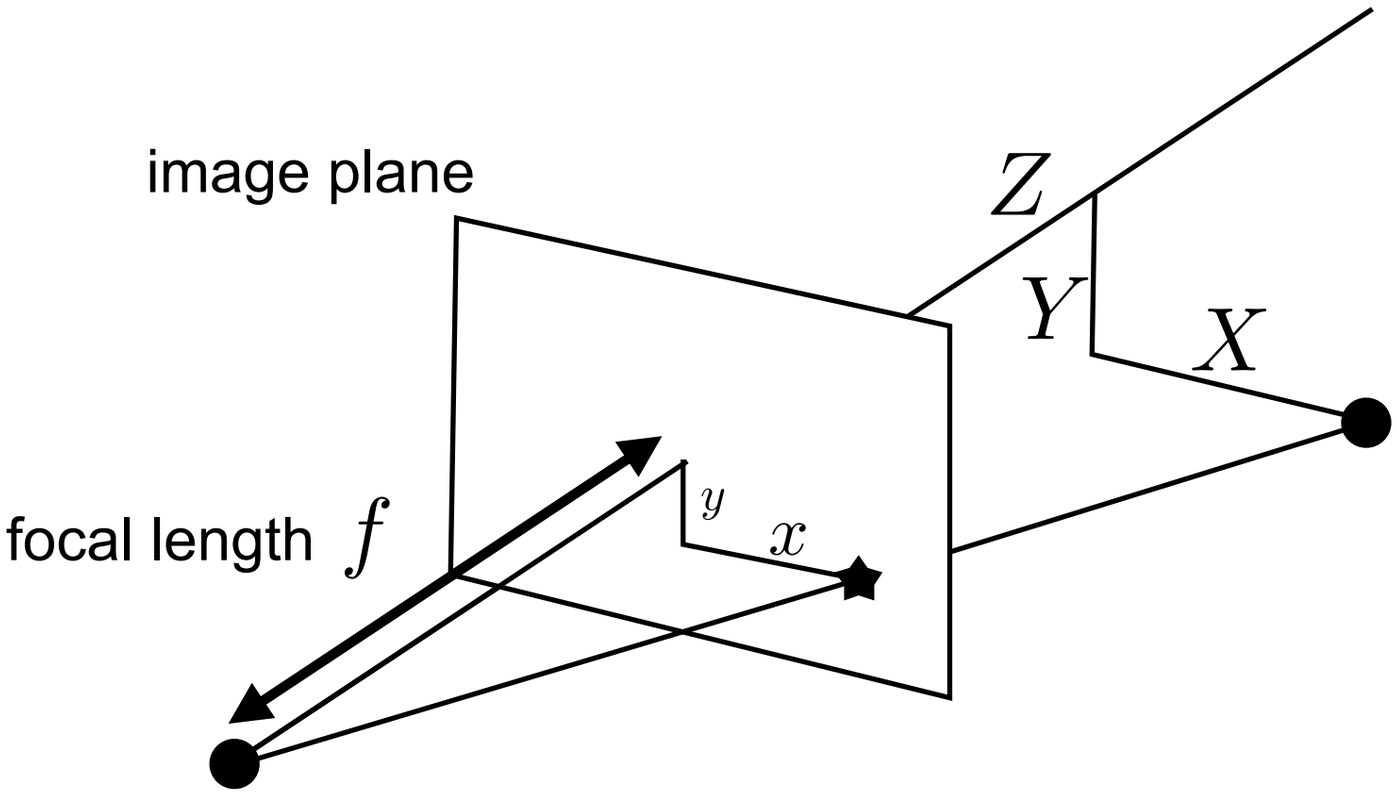
Pinhole Camera Model



Central projection:

$$\frac{x}{f} = \frac{X}{Z}$$

Pinhole Camera Model

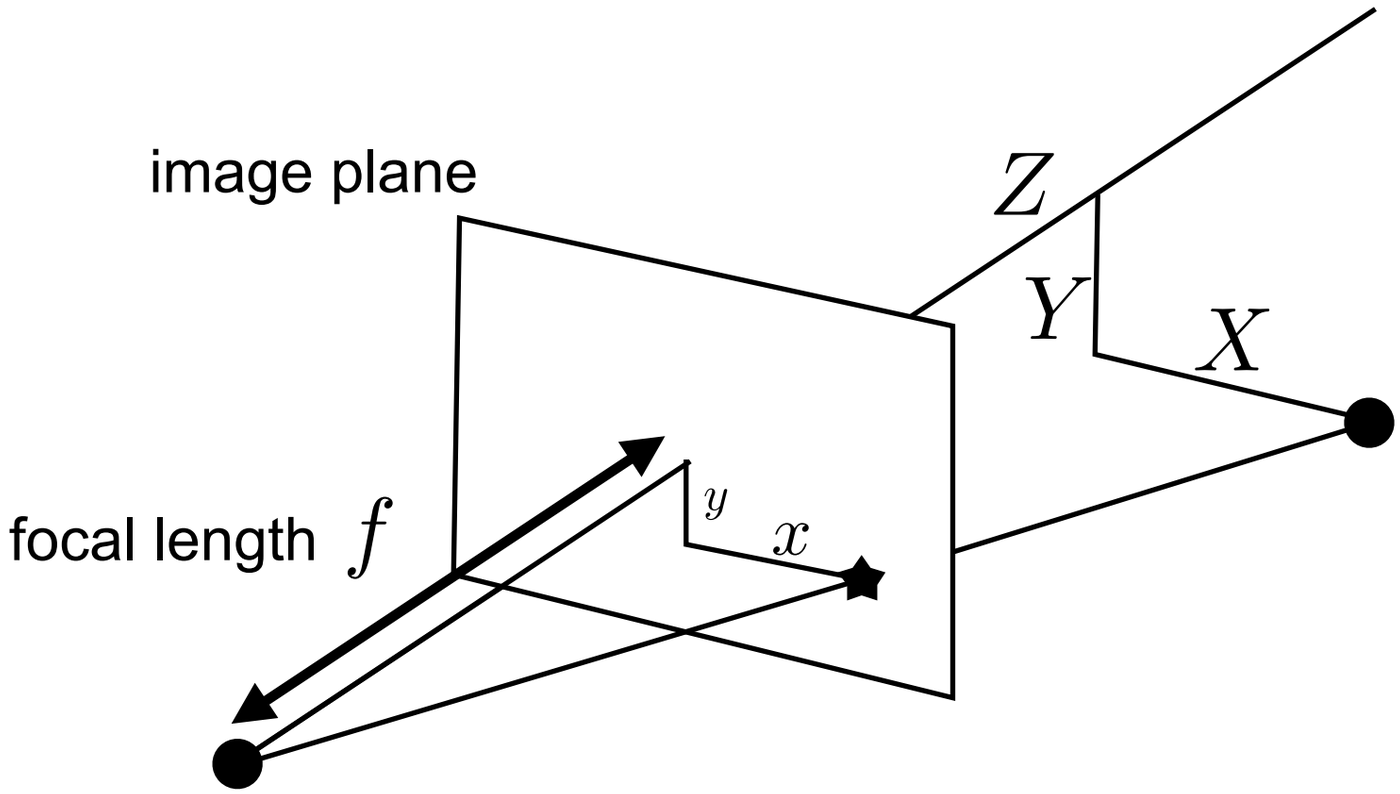


Central projection:

$$\frac{x}{f} = \frac{X}{Z}$$

- In homogeneous coordinates: $\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} fX/Z \\ fY/Z \\ 1 \end{pmatrix} = \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix}$

Pinhole Camera Model



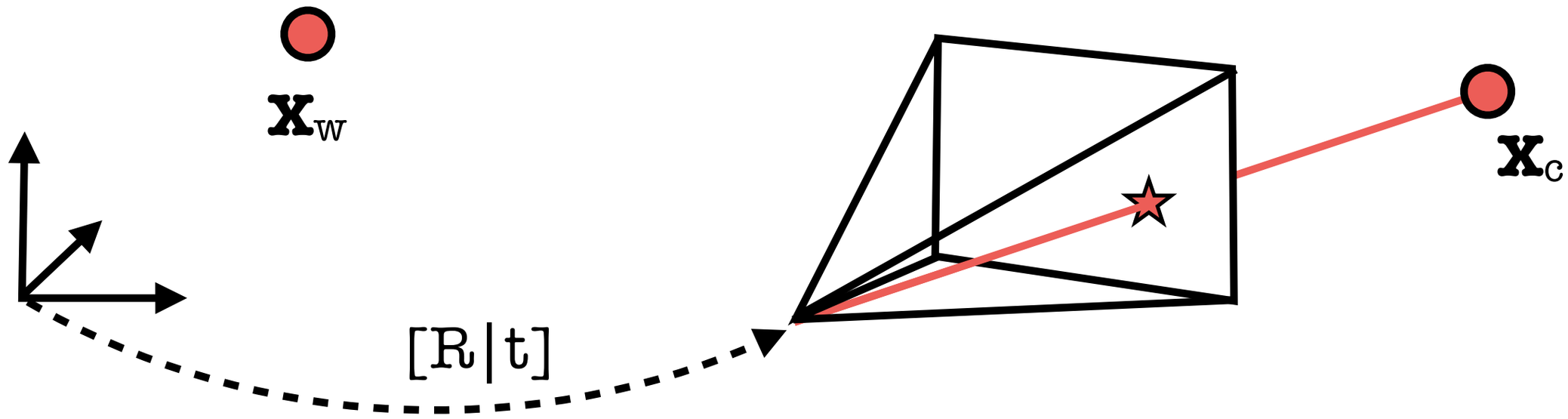
Central projection:

$$\frac{x}{f} = \frac{X}{Z}$$

- In homogeneous coordinates: $\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} fX/Z \\ fY/Z \\ 1 \end{pmatrix} = \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix}$

- Projection: $\alpha \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathtt{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$

The Camera Pose



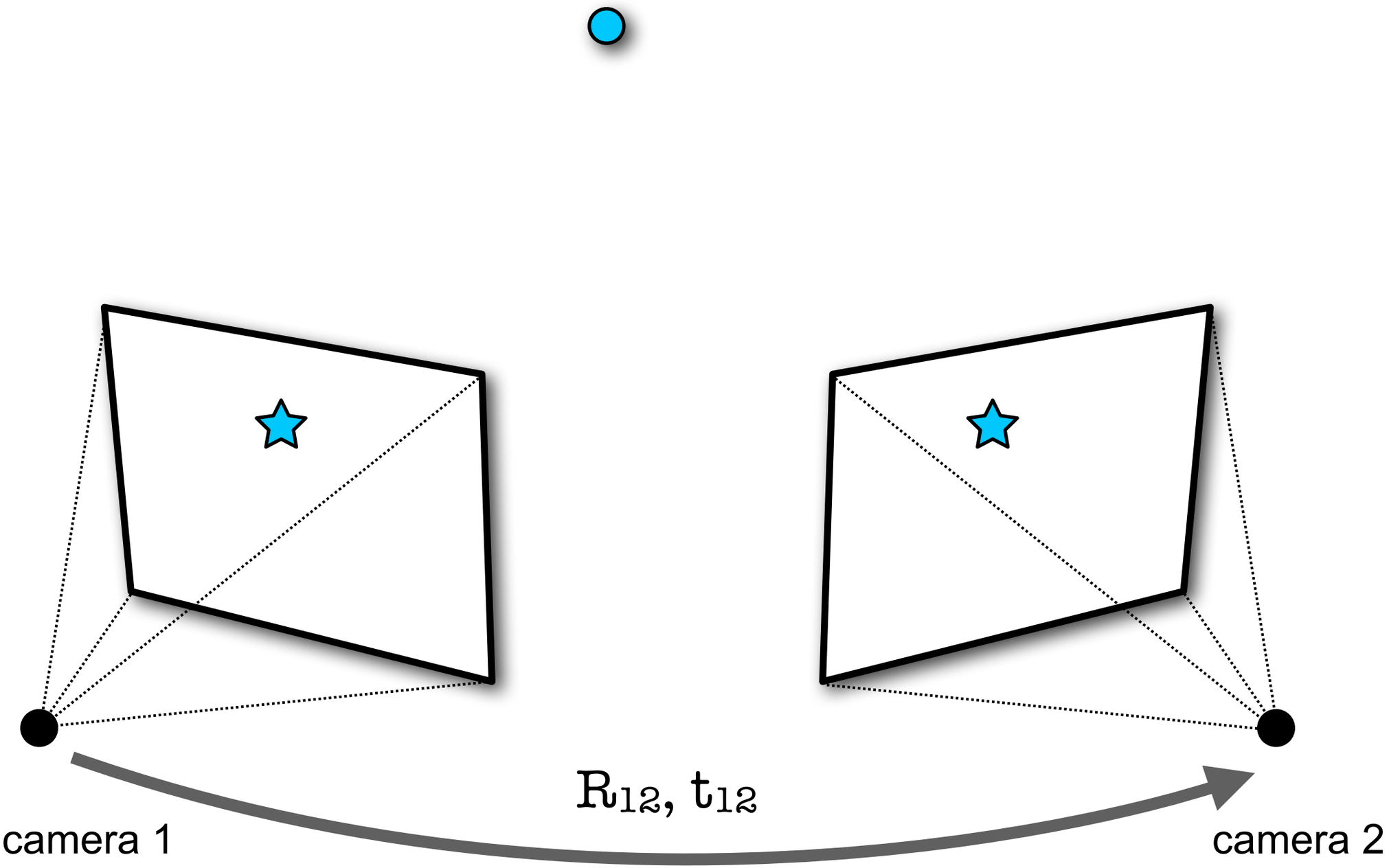
world coordinates

camera coordinates

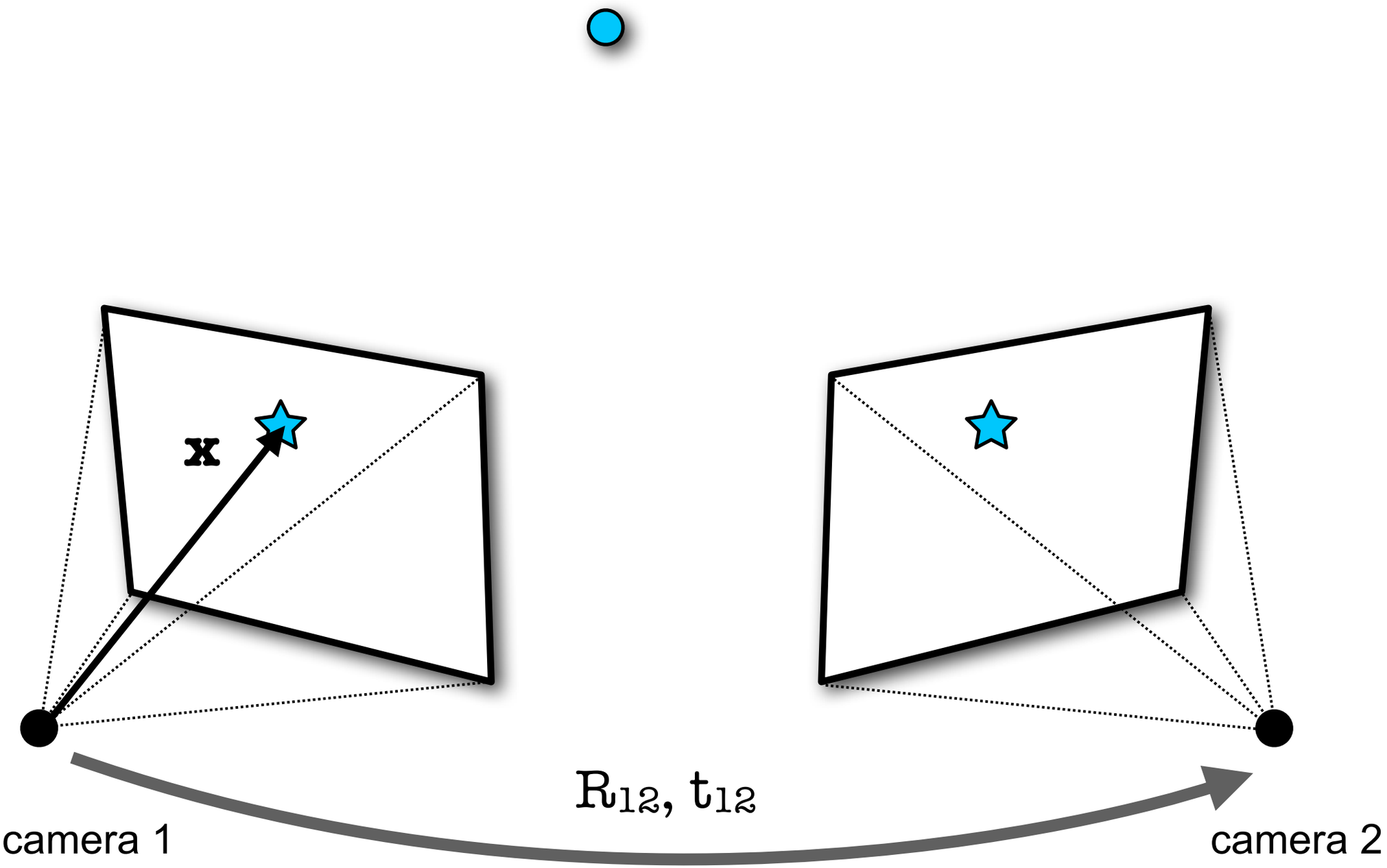
- Camera pose: Rotation $\mathtt{R}$, translation $\mathtt{t}$
- Transforms global to camera coordinates:

$$\begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} = [\mathtt{R} \mid \mathtt{t}] \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

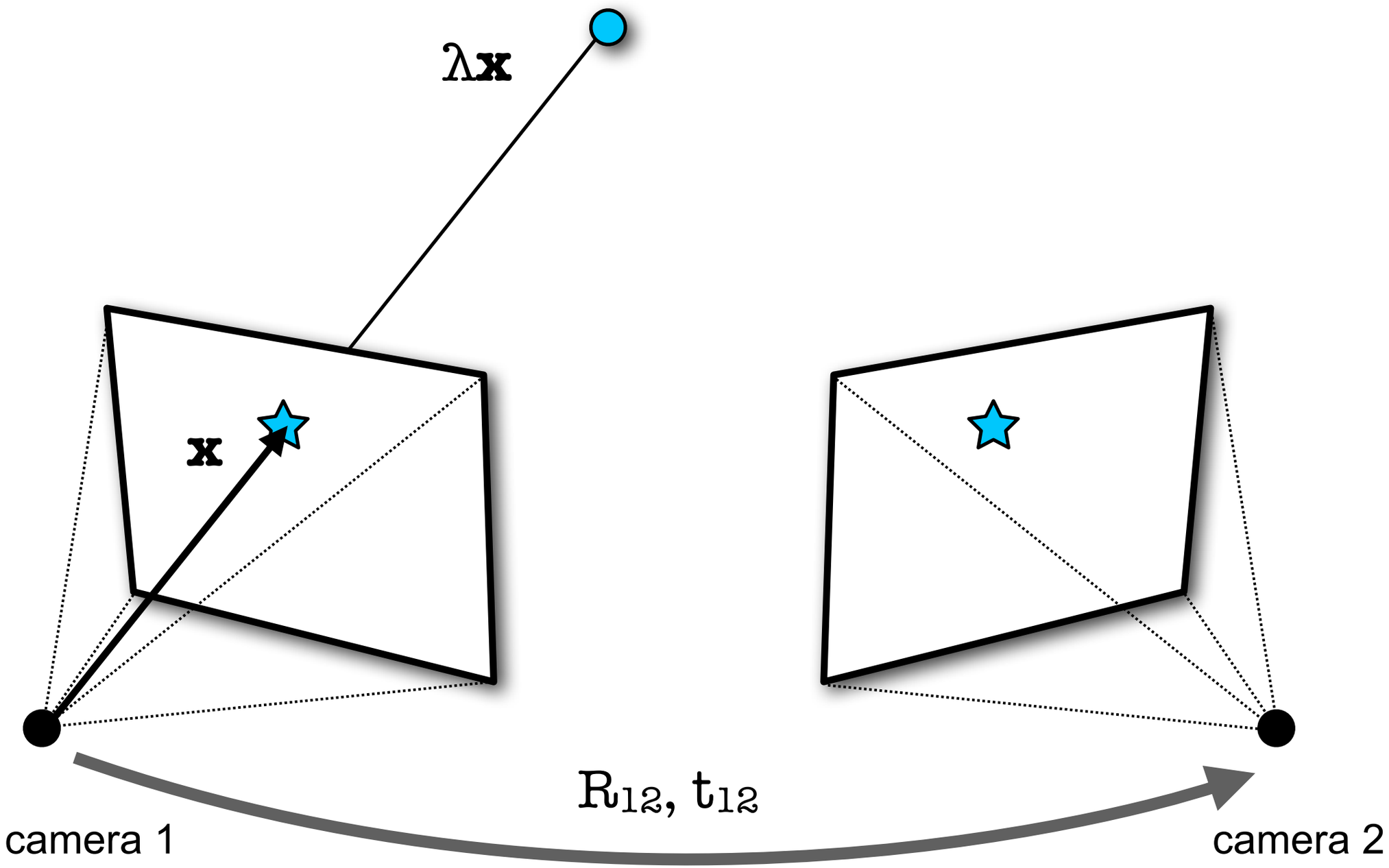
Epipolar Geometry



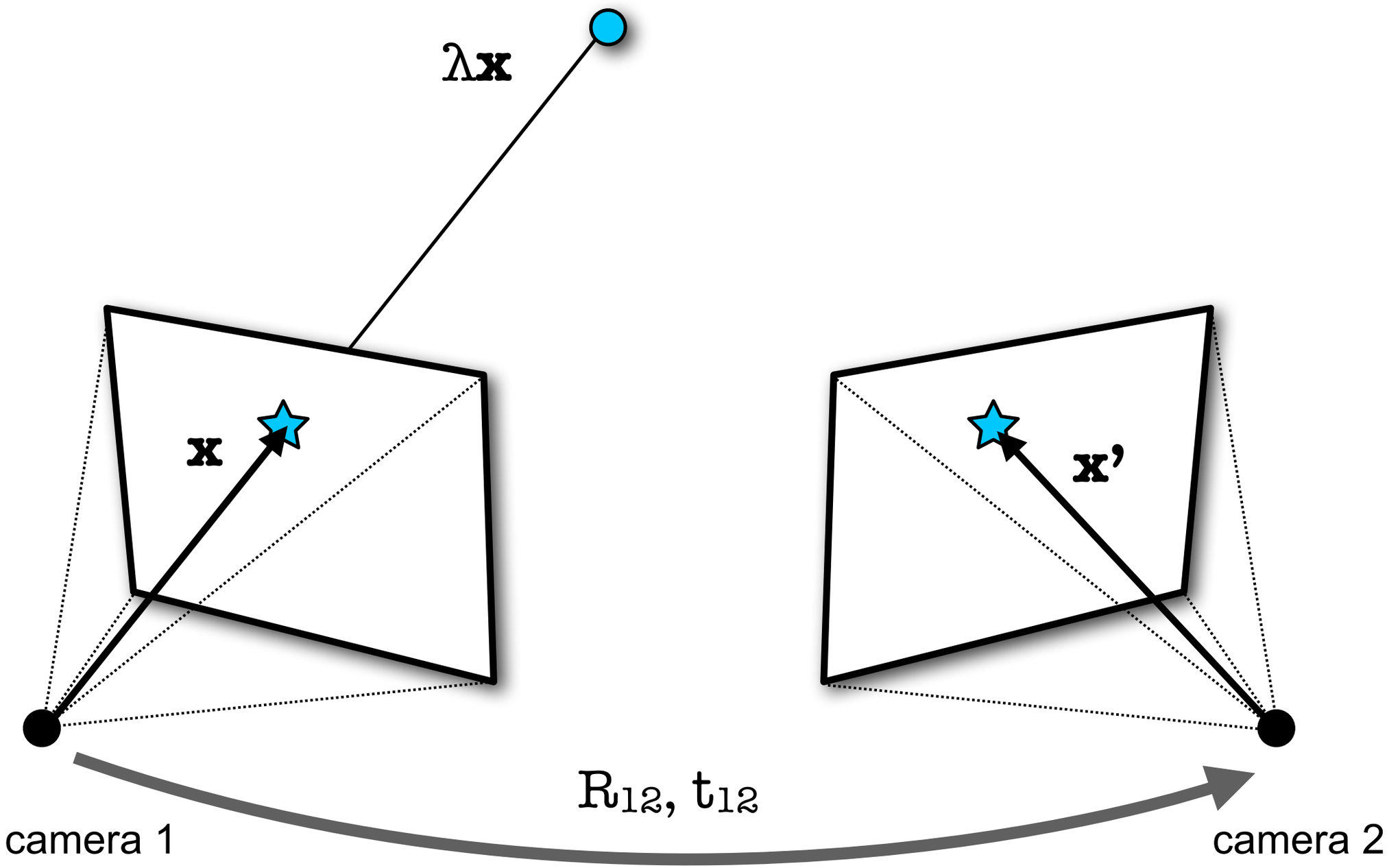
Epipolar Geometry



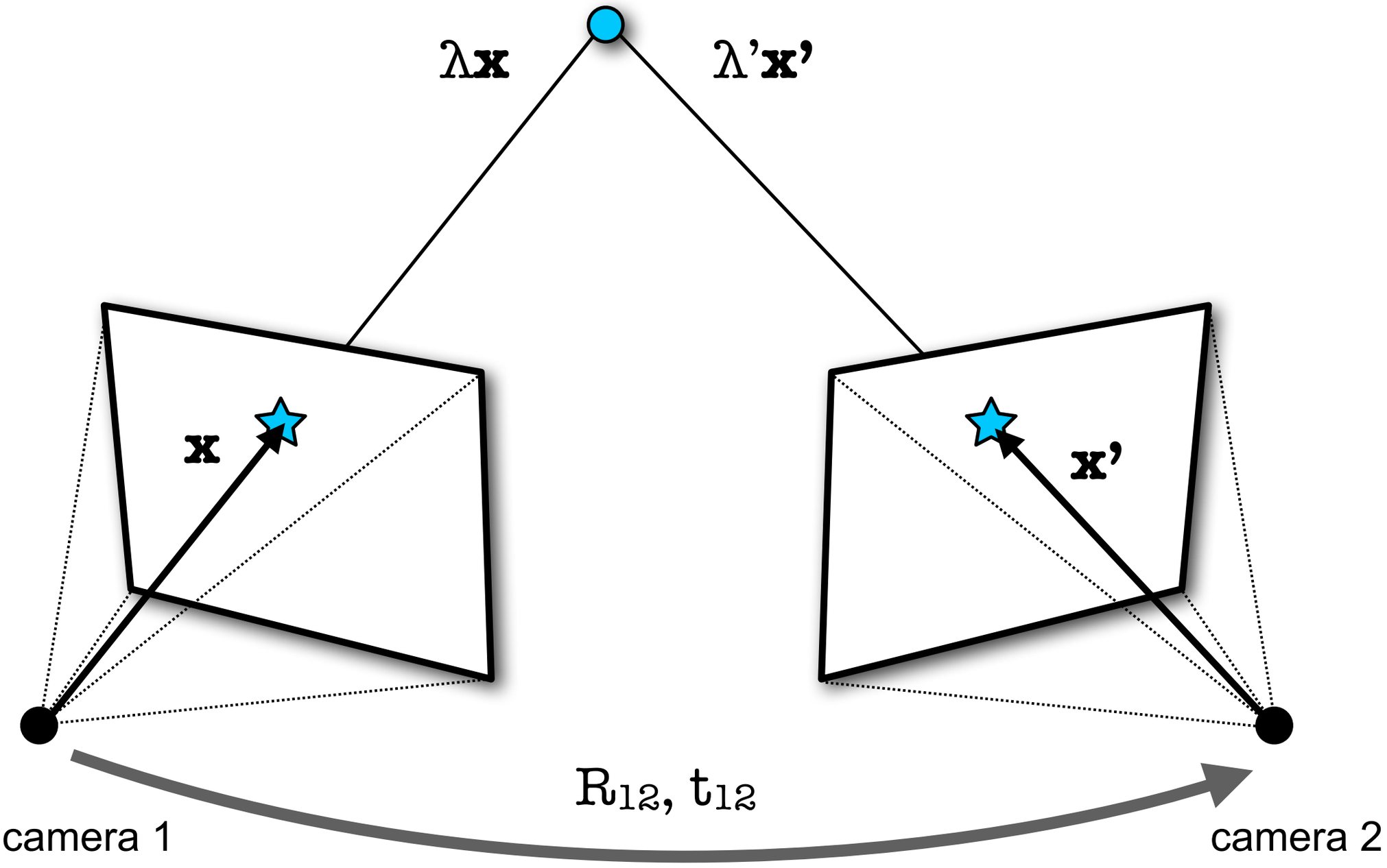
Epipolar Geometry



Epipolar Geometry

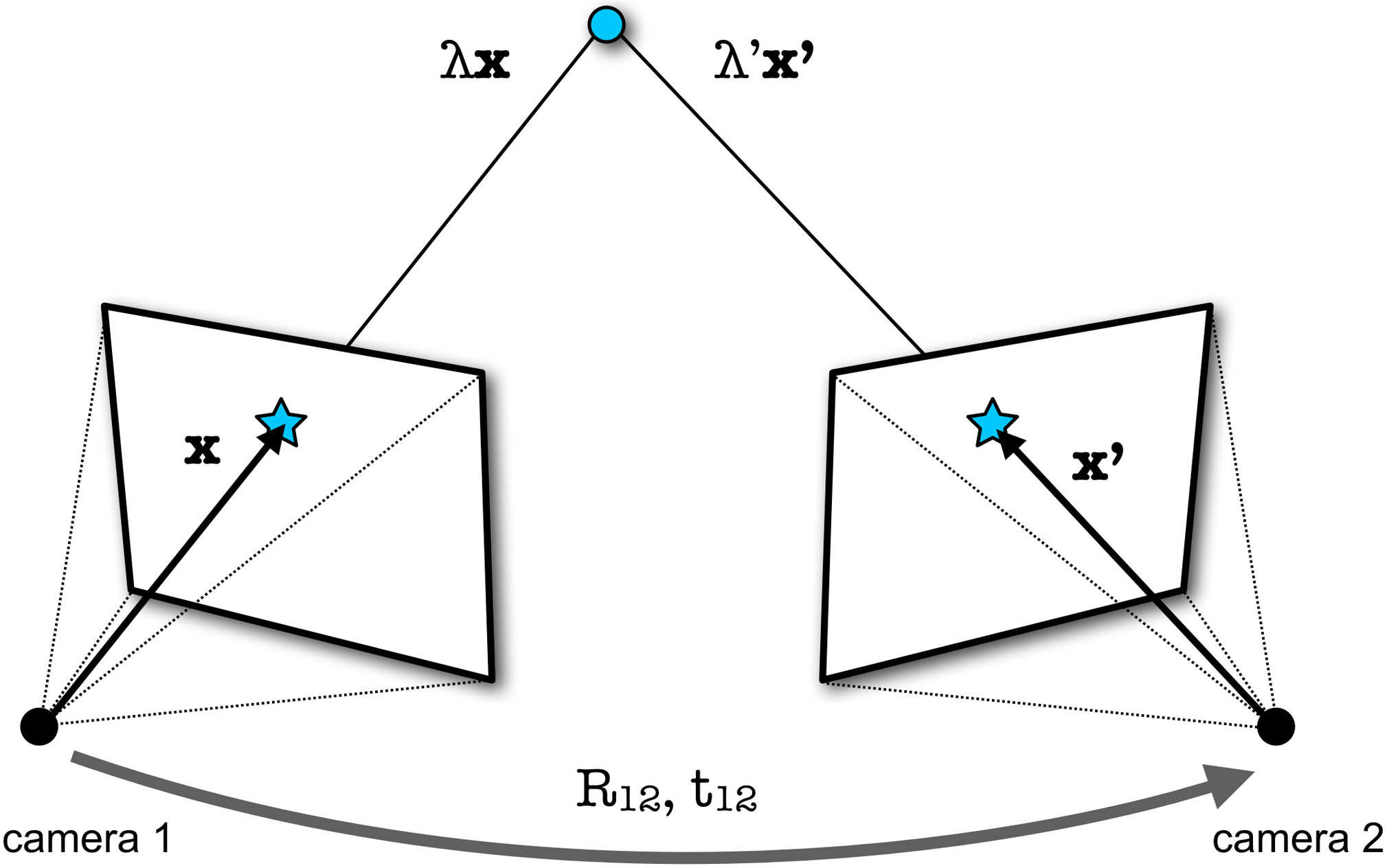


Epipolar Geometry



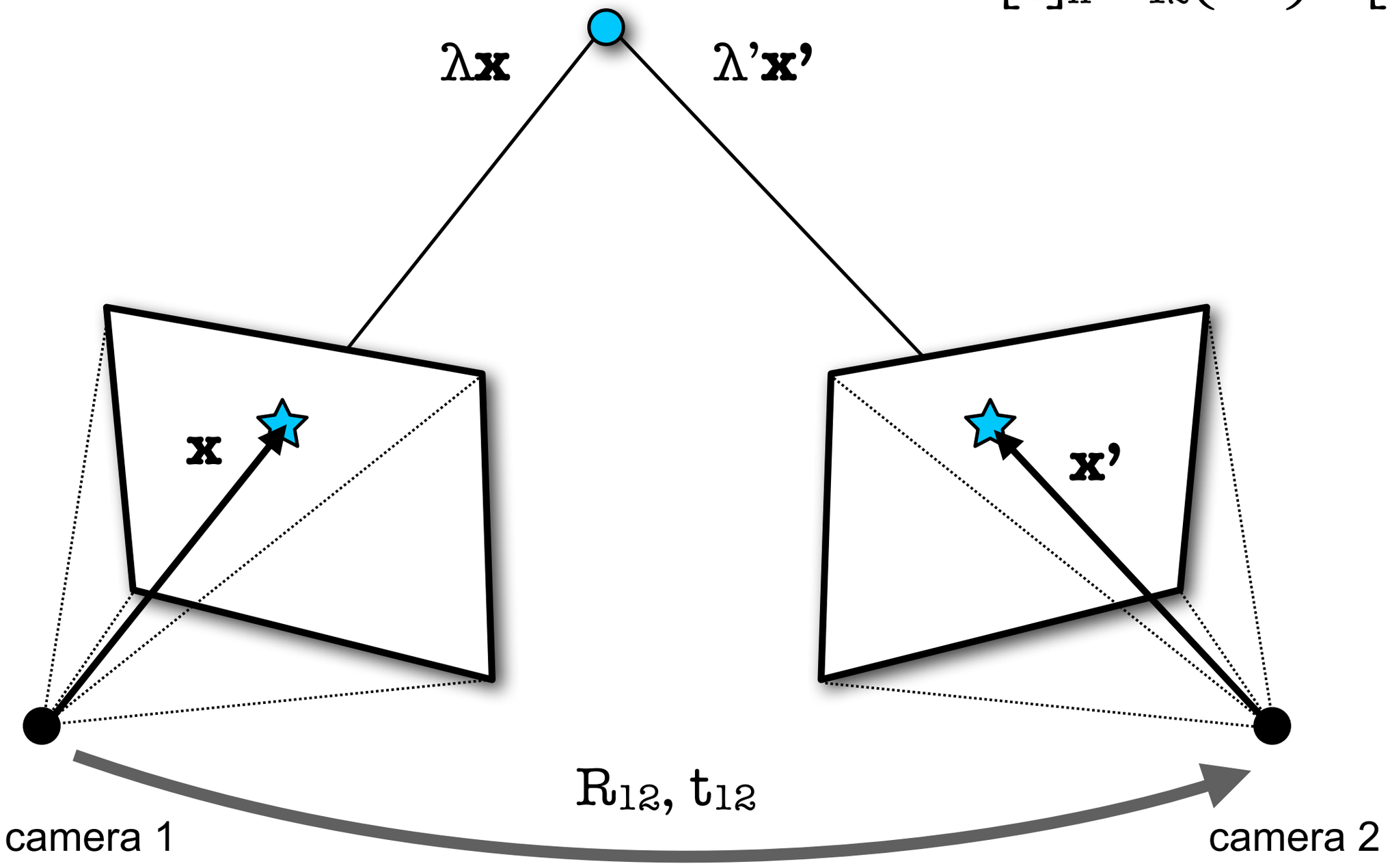
Epipolar Geometry

- $R_{12}(\lambda \mathbf{x}) + t_{12} = \lambda' \mathbf{x}'$



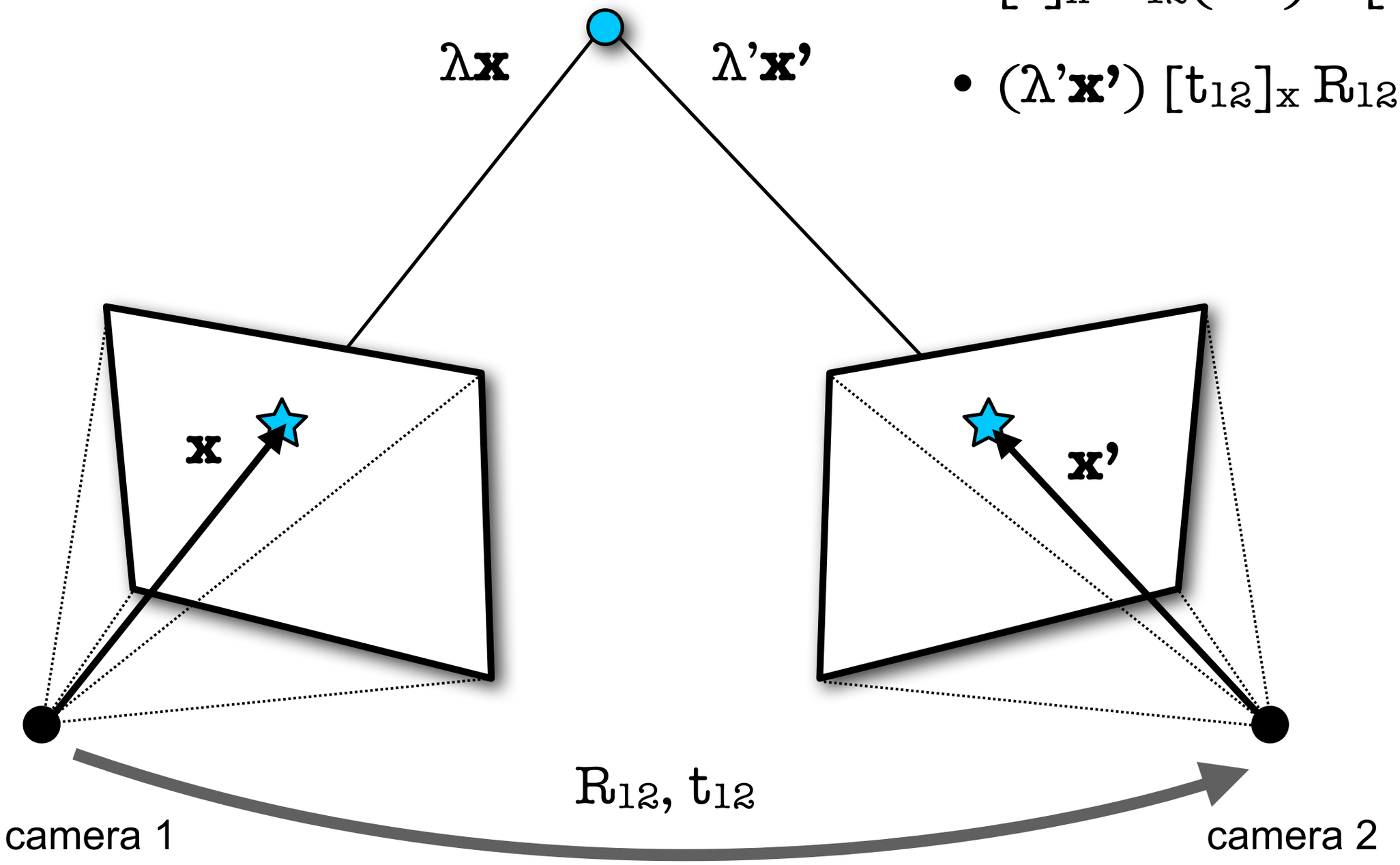
Epipolar Geometry

- $R_{12}(\lambda \mathbf{x}) + t_{12} = \lambda' \mathbf{x}'$
- $[t]_x R_{12}(\lambda \mathbf{x}) = [t_{12}]_x (\lambda' \mathbf{x}')$



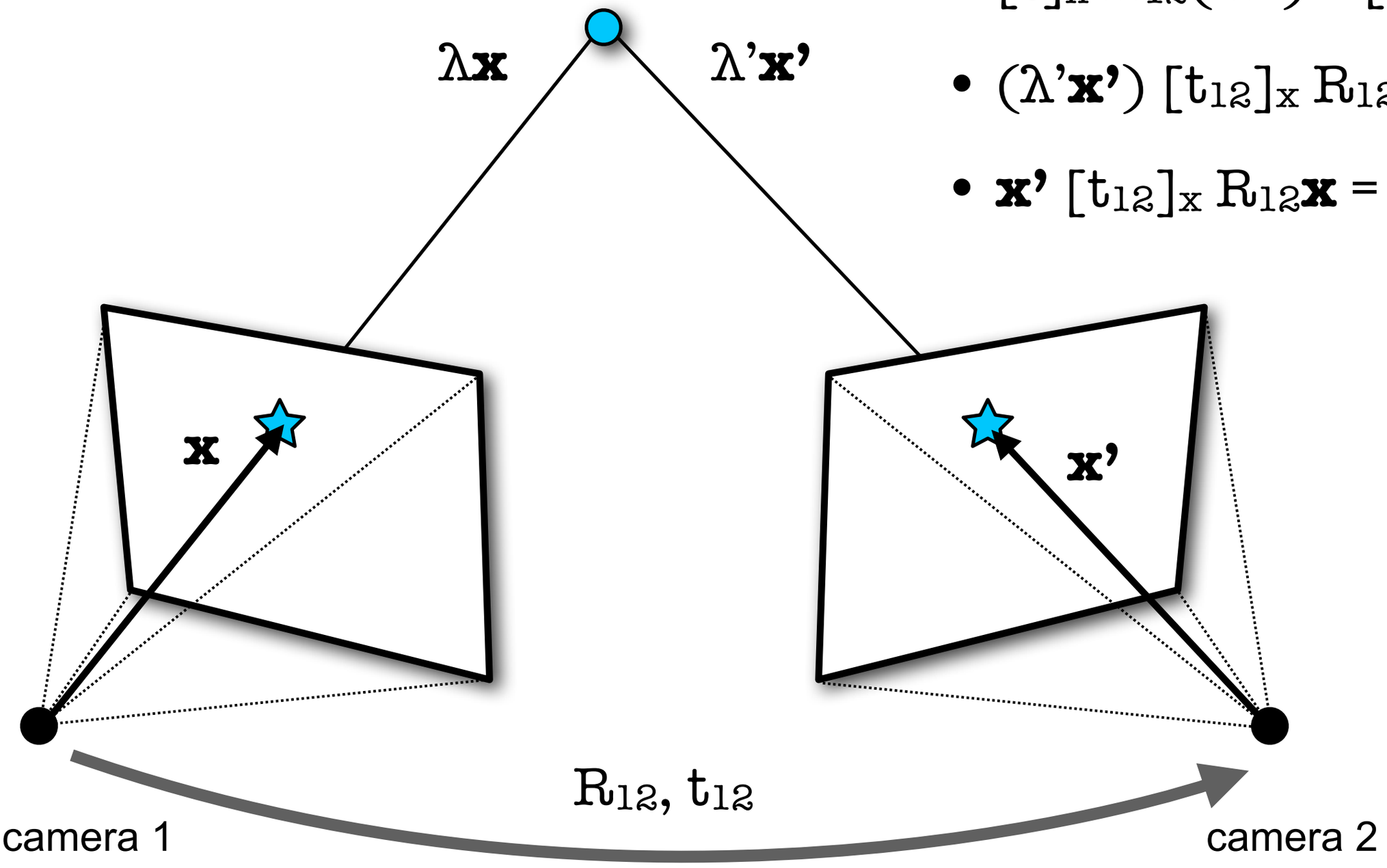
Epipolar Geometry

- $R_{12}(\lambda \mathbf{x}) + t_{12} = \lambda' \mathbf{x}'$
- $[t]_x R_{12}(\lambda \mathbf{x}) = [t_{12}]_x (\lambda' \mathbf{x}')$
- $(\lambda' \mathbf{x}') [t_{12}]_x R_{12}(\lambda \mathbf{x}) = 0$

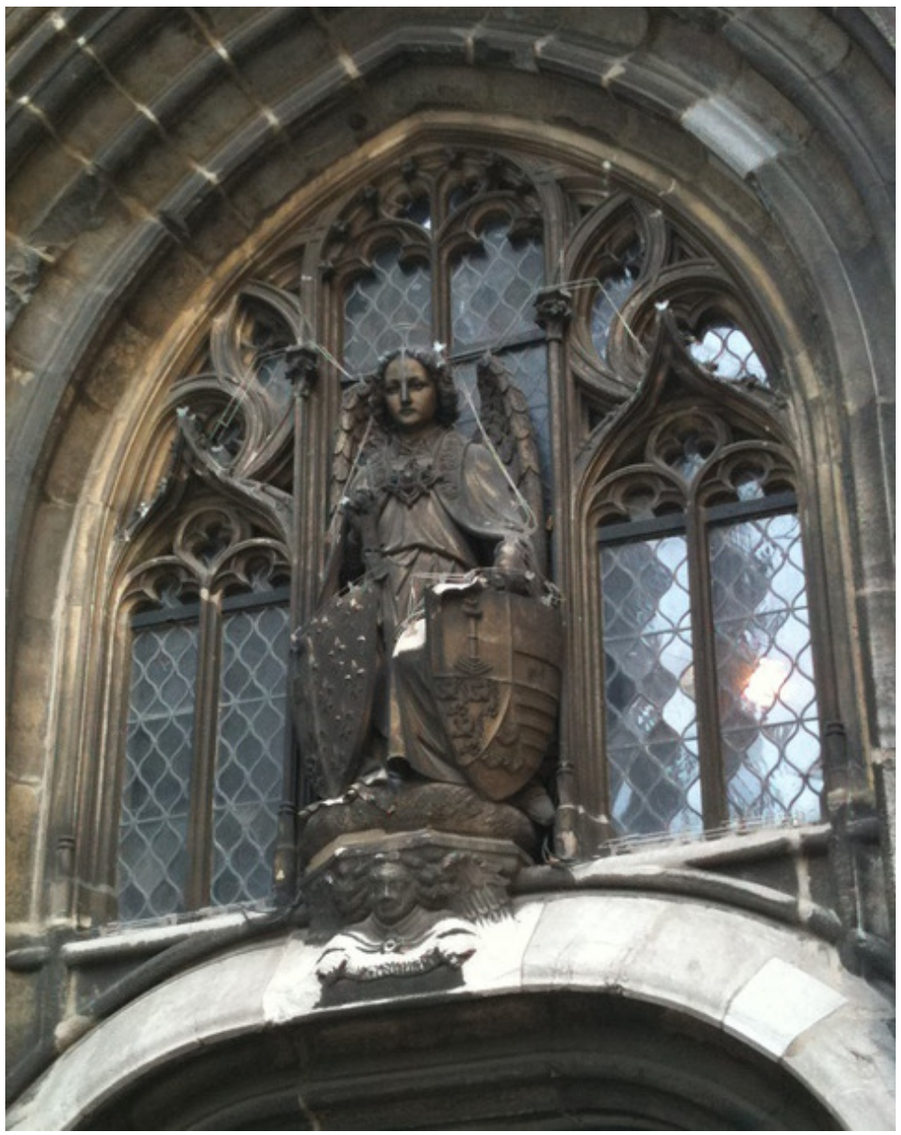


Epipolar Geometry

- $R_{12}(\lambda \mathbf{x}) + t_{12} = \lambda' \mathbf{x}'$
- $[t]_x R_{12}(\lambda \mathbf{x}) = [t_{12}]_x (\lambda' \mathbf{x}')$
- $(\lambda' \mathbf{x}') [t_{12}]_x R_{12}(\lambda \mathbf{x}) = 0$
- $\mathbf{x}' [t_{12}]_x R_{12} \mathbf{x} = 0$



Scale Invariant Feature Transform (SIFT)



[Lowe, IJCV'04]

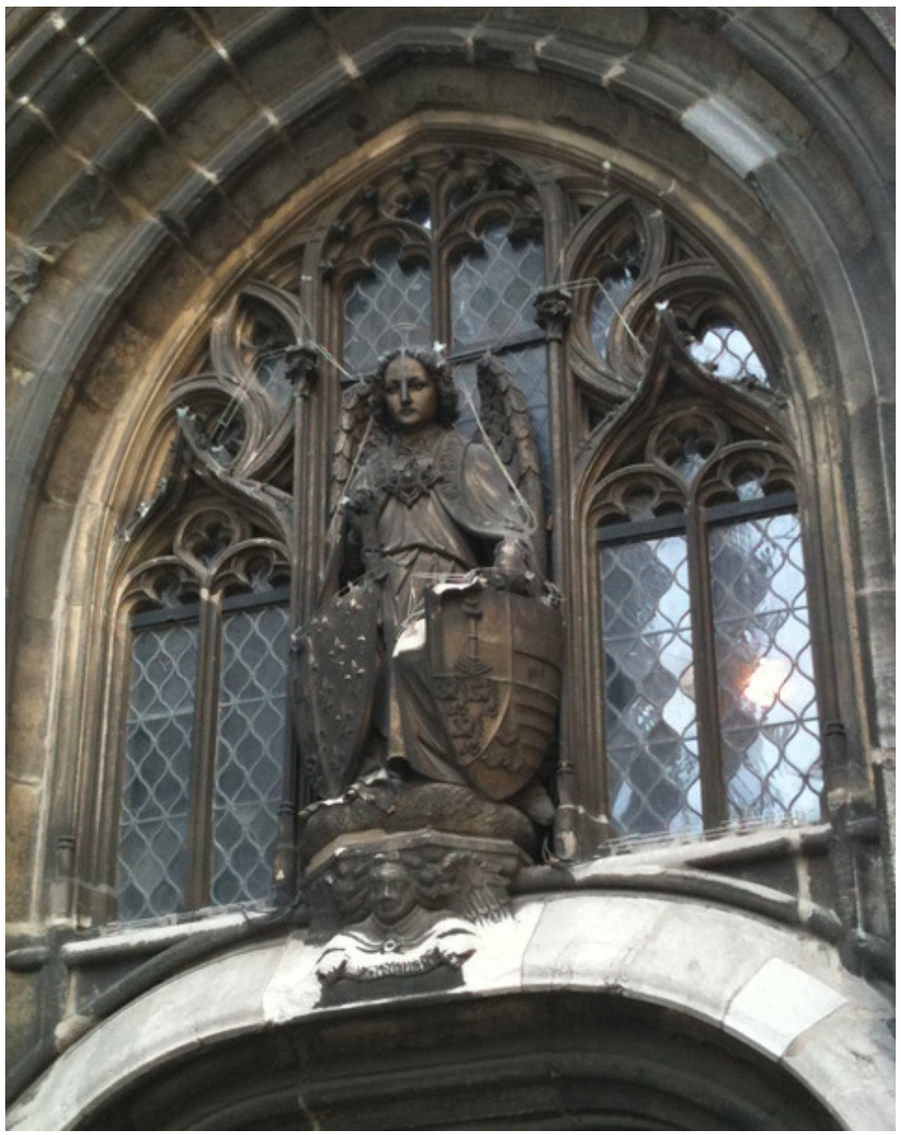
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales

[Lowe, IJCV'04]

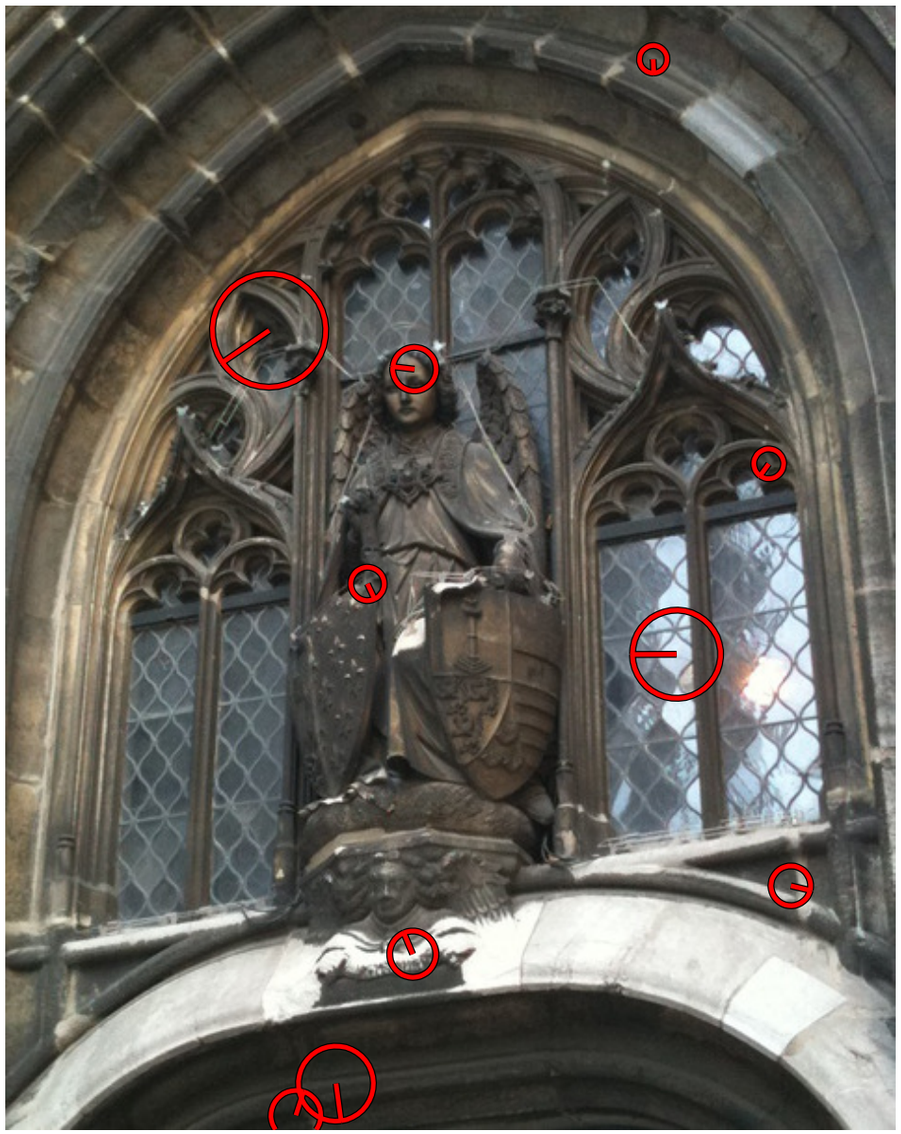
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale

[Lowe, IJCV'04]

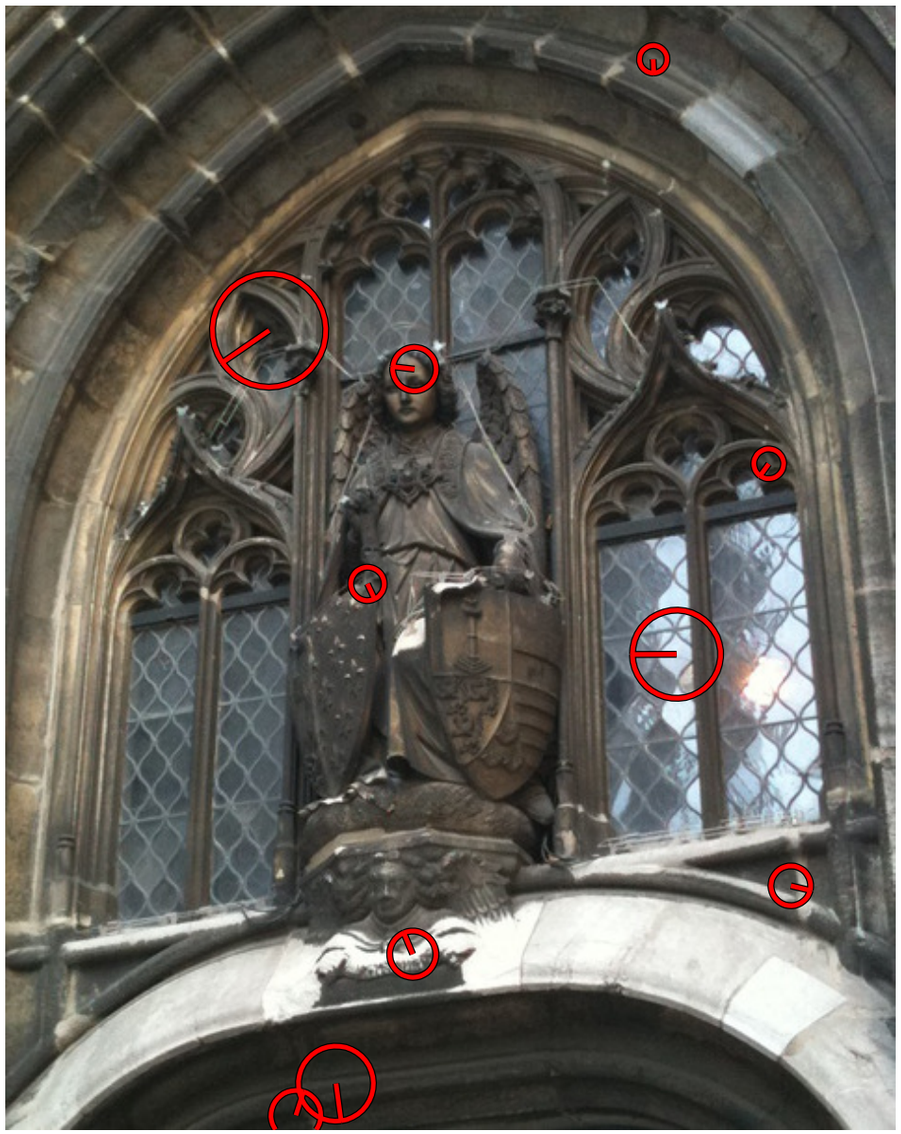
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale

[Lowe, IJCV'04]

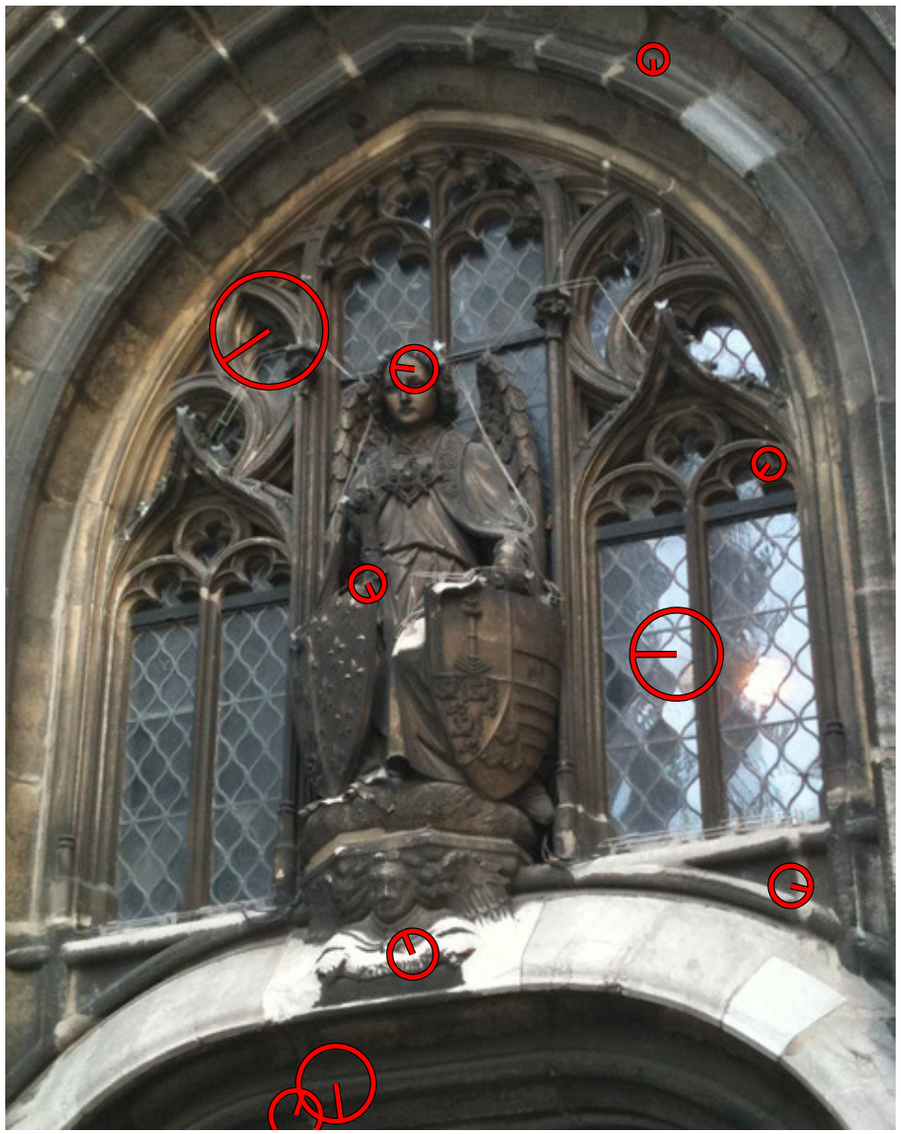
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint

[Lowe, IJCV'04]

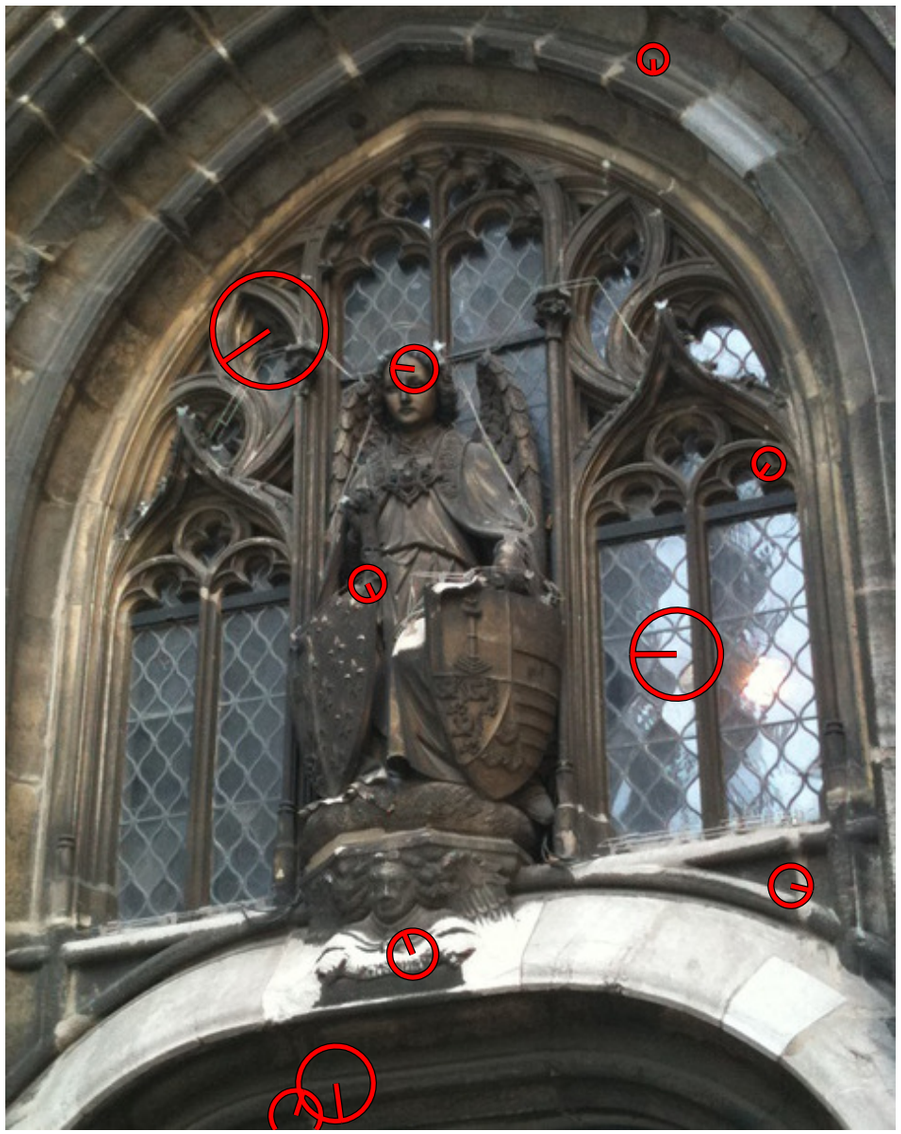
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms

[Lowe, IJCV'04]

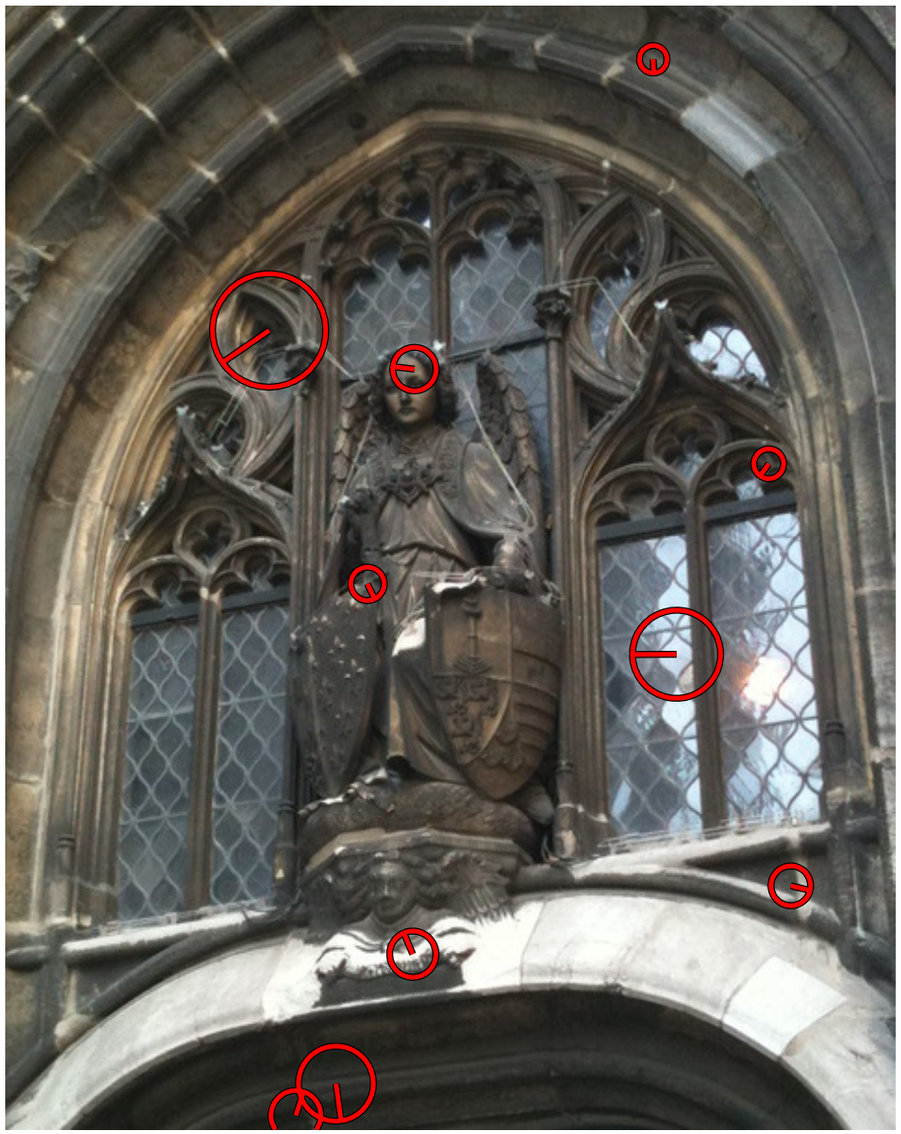
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector

[Lowe, IJCV'04]

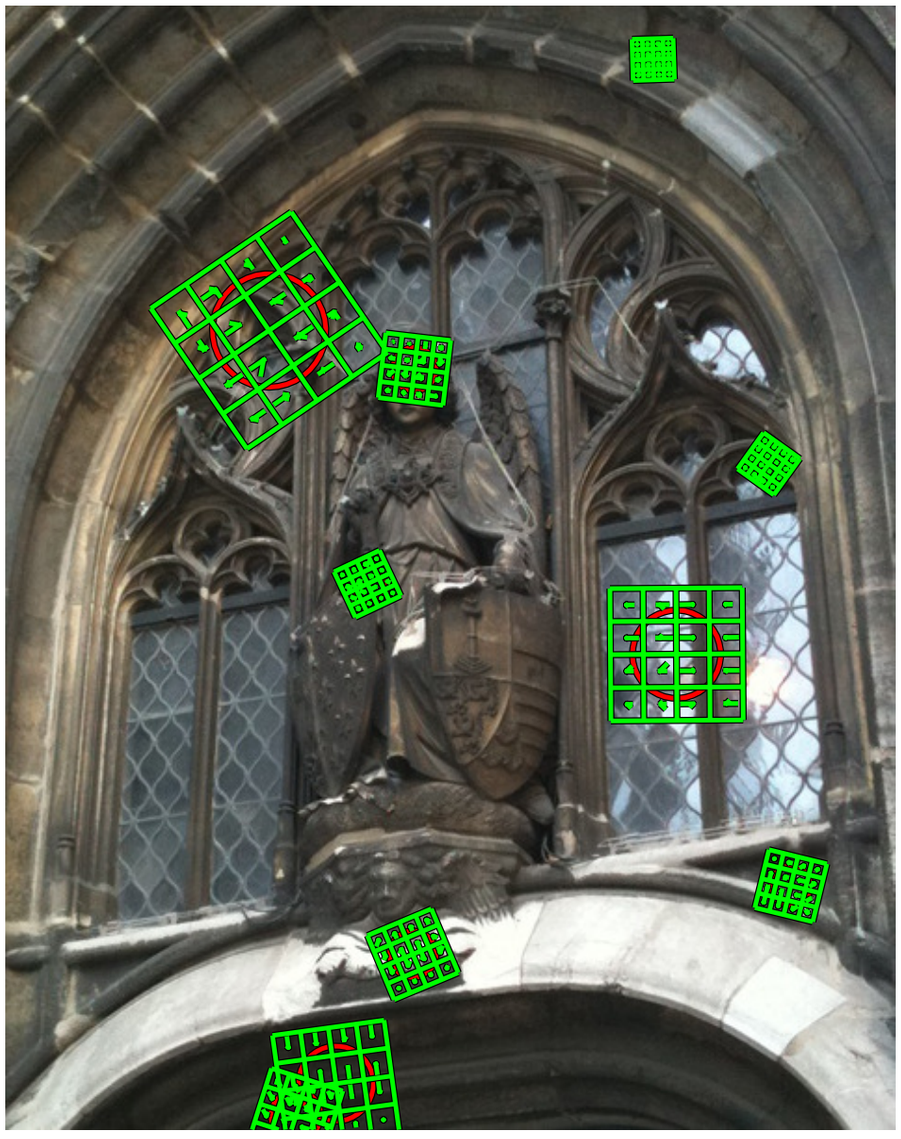
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector
  - Description of image patch around keypoint

[Lowe, IJCV'04]

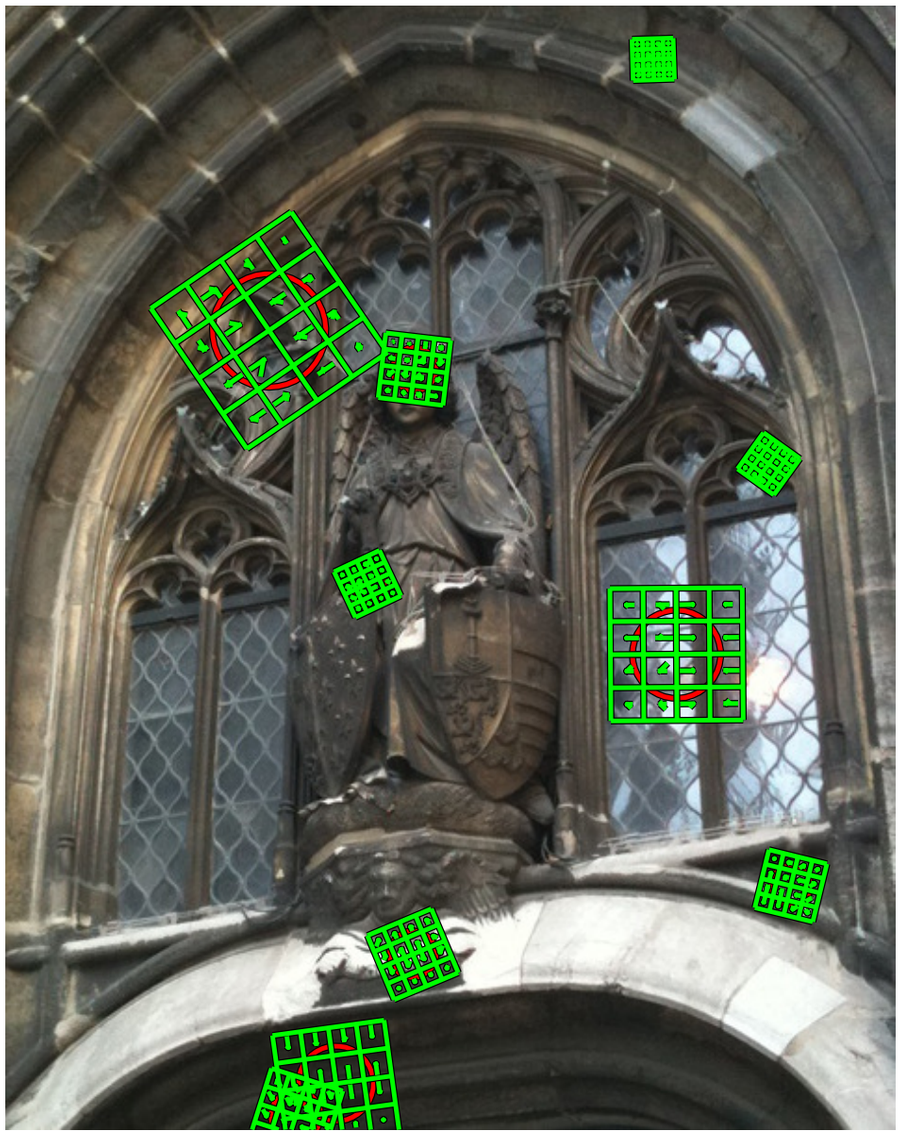
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector
- Description of image patch around keypoint

[Lowe, IJCV'04]

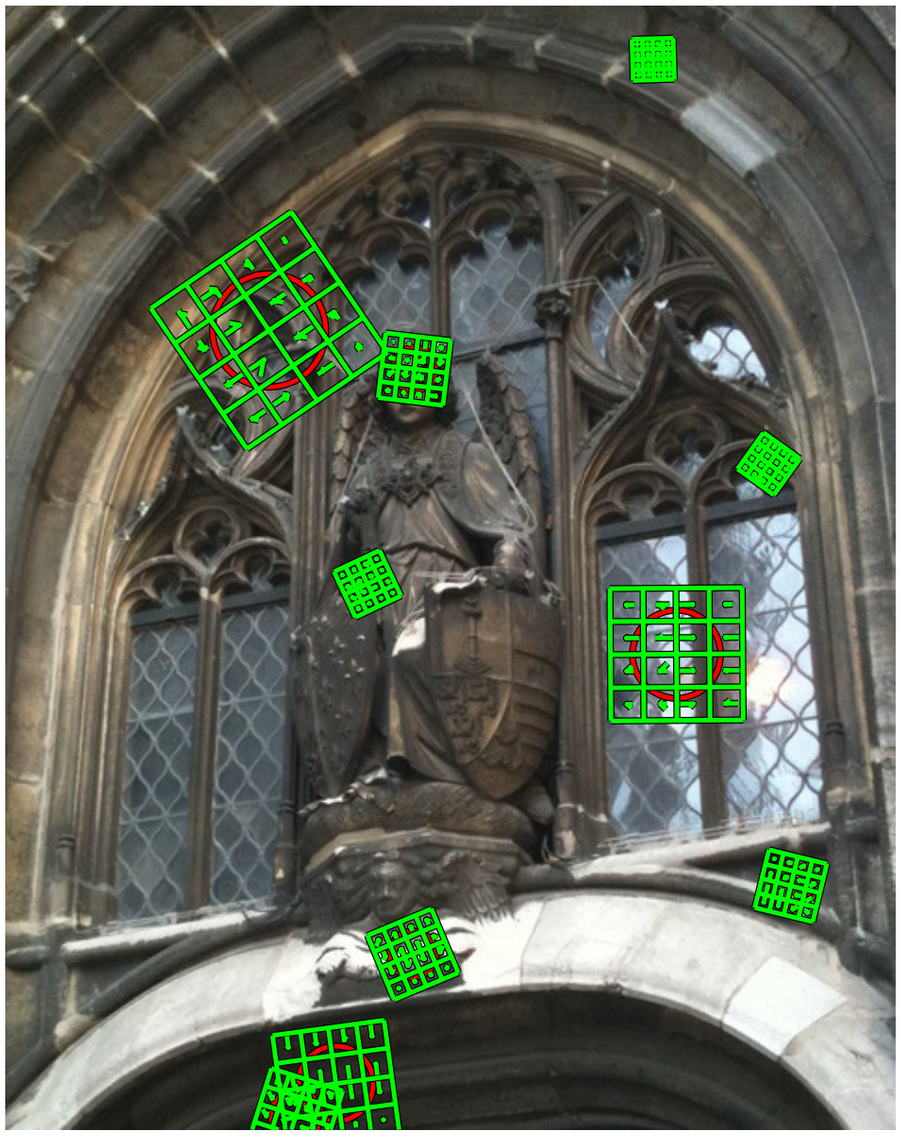
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector
  - Description of image patch around keypoint
- **Feature**: Keypoint + descriptor

[Lowe, IJCV'04]

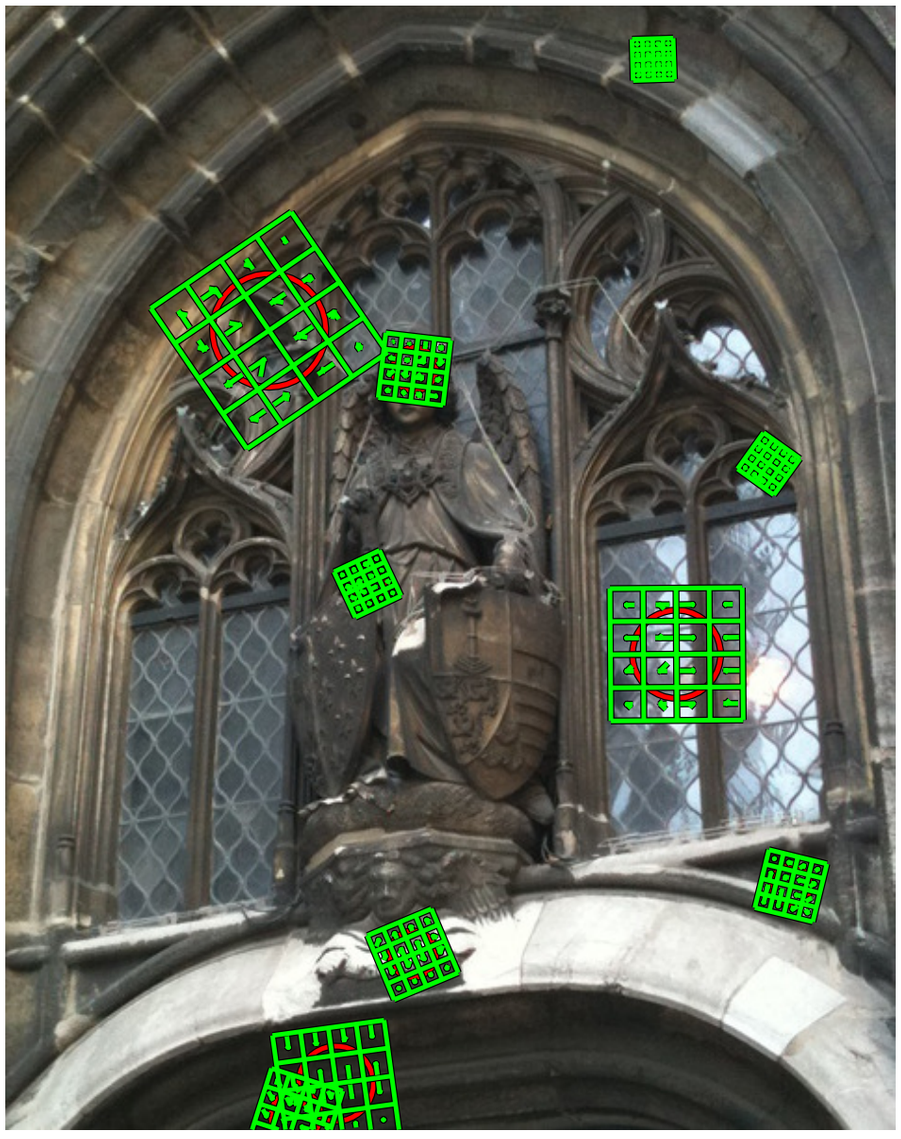
Scale Invariant Feature Transform (SIFT)



- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector
  - Description of image patch around keypoint
- **Feature**: Keypoint + descriptor
- Slow to compute, use GPU implementations

[Lowe, IJCV'04]

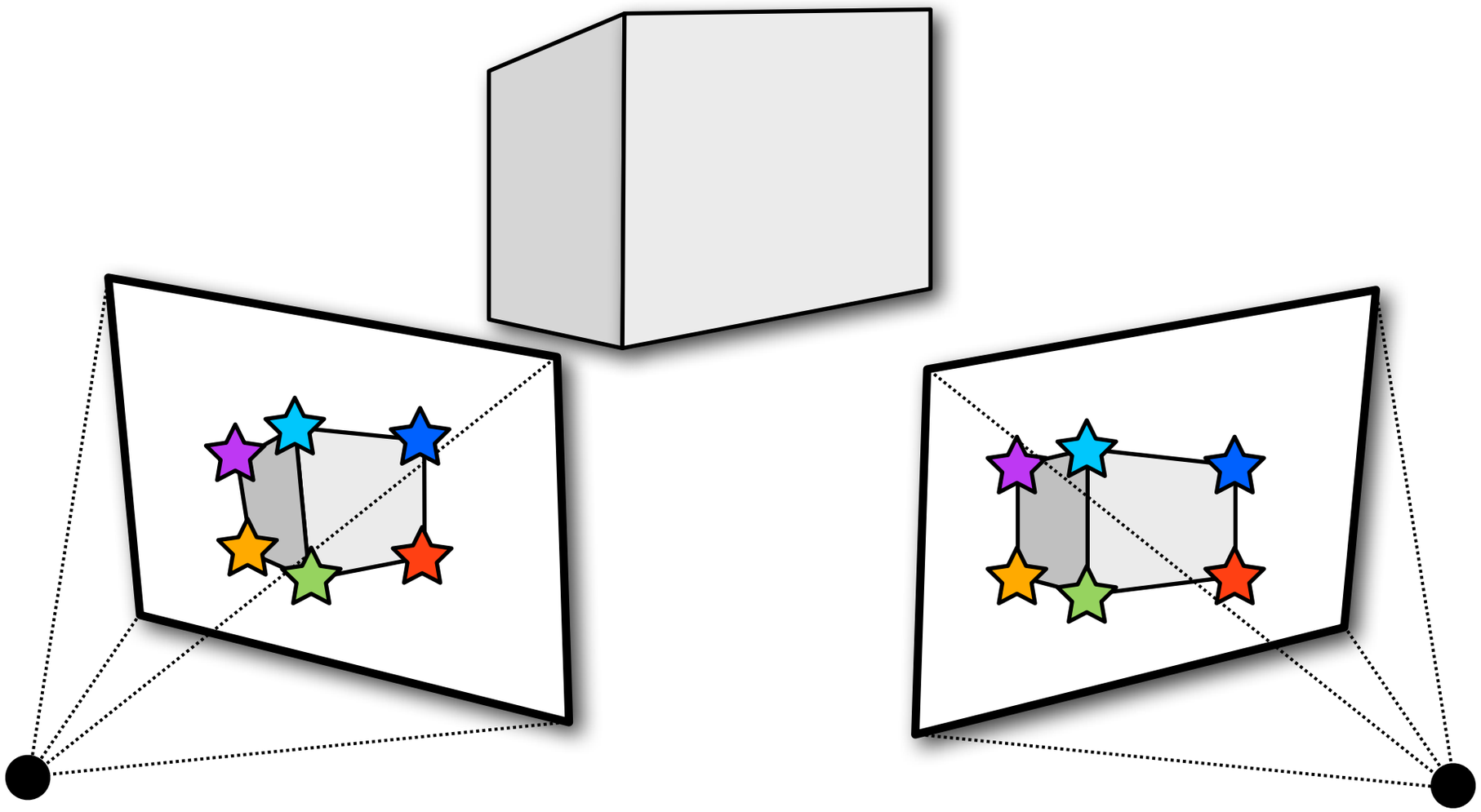
Scale Invariant Feature Transform (SIFT)



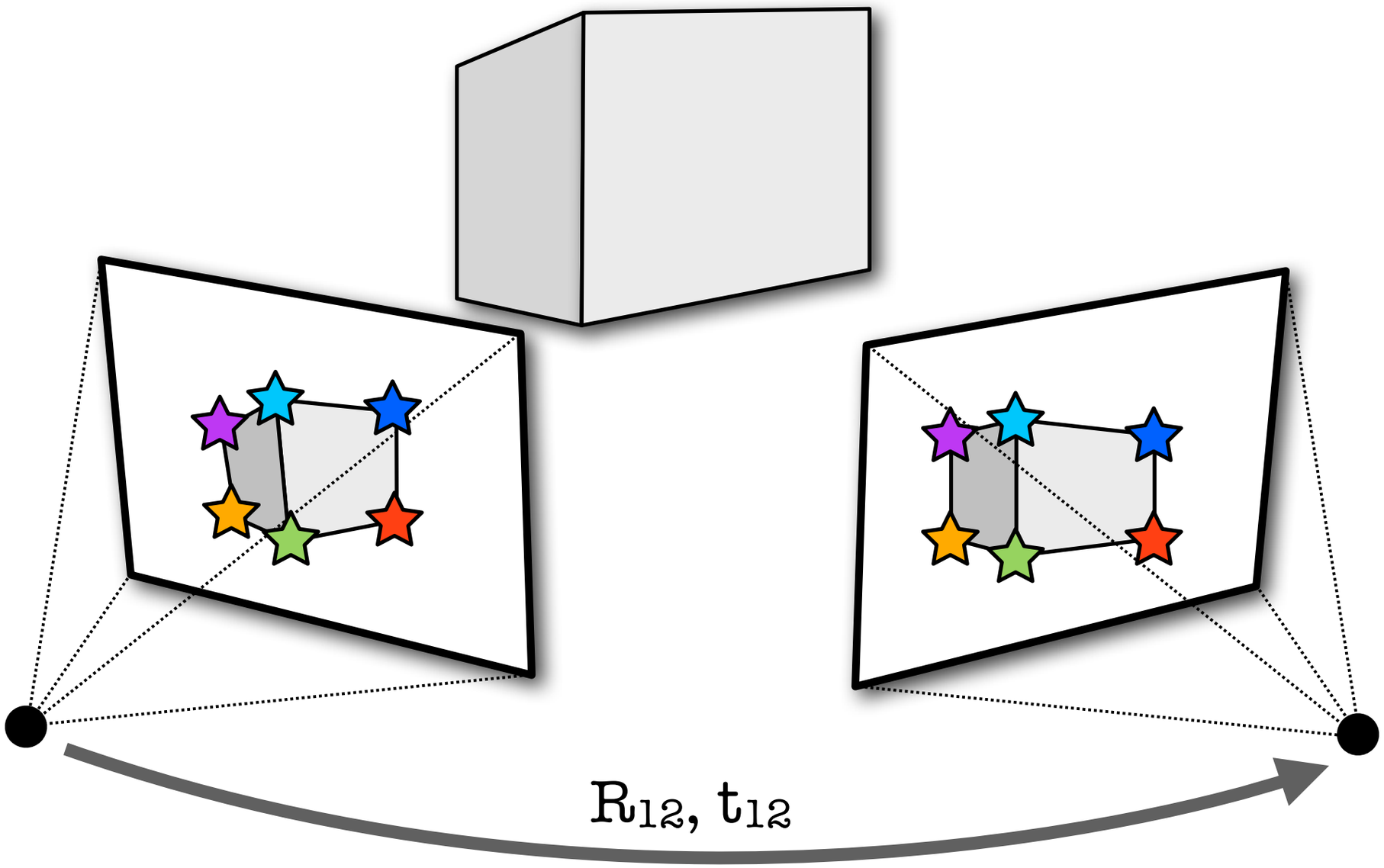
- Detect **keypoints** at multiple scales
- **Keypoint**: Position, orientation & scale
- Compute **descriptor** for each keypoint
  - Concatenate gradient histograms
  - 128 dimensional vector
  - Description of image patch around keypoint
- **Feature**: Keypoint + descriptor
- Slow to compute, use GPU implementations
- … but works really, really well!

[Lowe, IJCV'04]

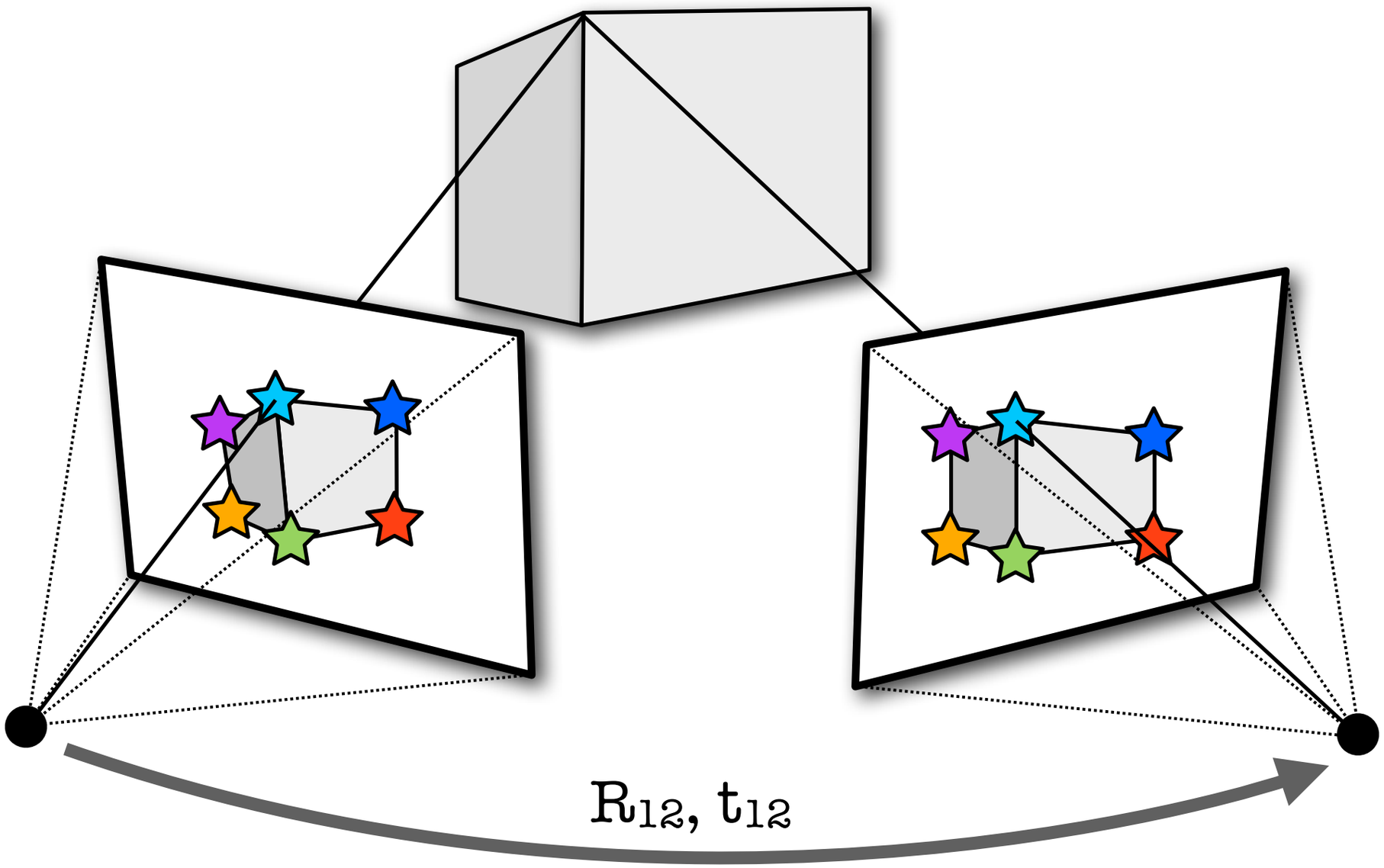
Structure-from-Motion (SfM)



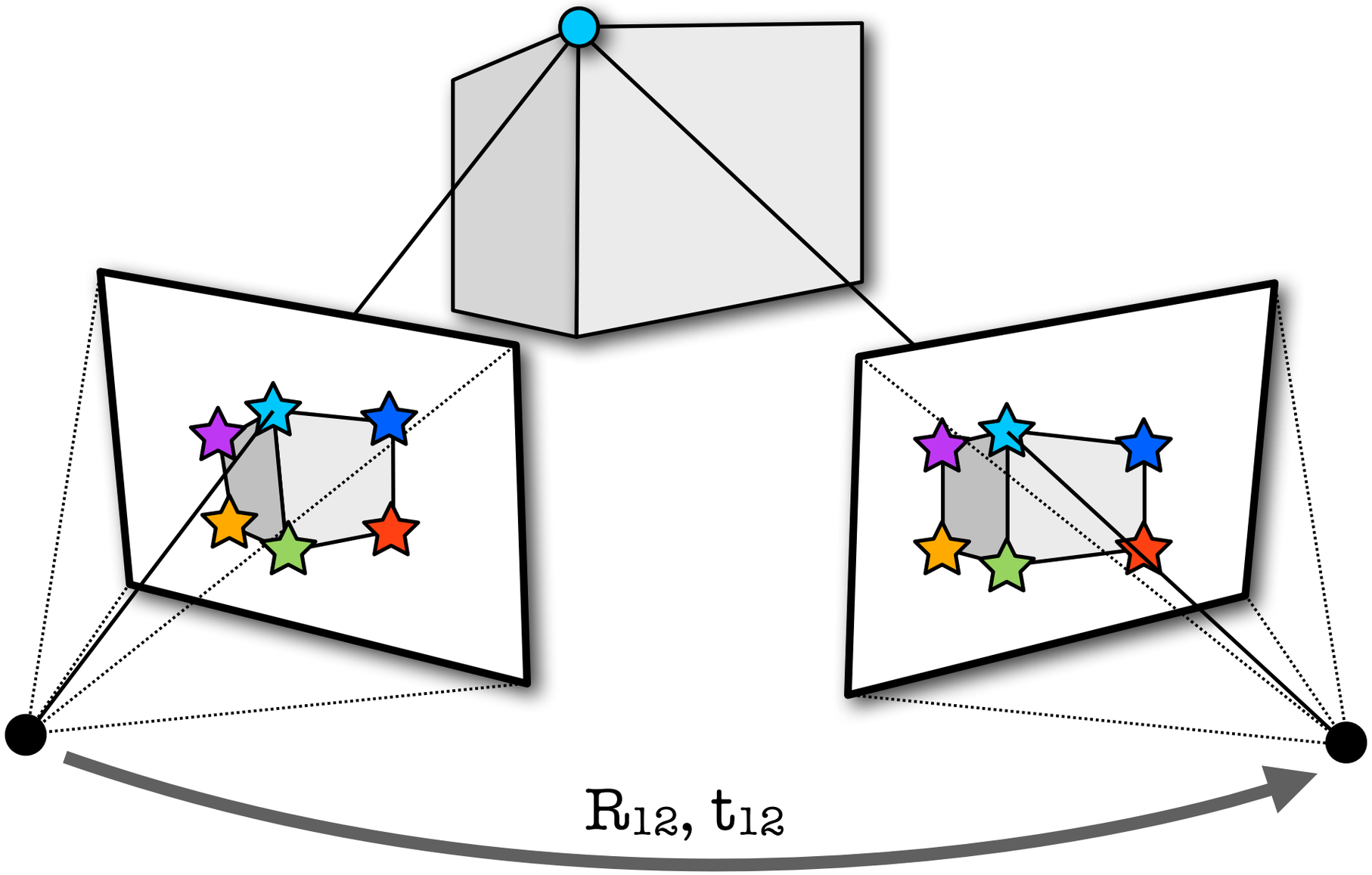
Structure-from-Motion (SfM)



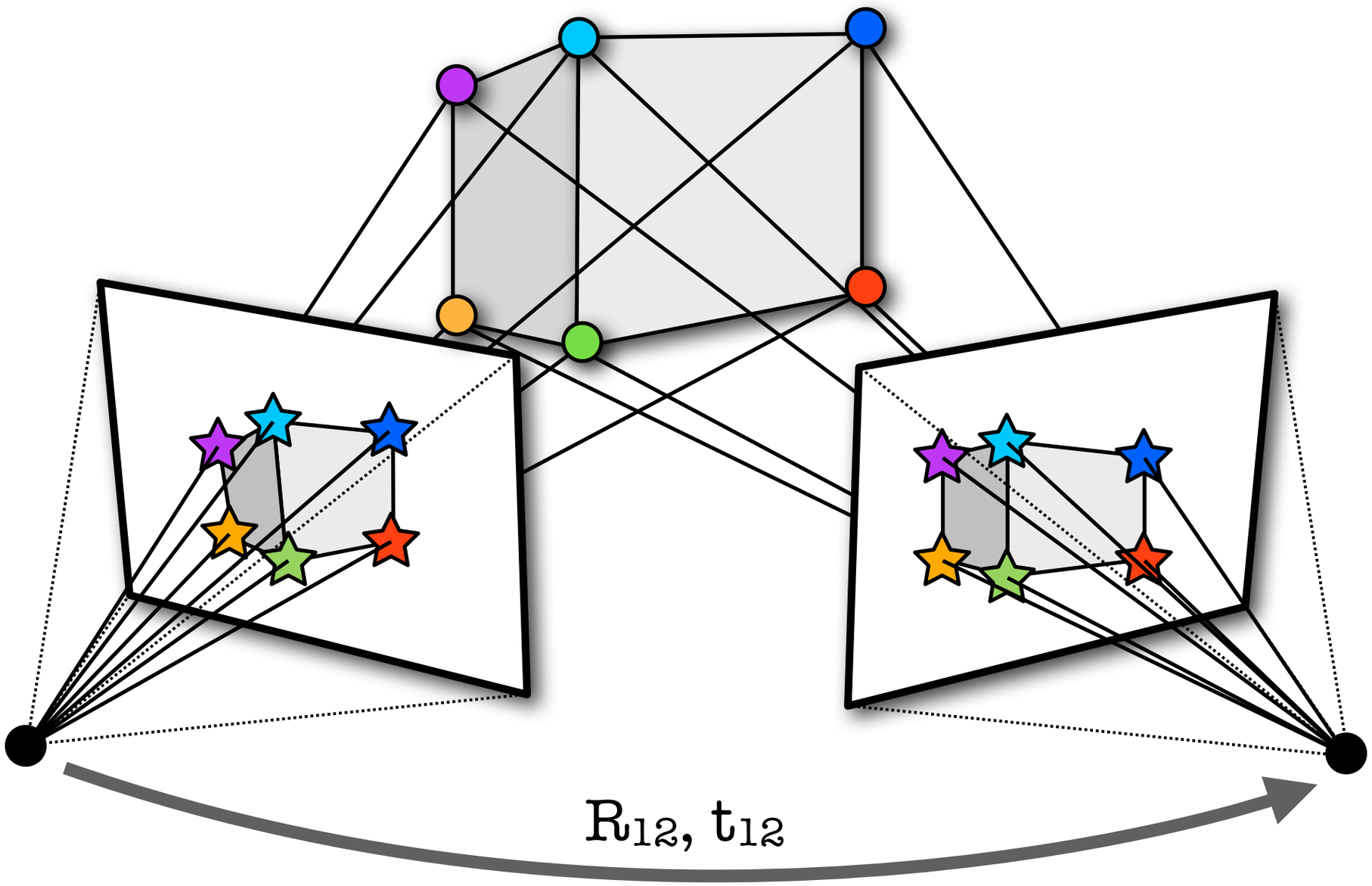
Structure-from-Motion (SfM)



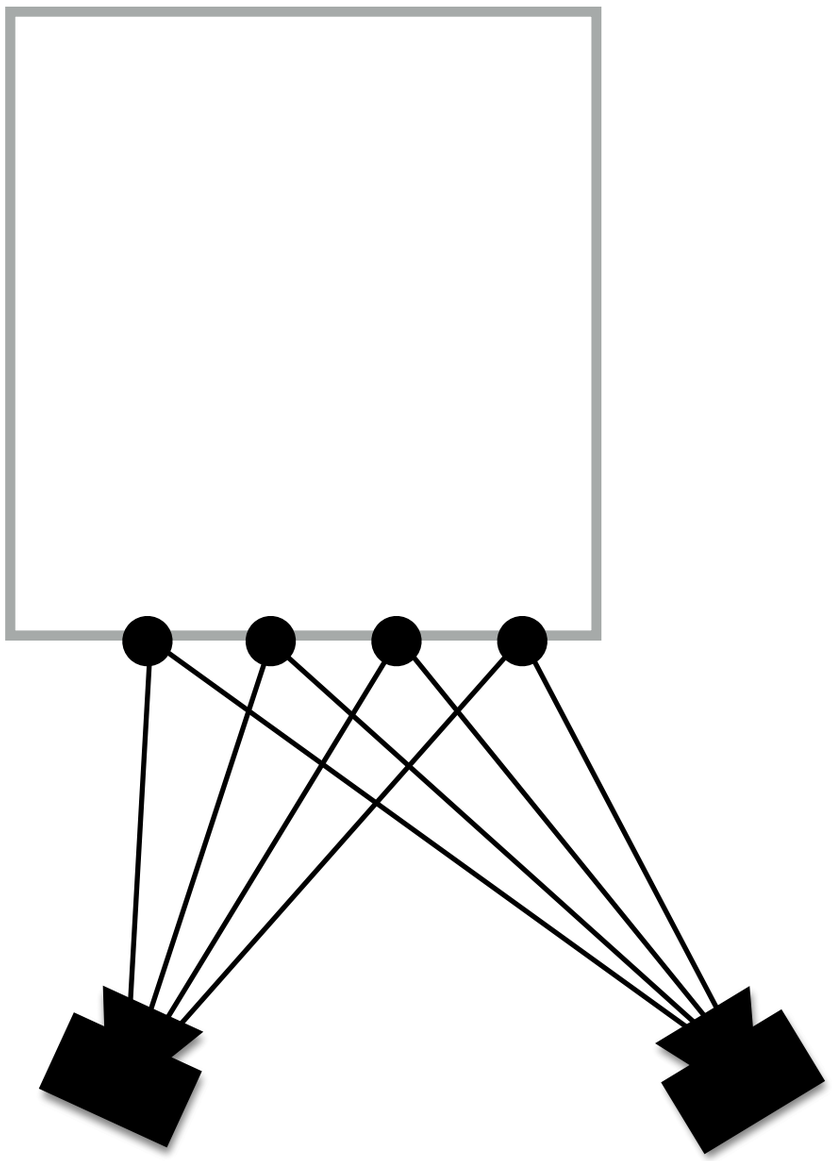
Structure-from-Motion (SfM)



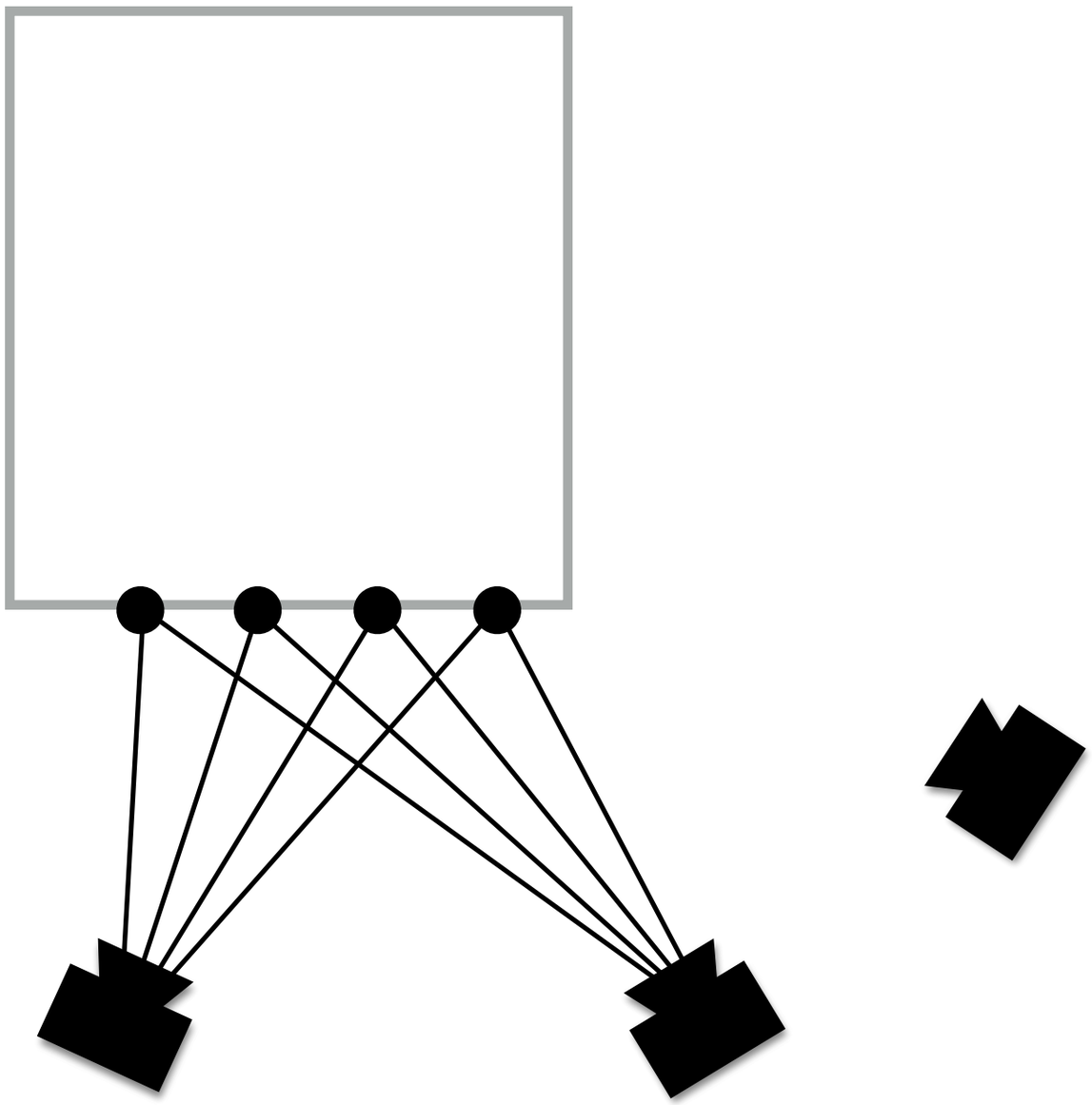
Structure-from-Motion (SfM)



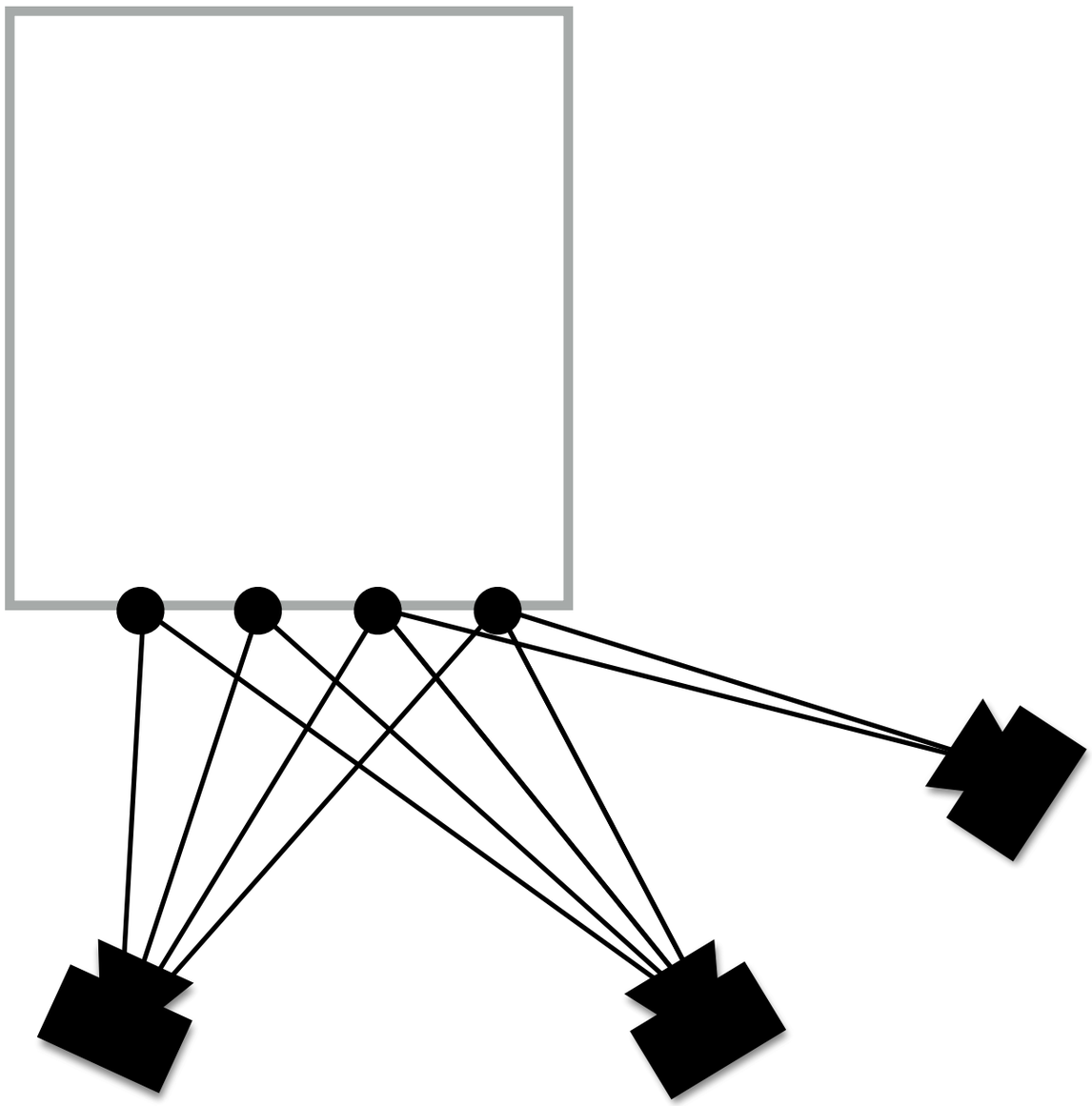
Incremental SfM



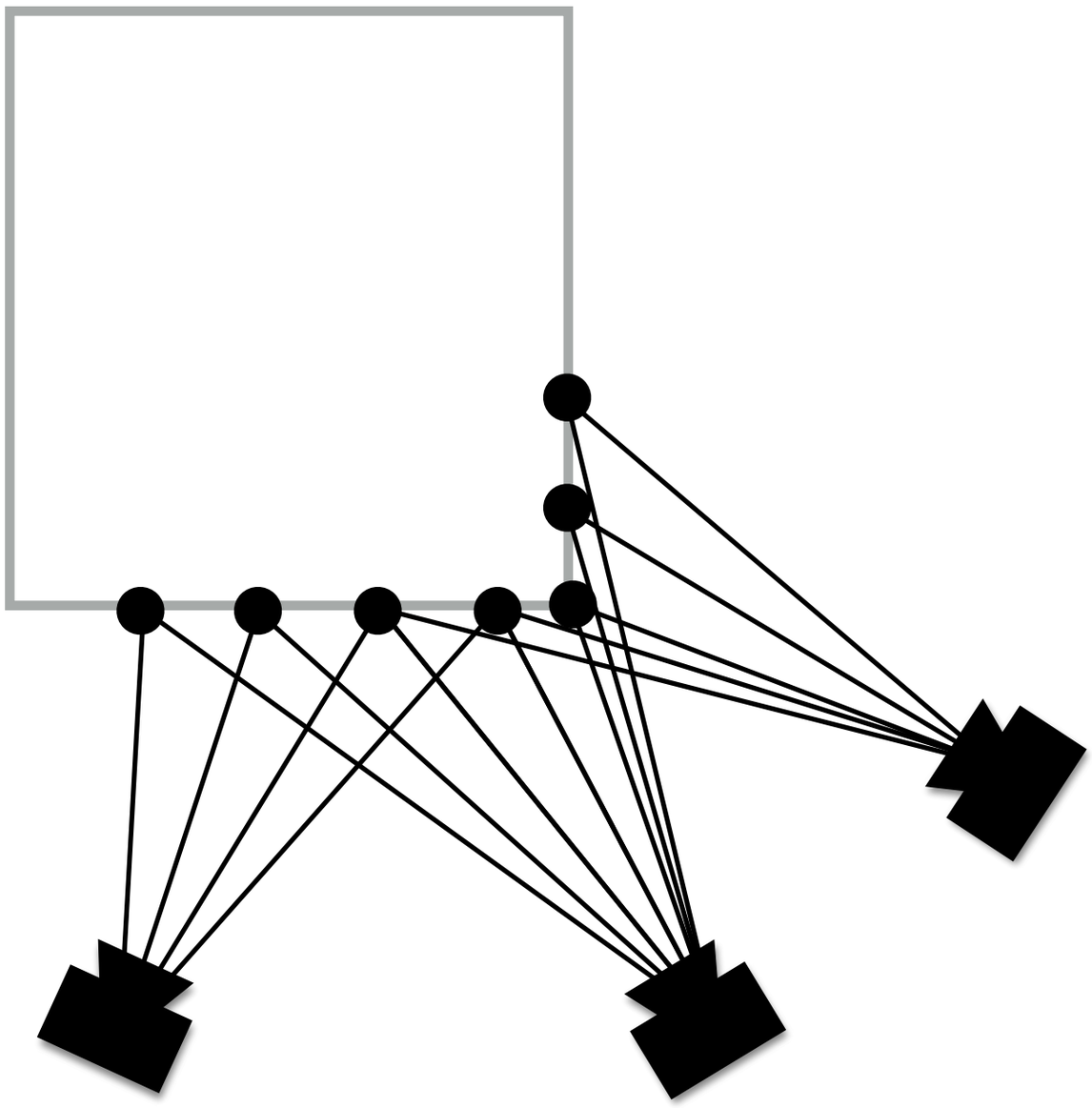
Incremental SfM



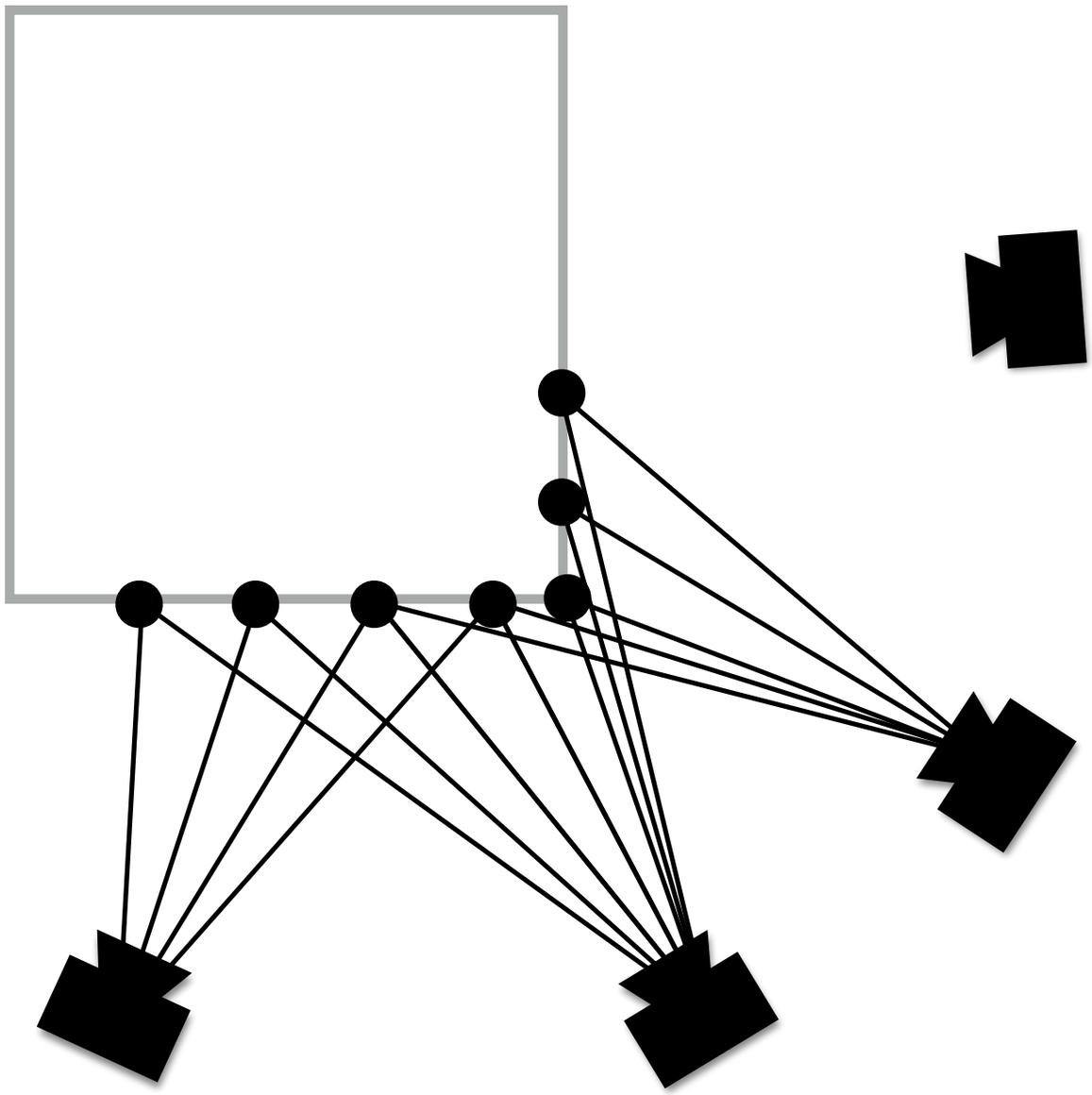
Incremental SfM



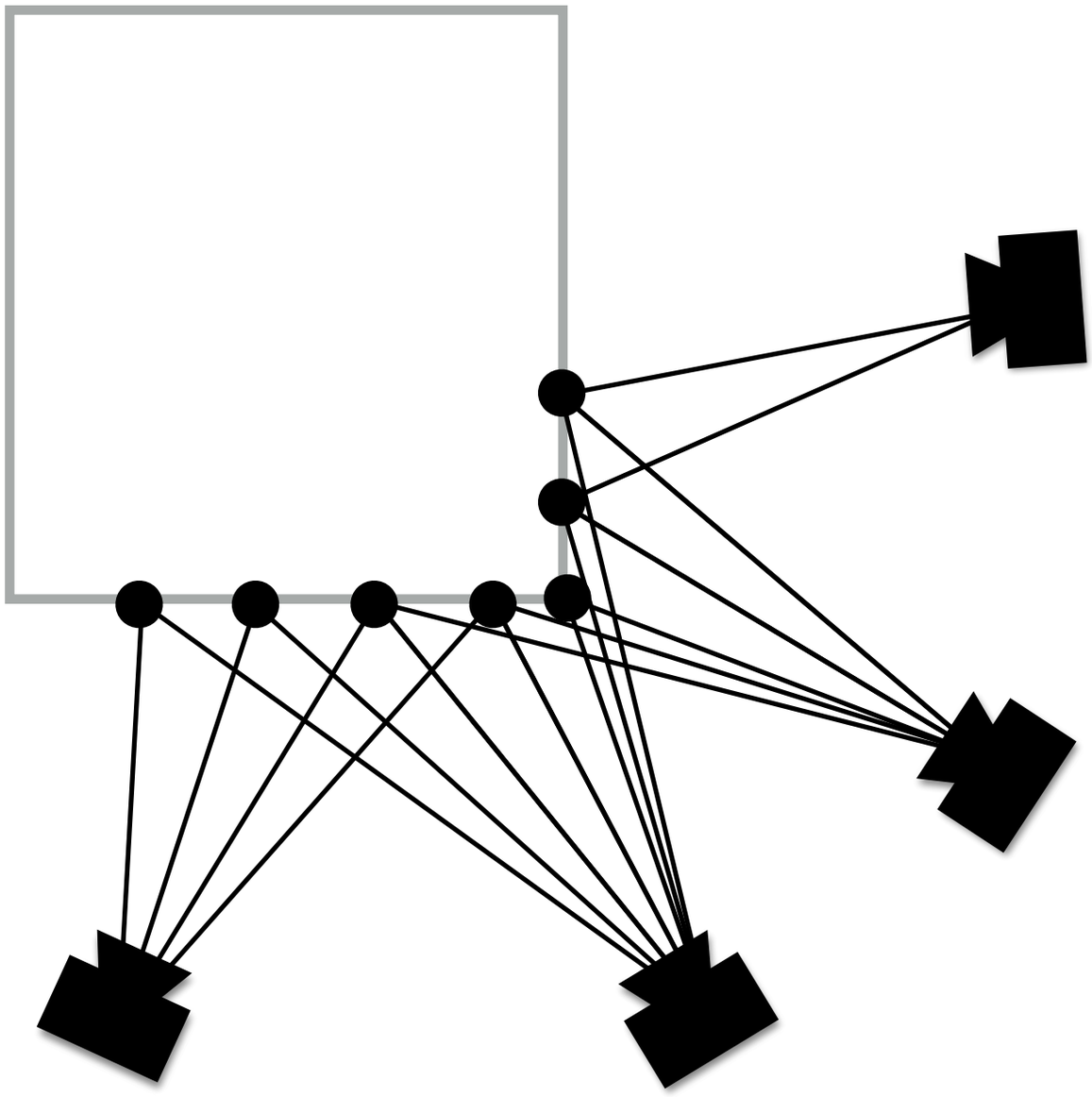
Incremental SfM



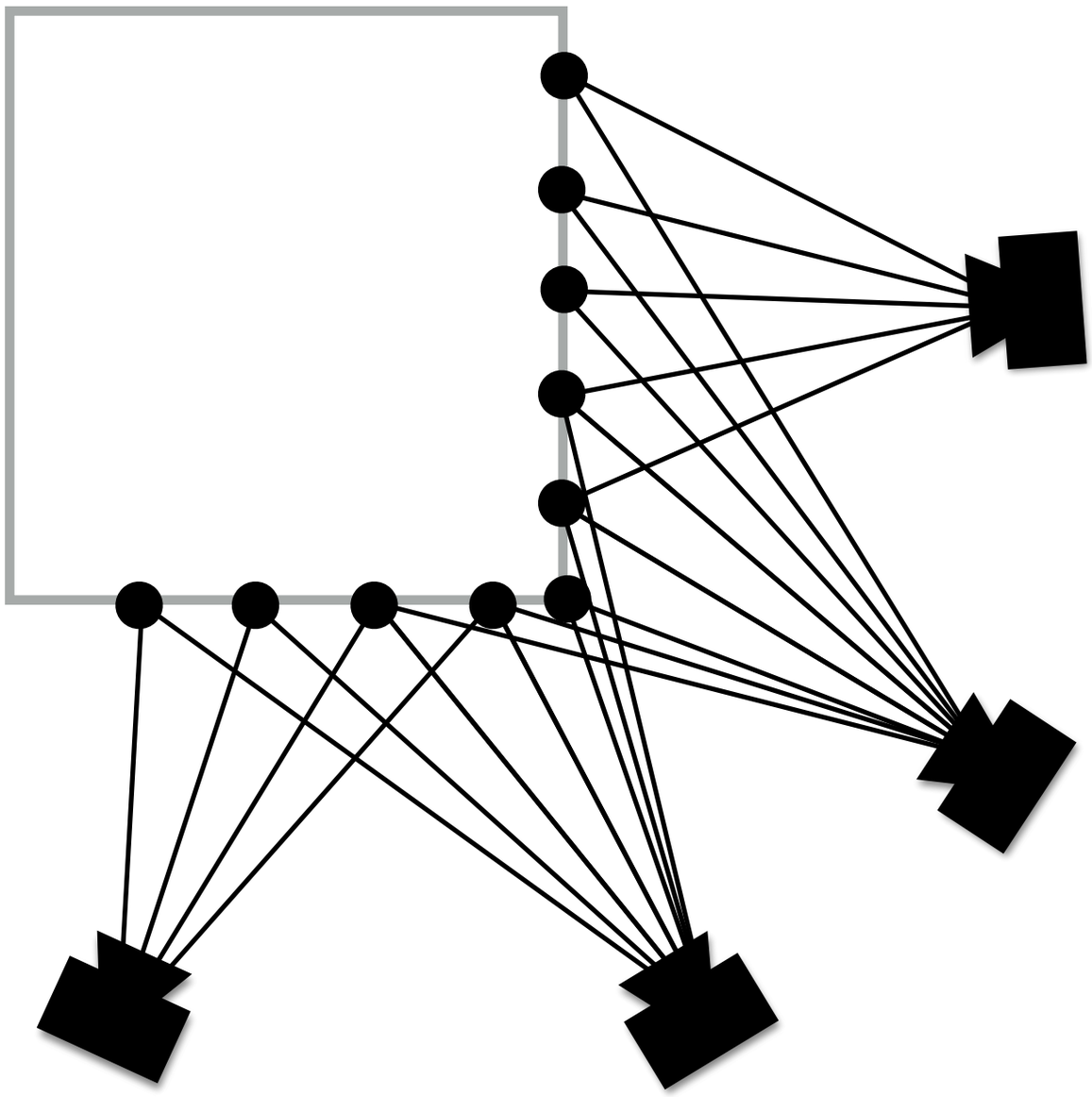
Incremental SfM



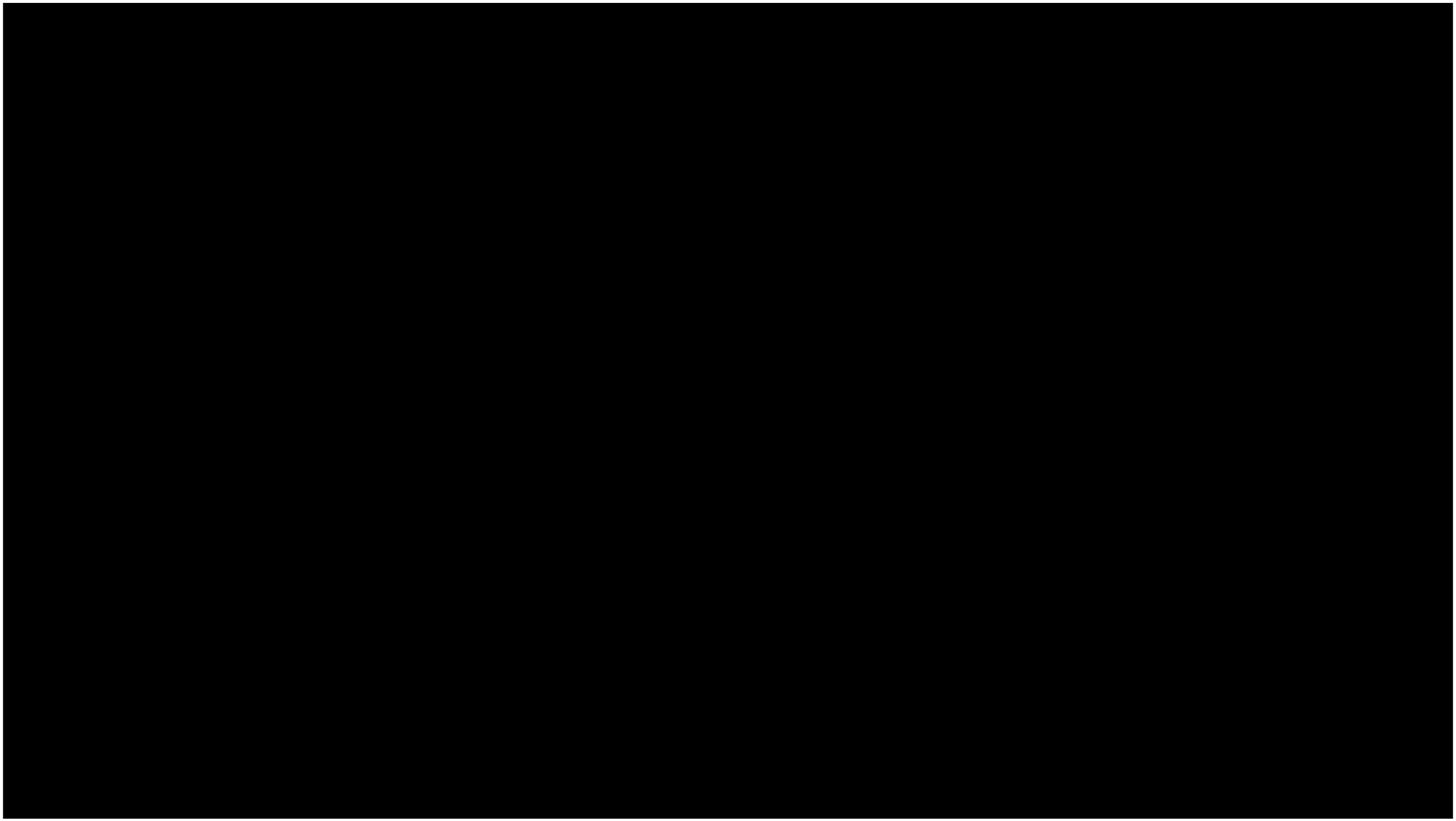
Incremental SfM



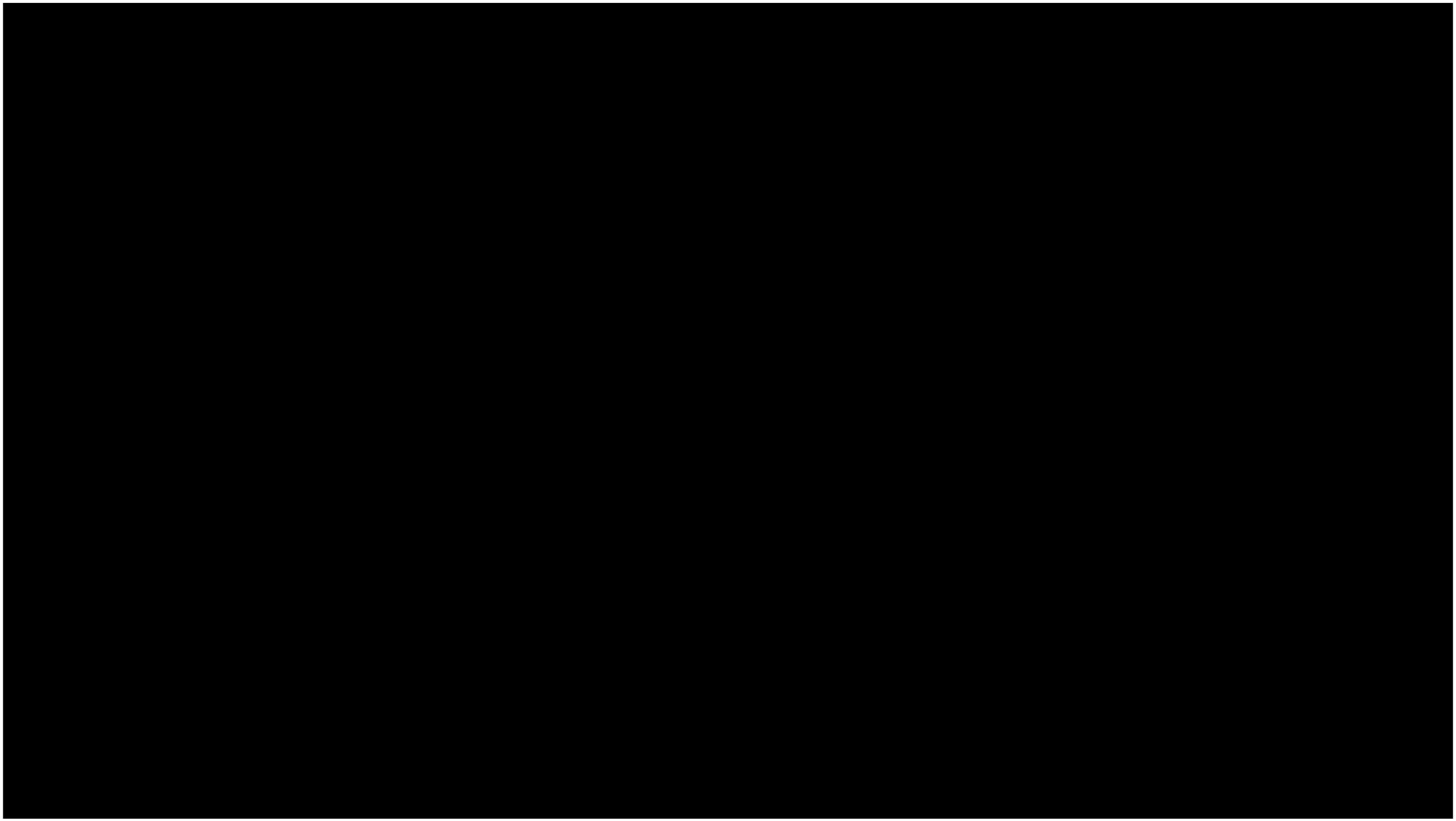
Incremental SfM



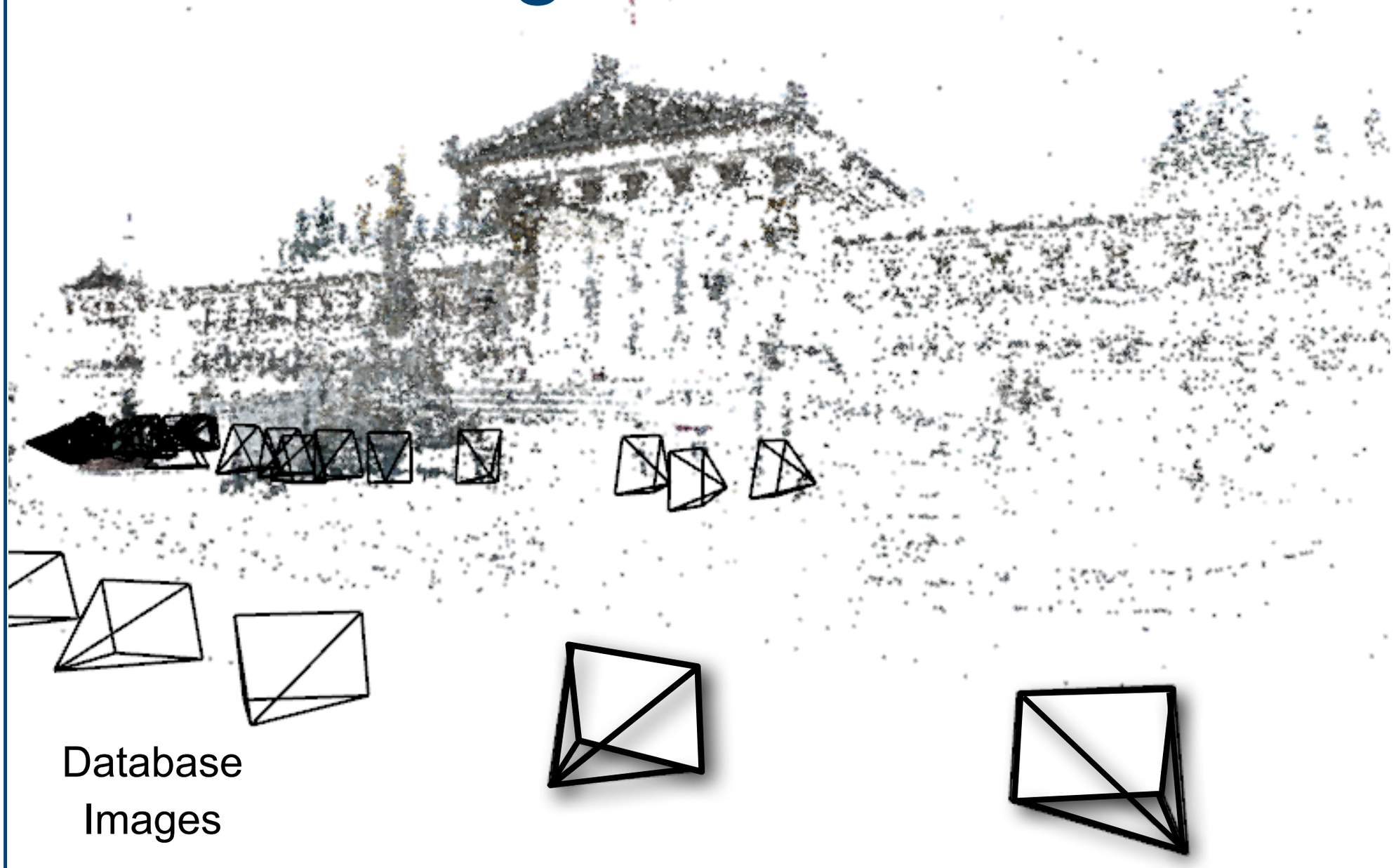
Incremental Structure-From-Motion



Incremental Structure-From-Motion

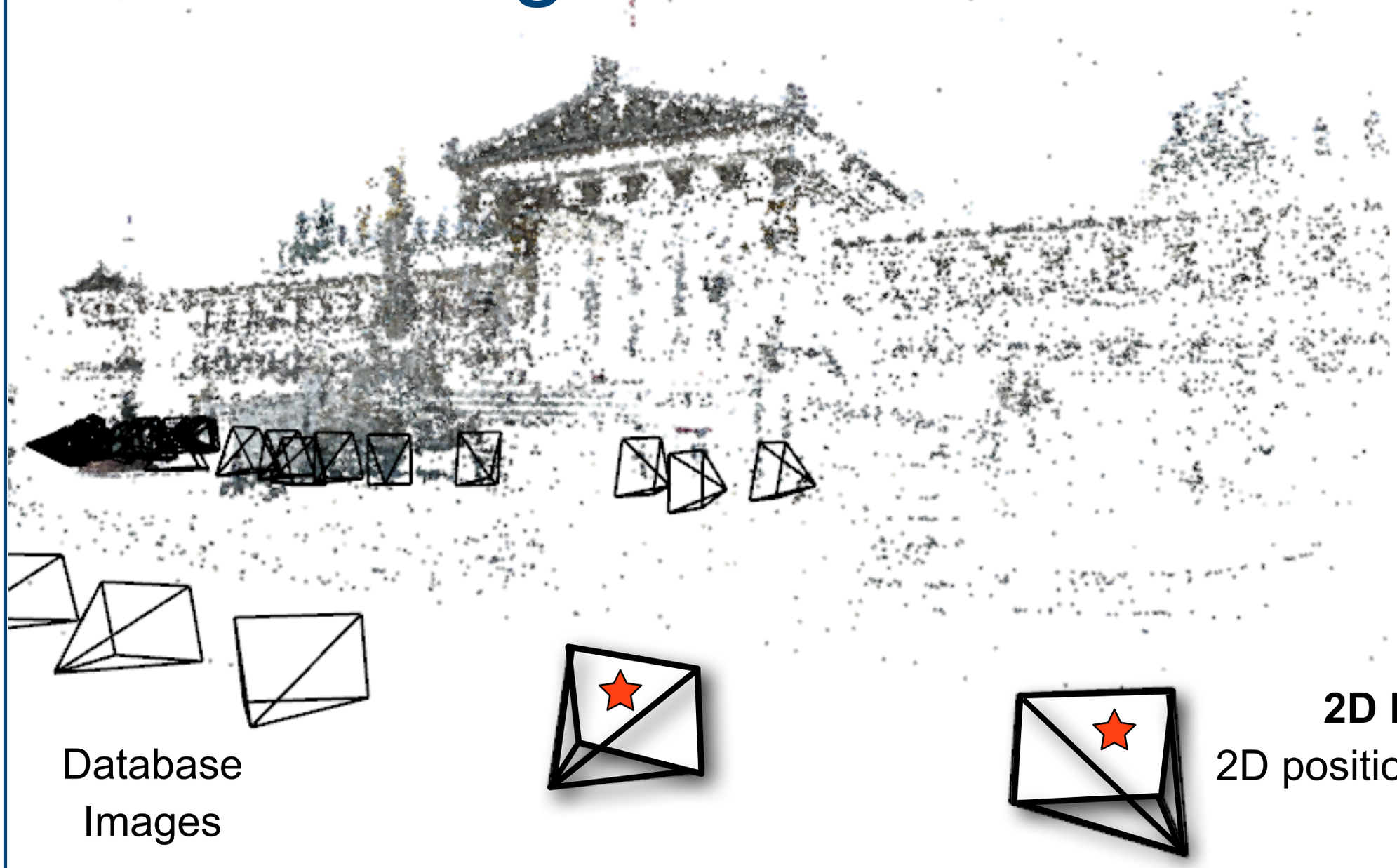


Establishing 2D-3D Matches



- 3D model from SfM

Establishing 2D-3D Matches

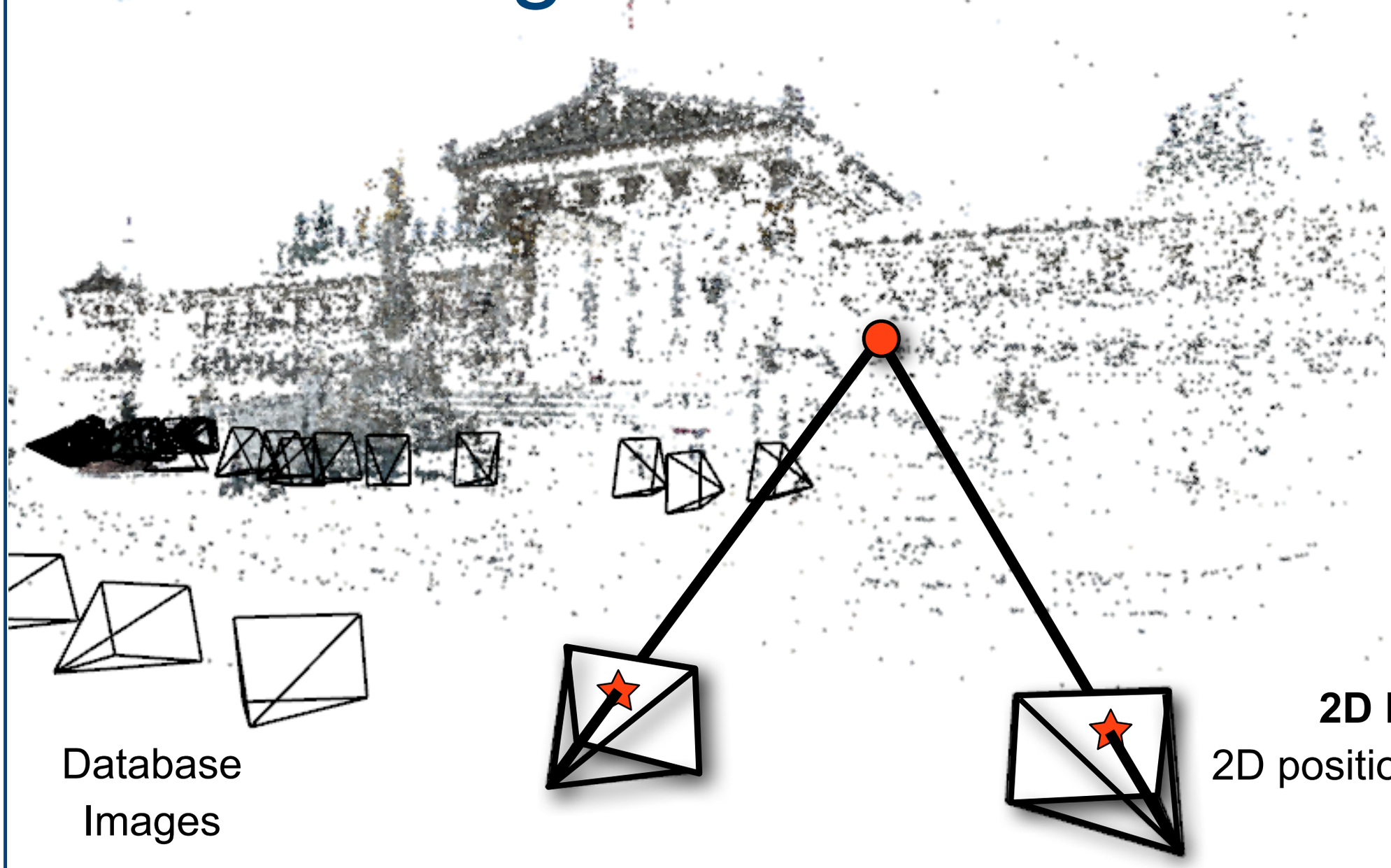


Database Images

2D Feature:
2D position + descriptor

- 3D model from SfM

Establishing 2D-3D Matches

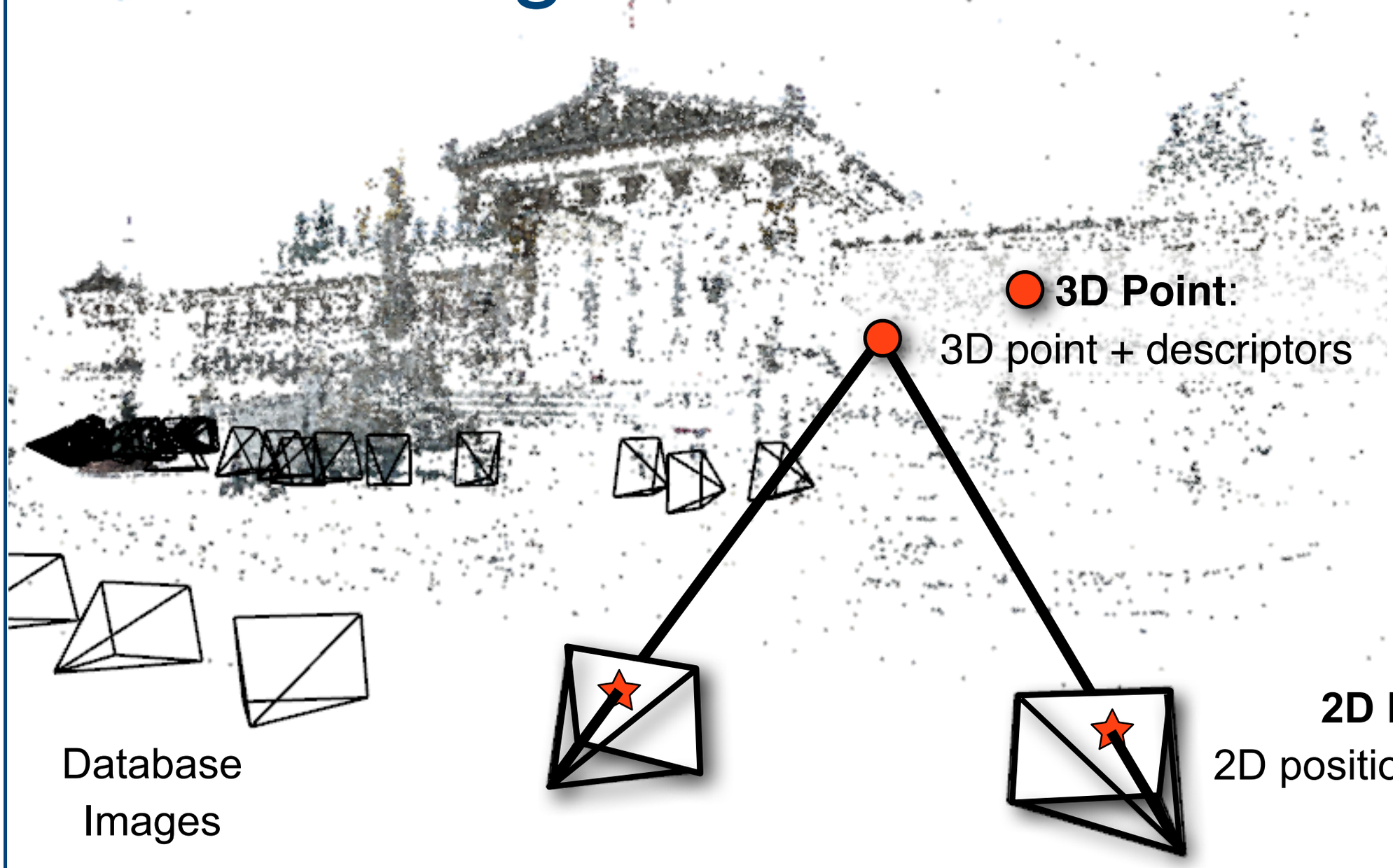


Database Images

2D Feature:
2D position + descriptor

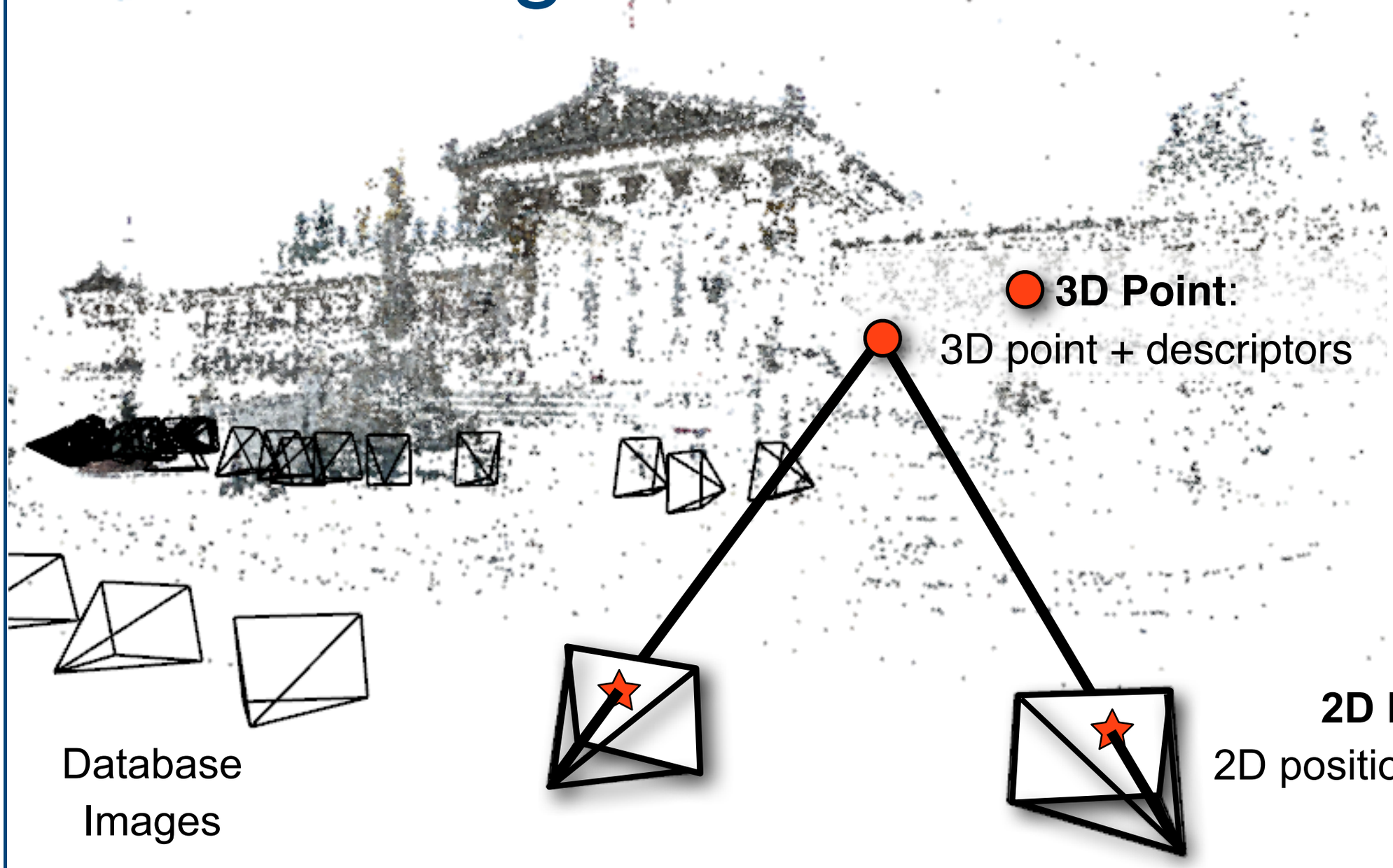
- 3D model from SfM

Establishing 2D-3D Matches



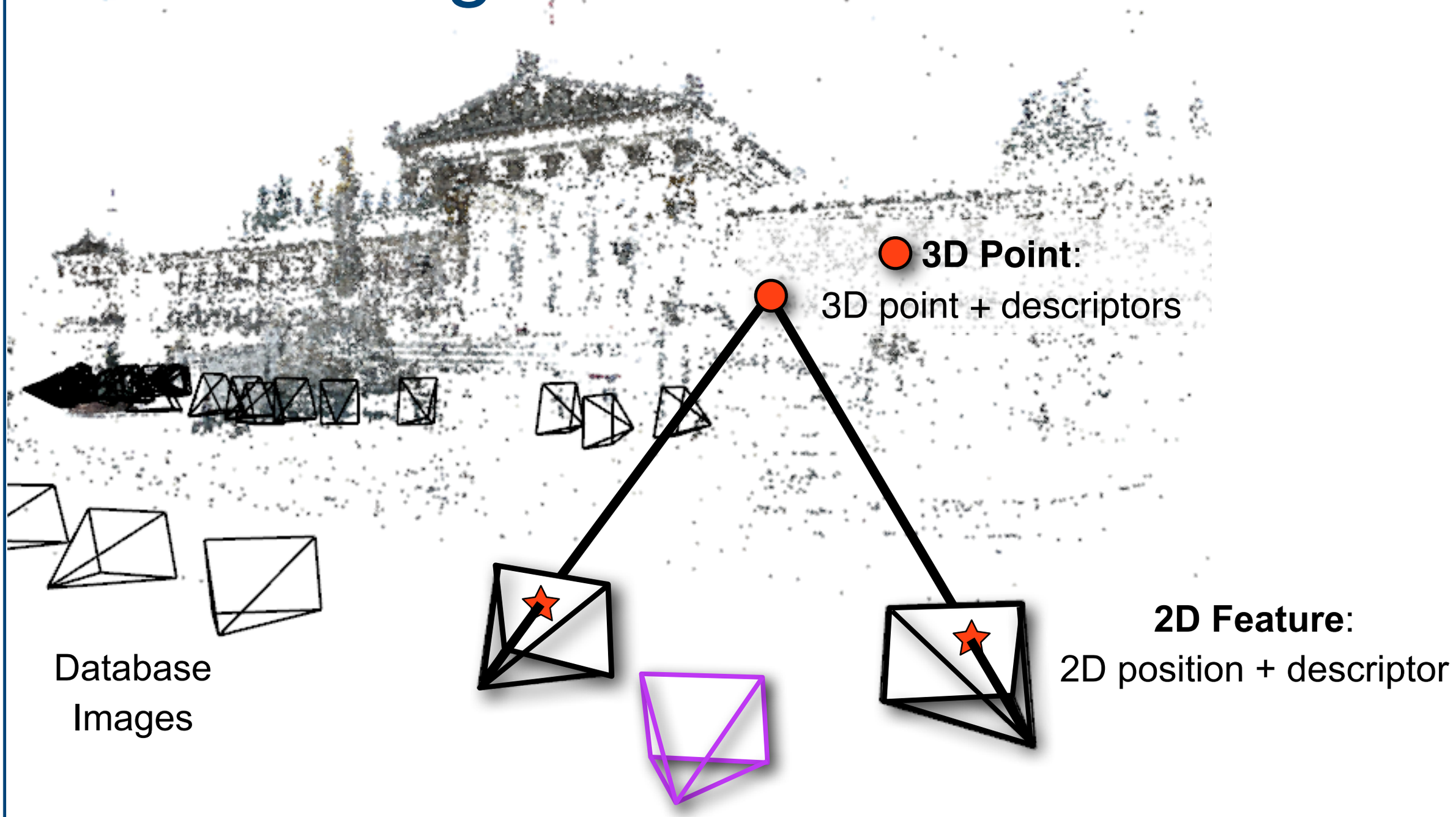
- 3D model from SfM

Establishing 2D-3D Matches



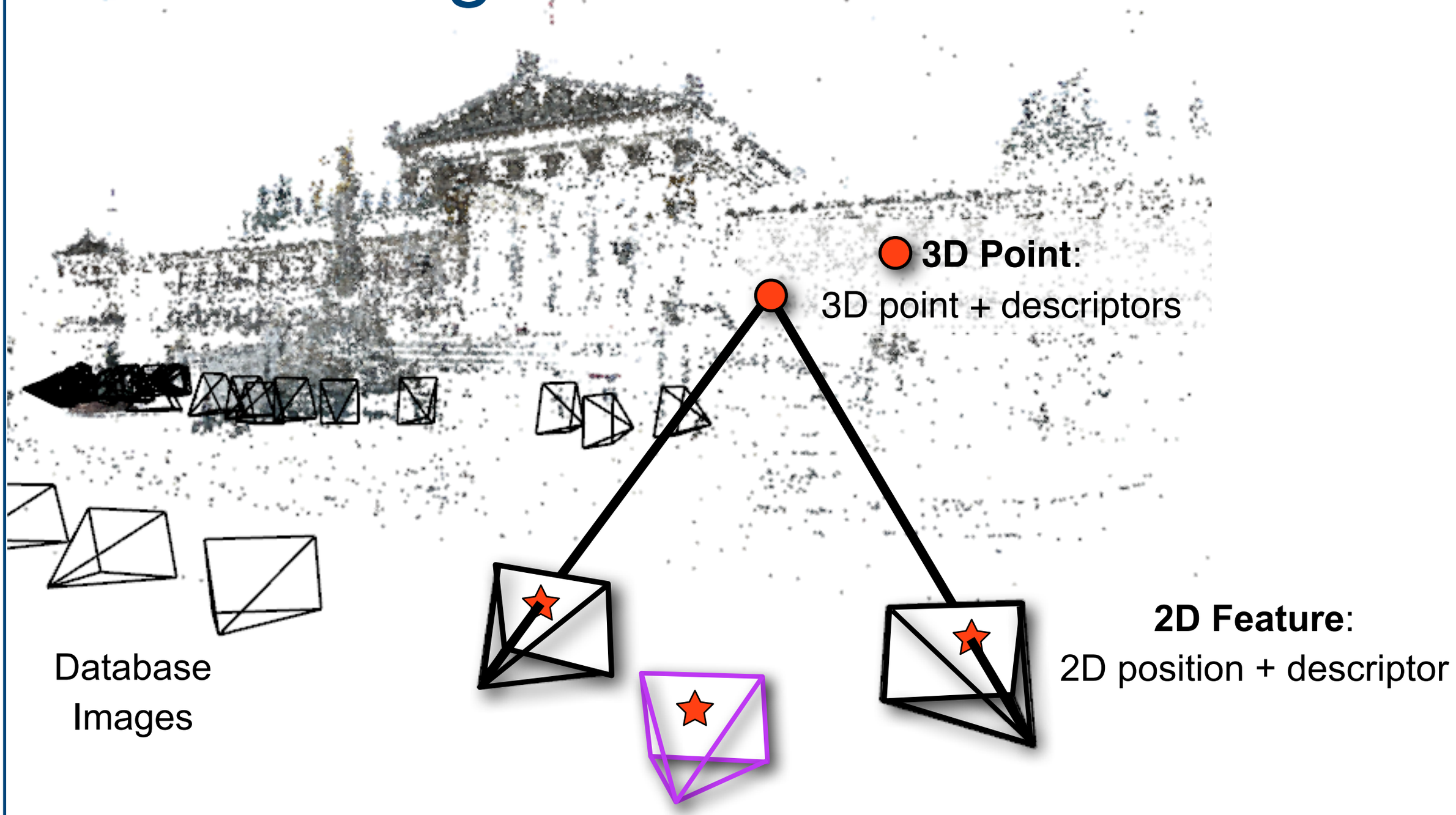
- 3D model from SfM
- 2D-3D correspondences from (SIFT) **descriptor matching**

Establishing 2D-3D Matches



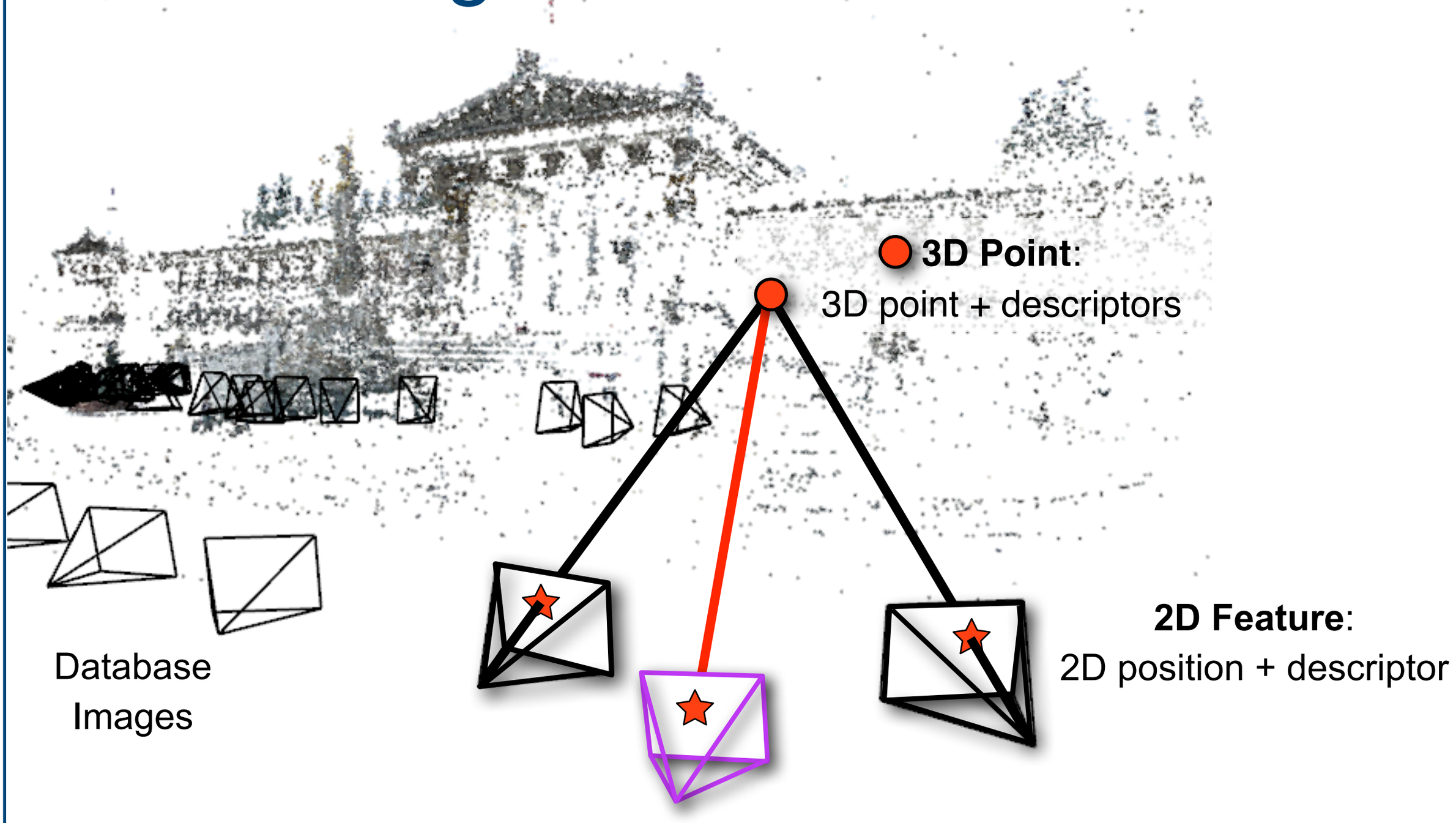
- 3D model from SfM
- 2D-3D correspondences from (SIFT) **descriptor matching**

Establishing 2D-3D Matches



- 3D model from SfM
- 2D-3D correspondences from (SIFT) **descriptor matching**

Establishing 2D-3D Matches



- 3D model from SfM
- 2D-3D correspondences from (SIFT) **descriptor matching**

Image-Based Localization & Place Recognition



Query Image

3D Model

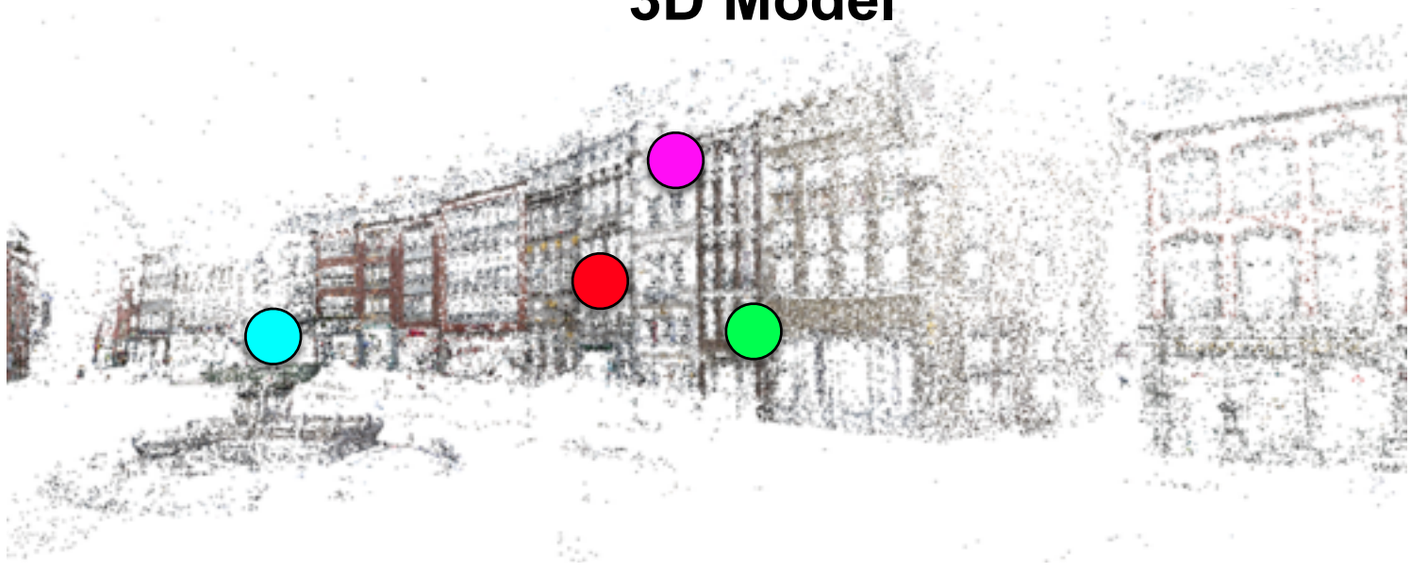


Image-Based Localization & Place Recognition

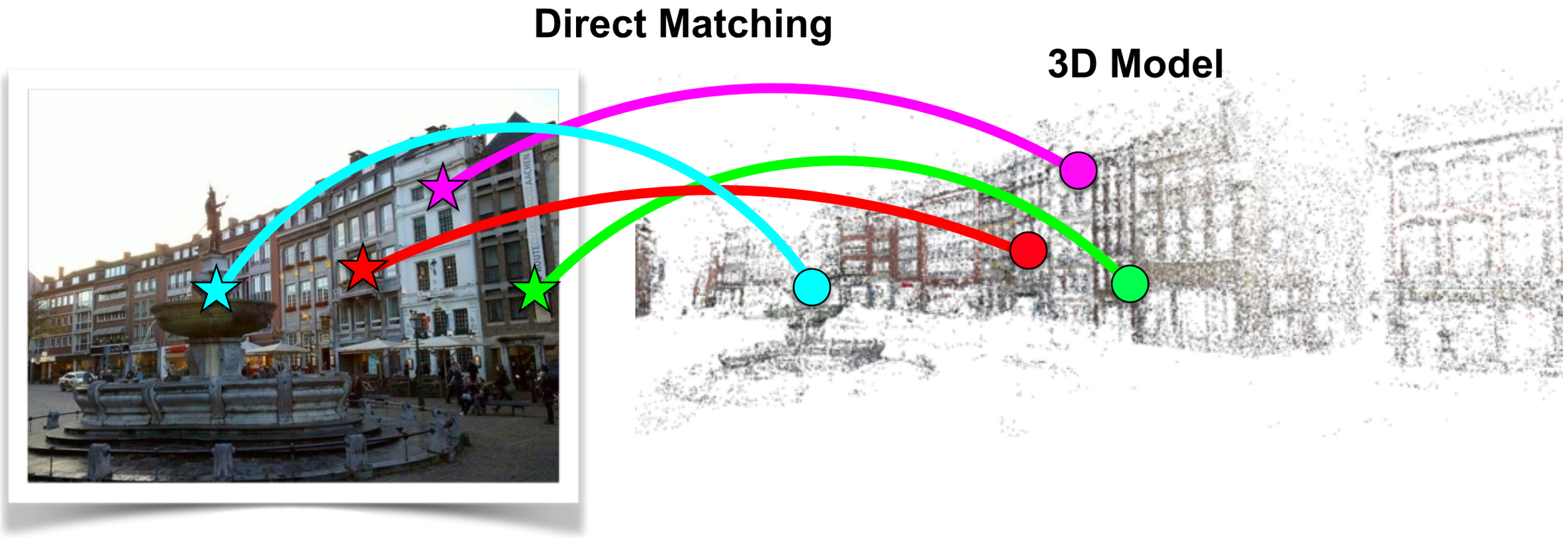
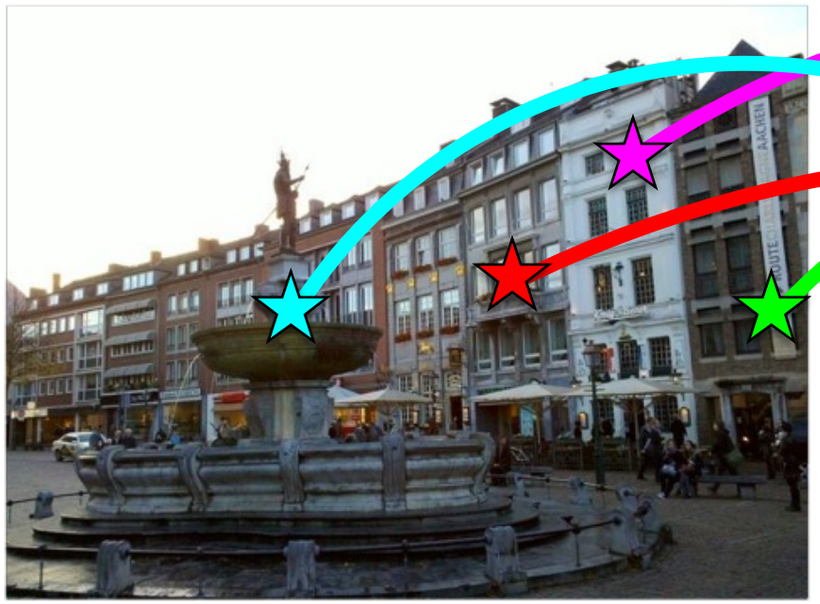


Image-Based Localization & Place Recognition

Direct Matching

3D Model



Query Image



Retrieved Database Image

Image-Based Localization & Place Recognition

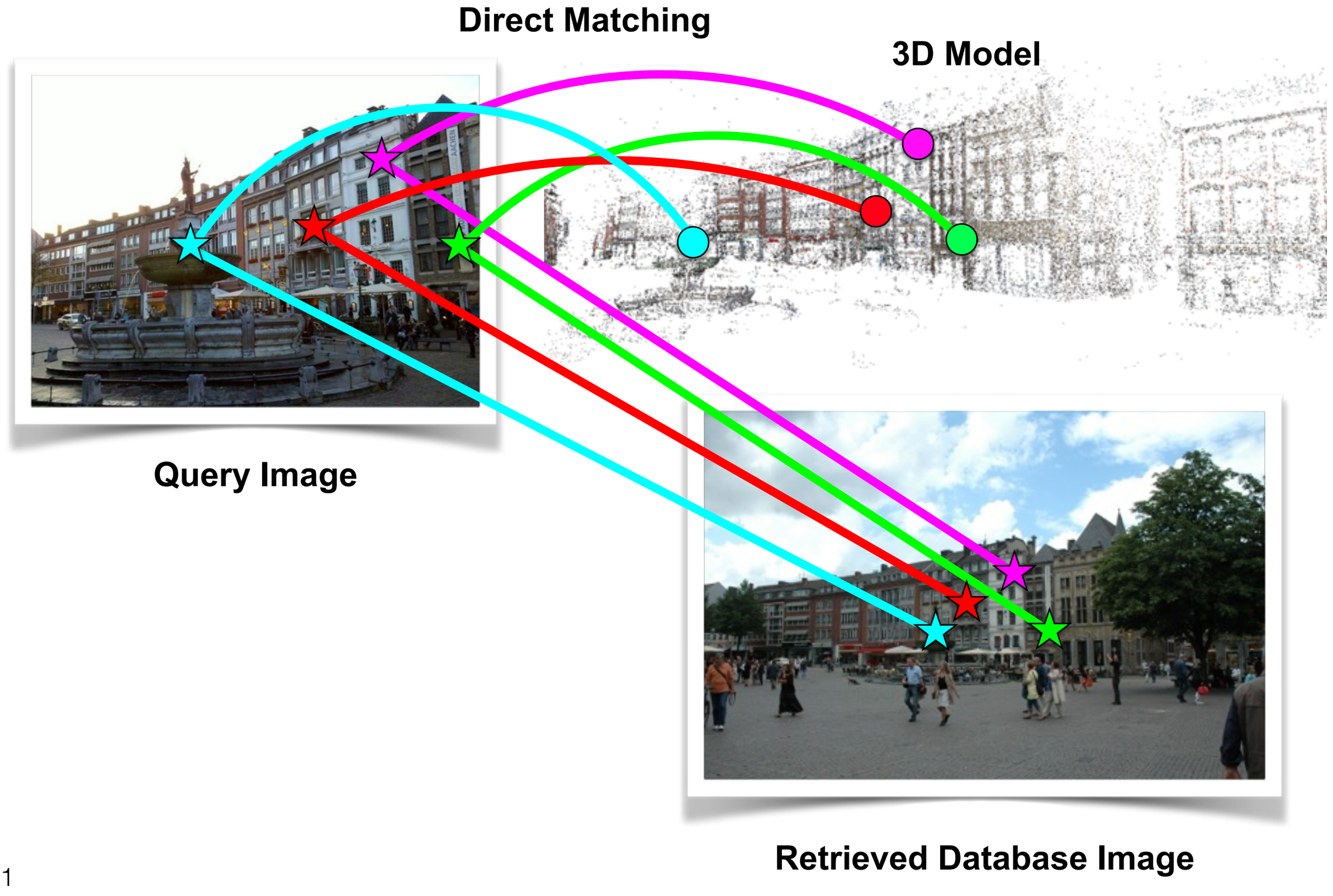
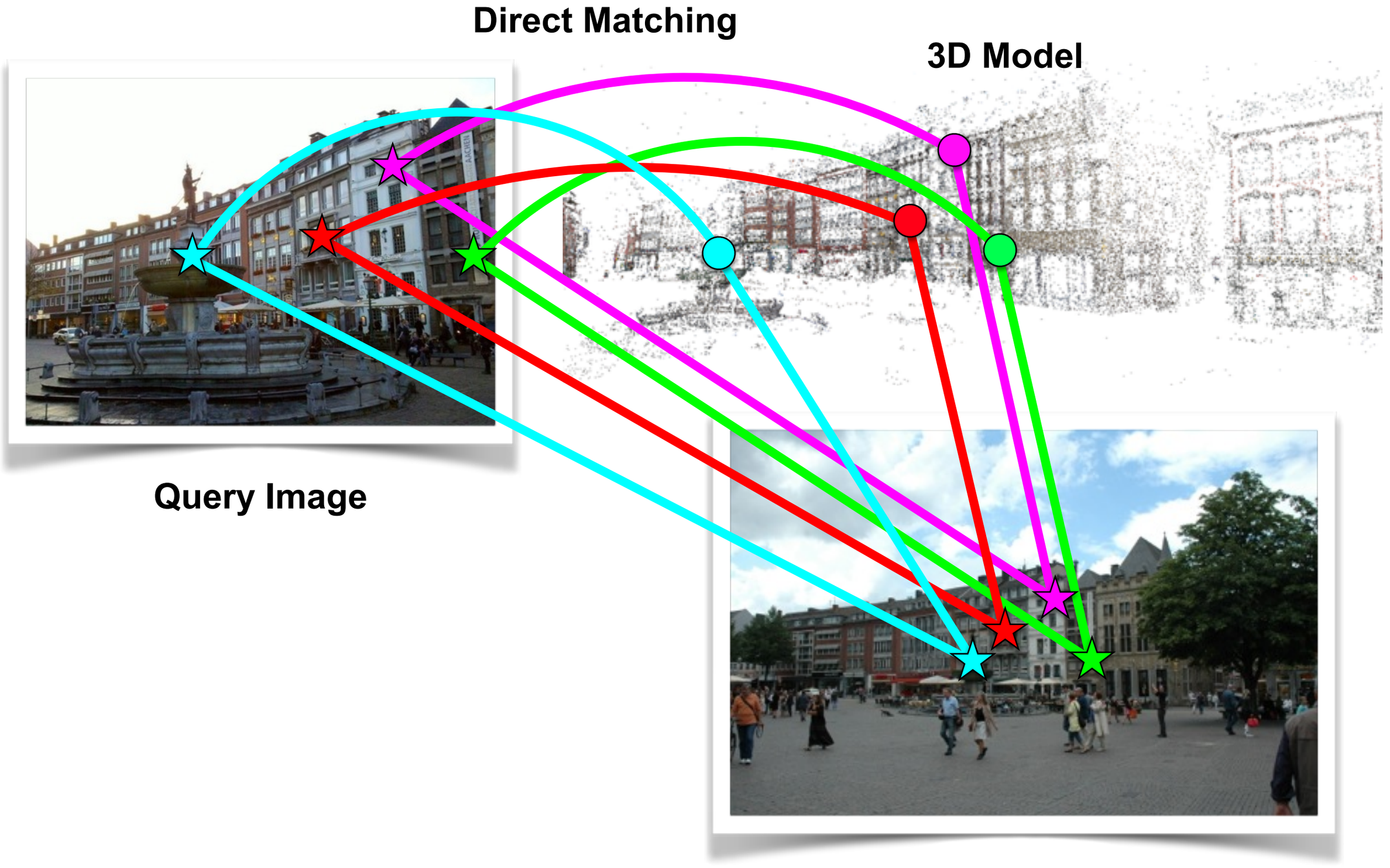


Image-Based Localization & Place Recognition



Fundamentals

Structure-from-Motion

Establish 2D-3D Matches

Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

(This is meant as a short overview, no in-depth discussion)

Fundamentals

Structure-from-Motion

Establish 2D-3D Matches

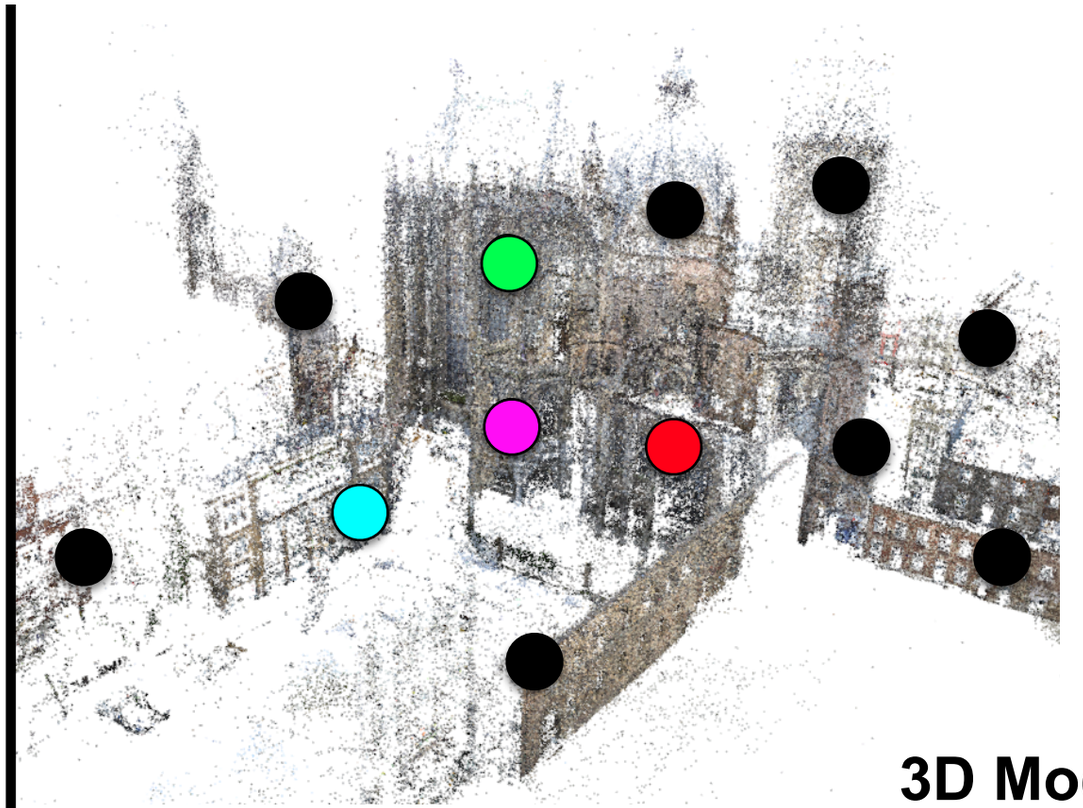
Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

(This is meant as a short overview, no in-depth discussion)

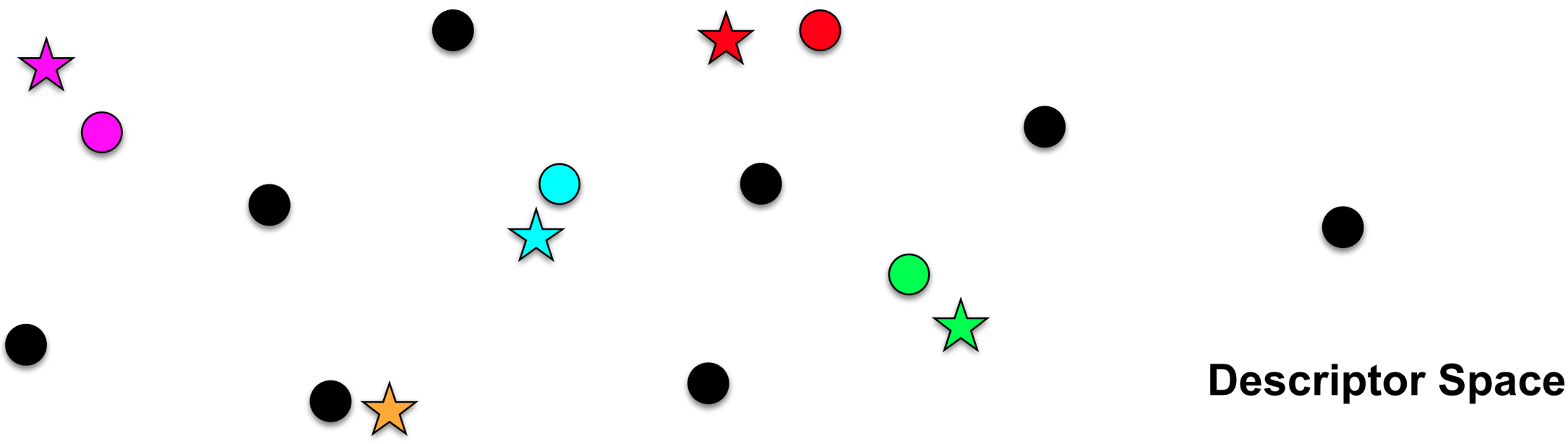
Matching = Nearest Neighbor Search



Query Image



3D Model

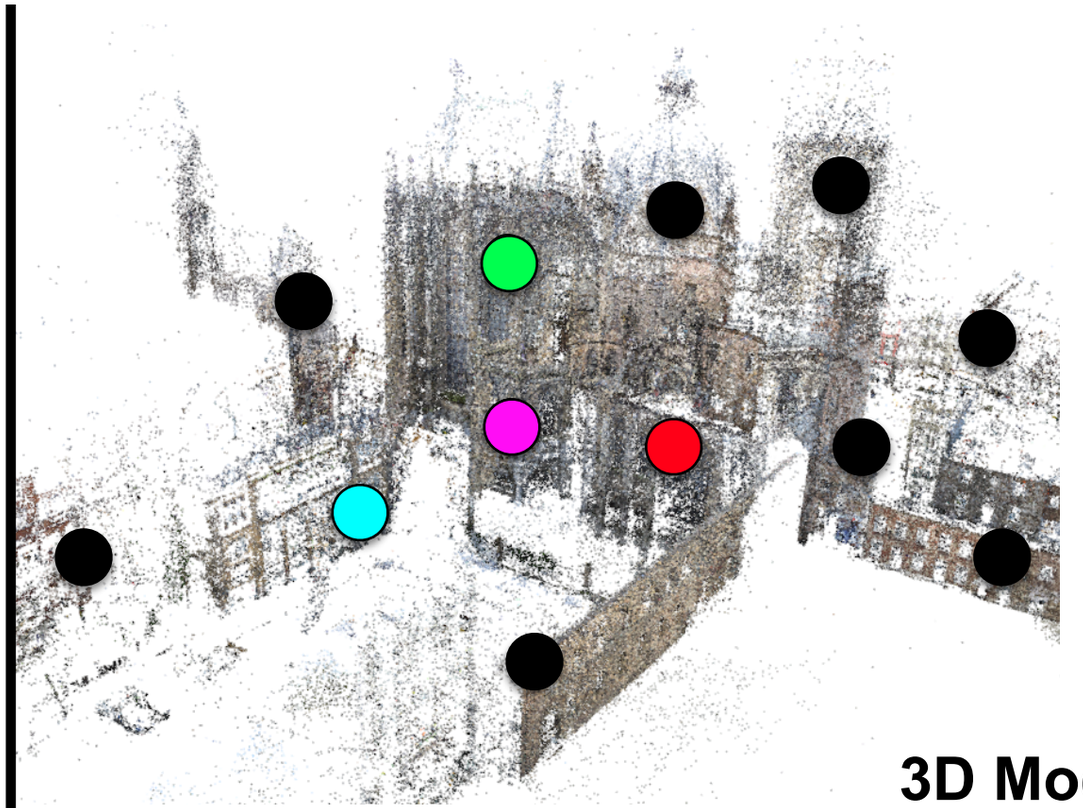


Descriptor Space

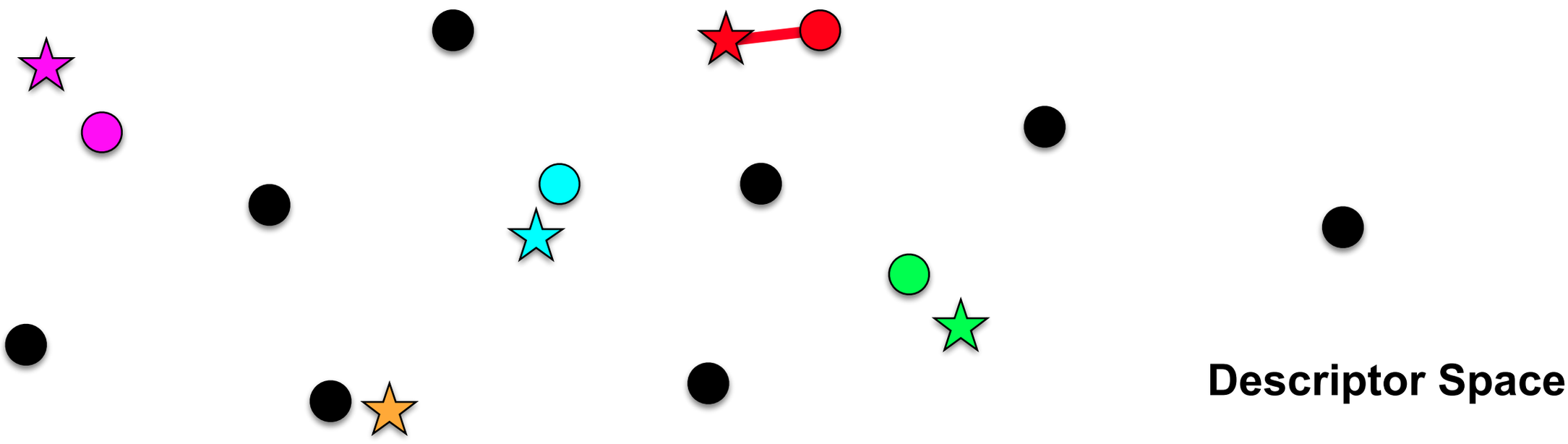
Matching = Nearest Neighbor Search



Query Image



3D Model

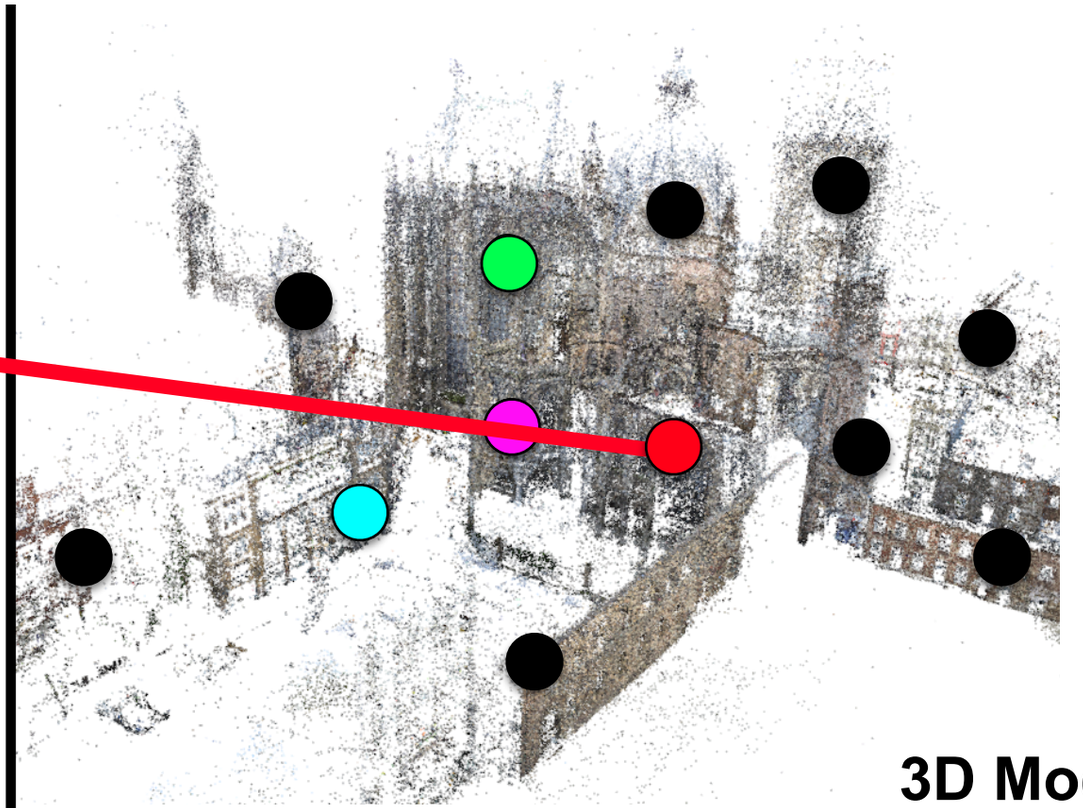


Descriptor Space

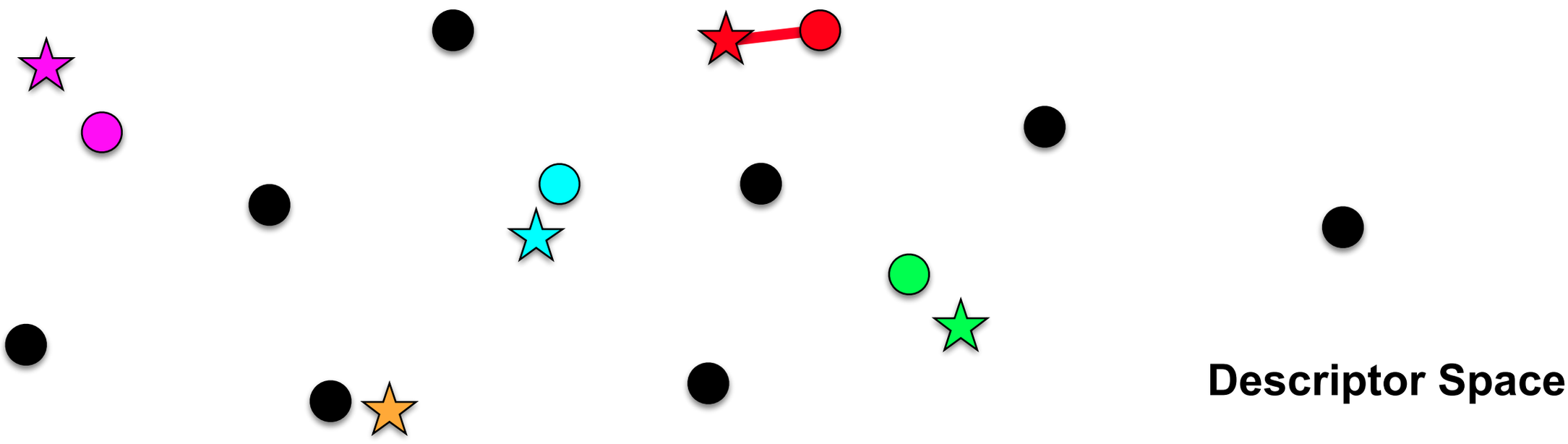
Matching = Nearest Neighbor Search



Query Image



3D Model

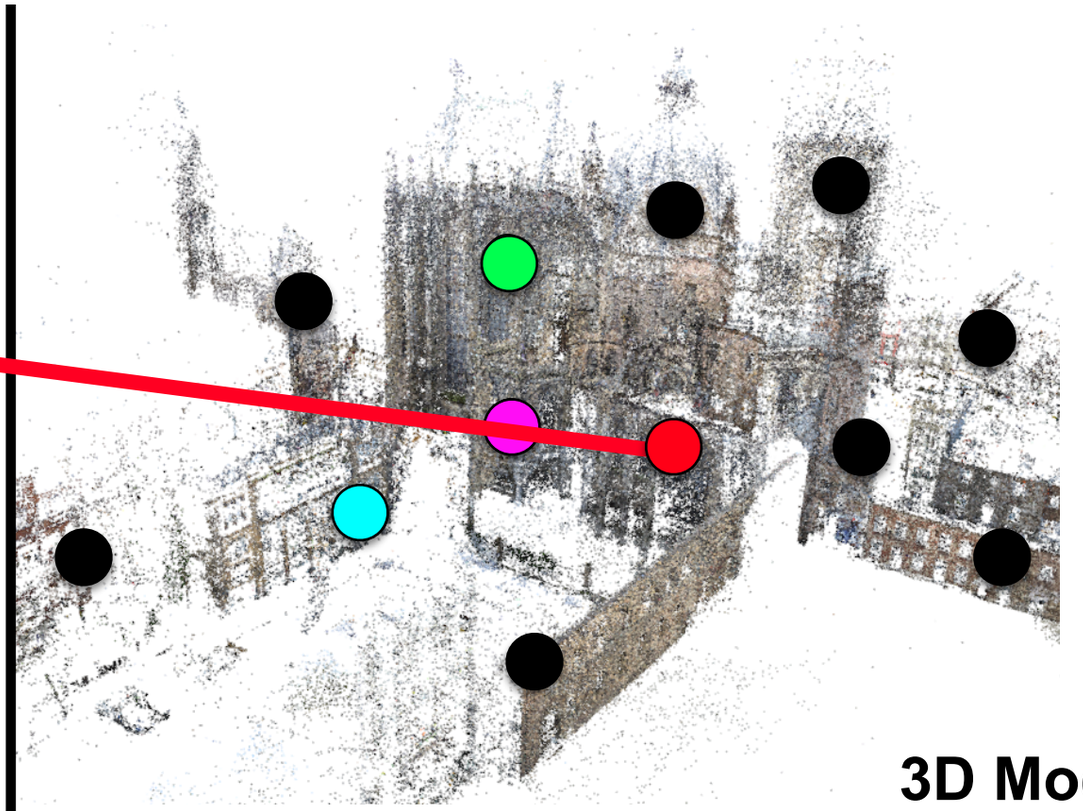


Descriptor Space

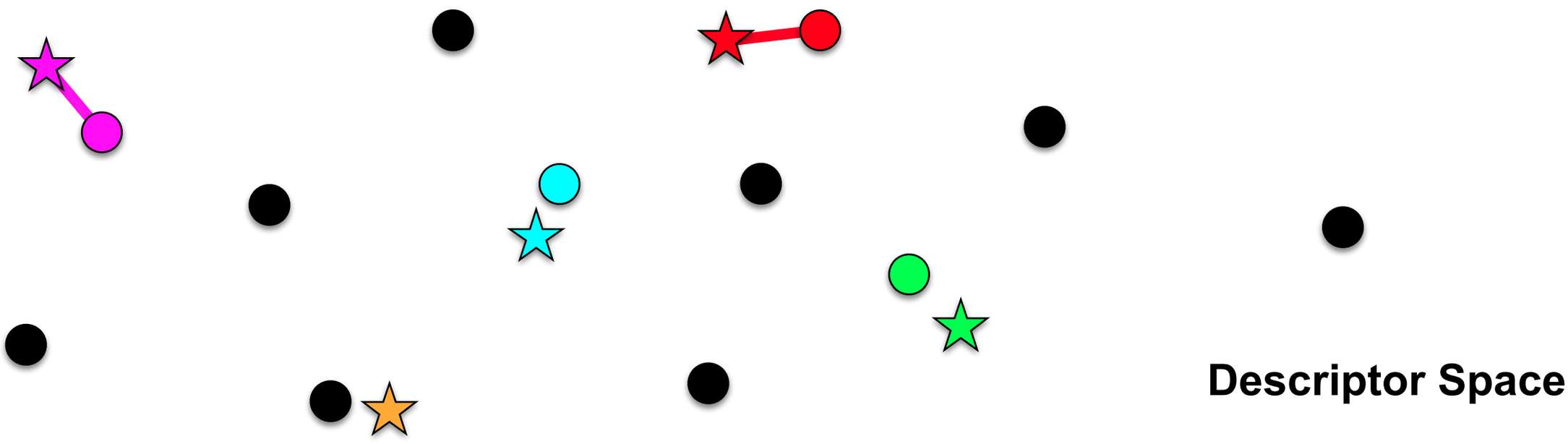
Matching = Nearest Neighbor Search



Query Image

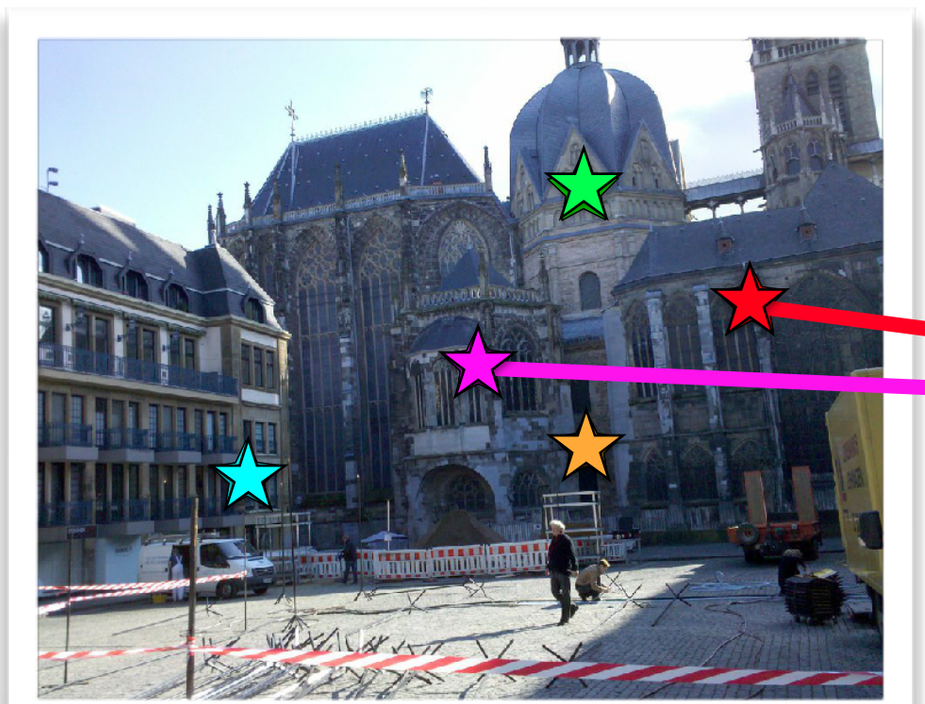


3D Model

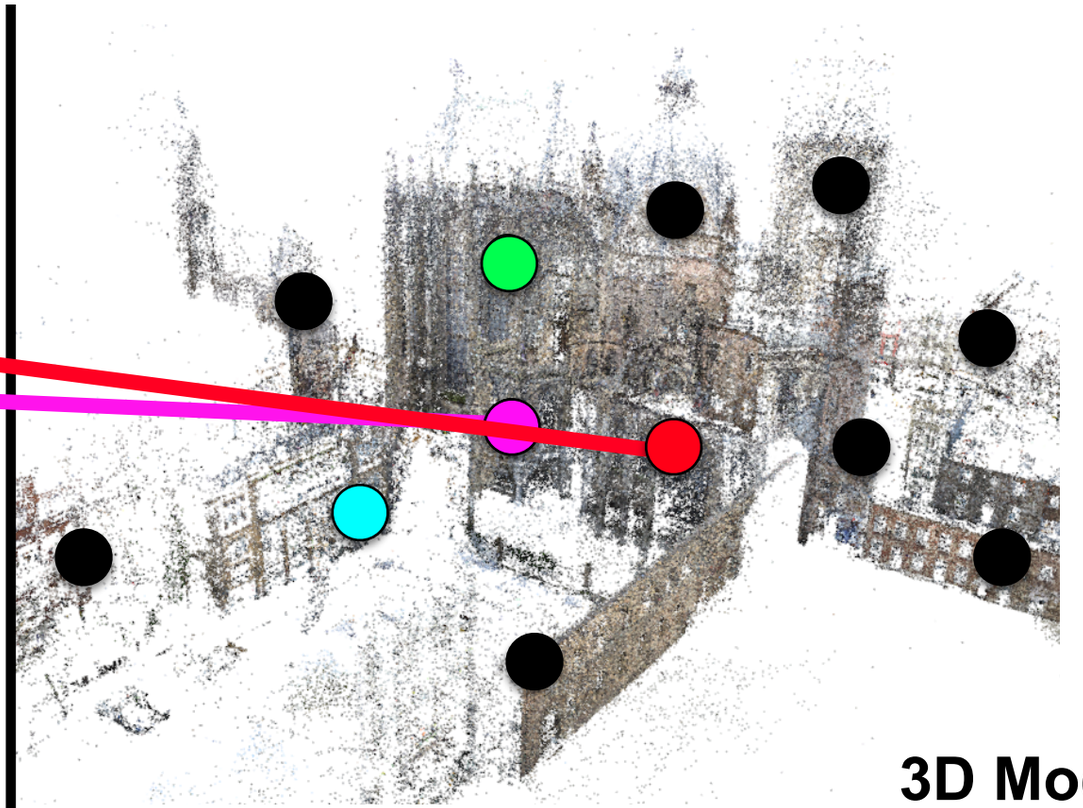


Descriptor Space

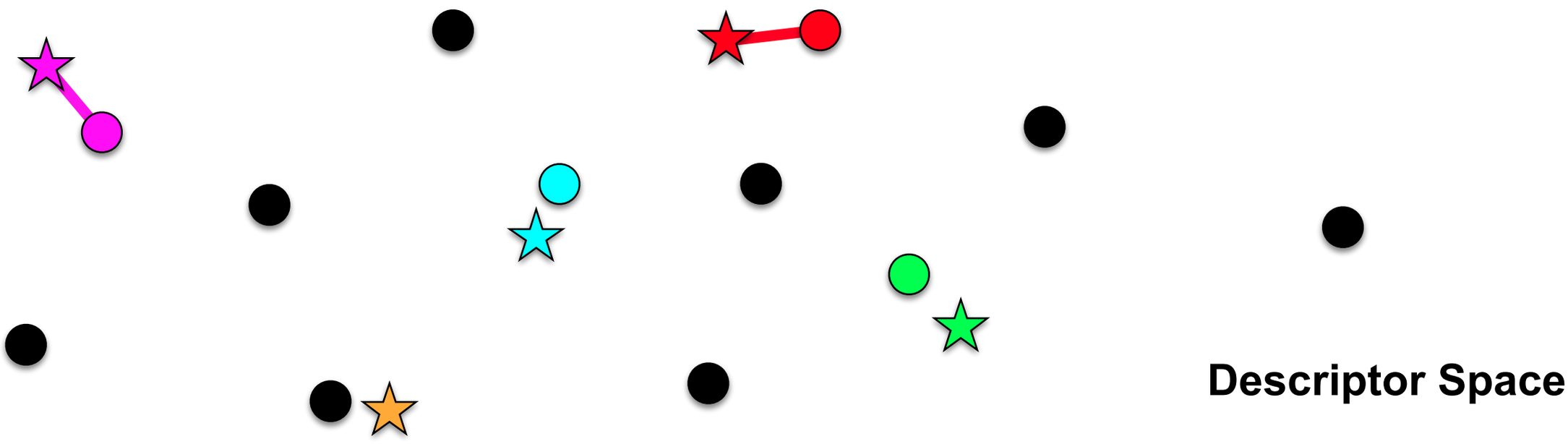
Matching = Nearest Neighbor Search



Query Image

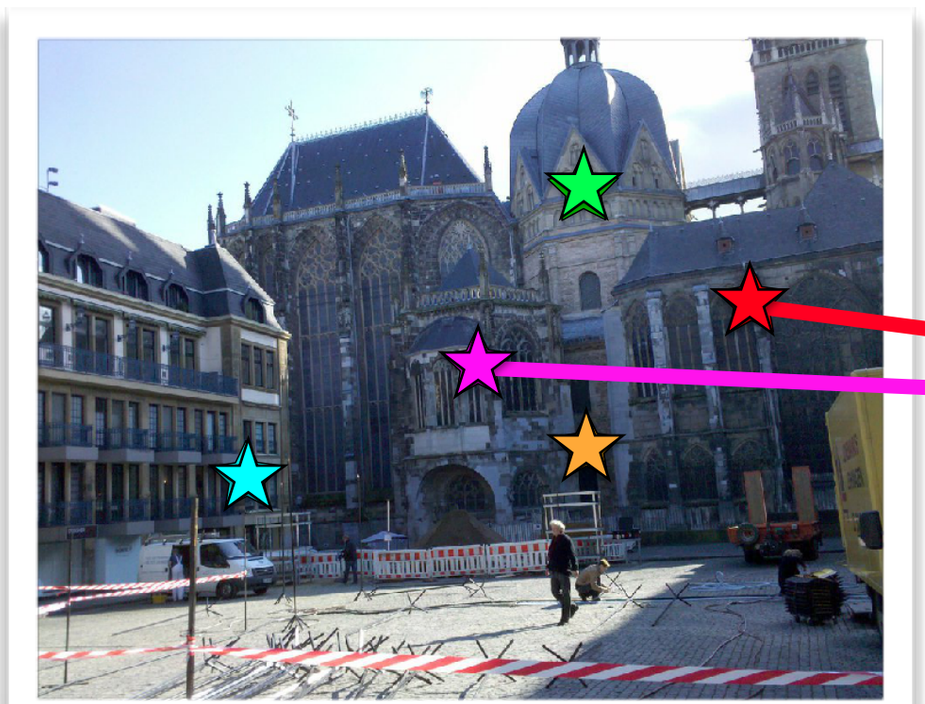


3D Model

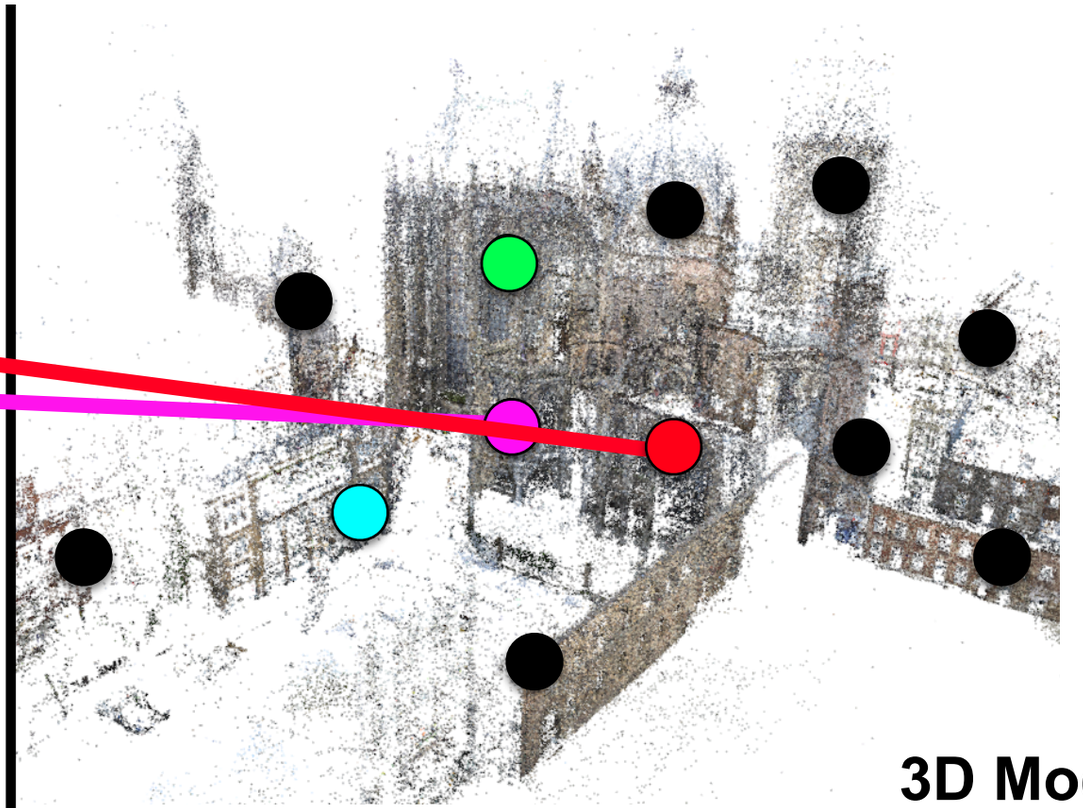


Descriptor Space

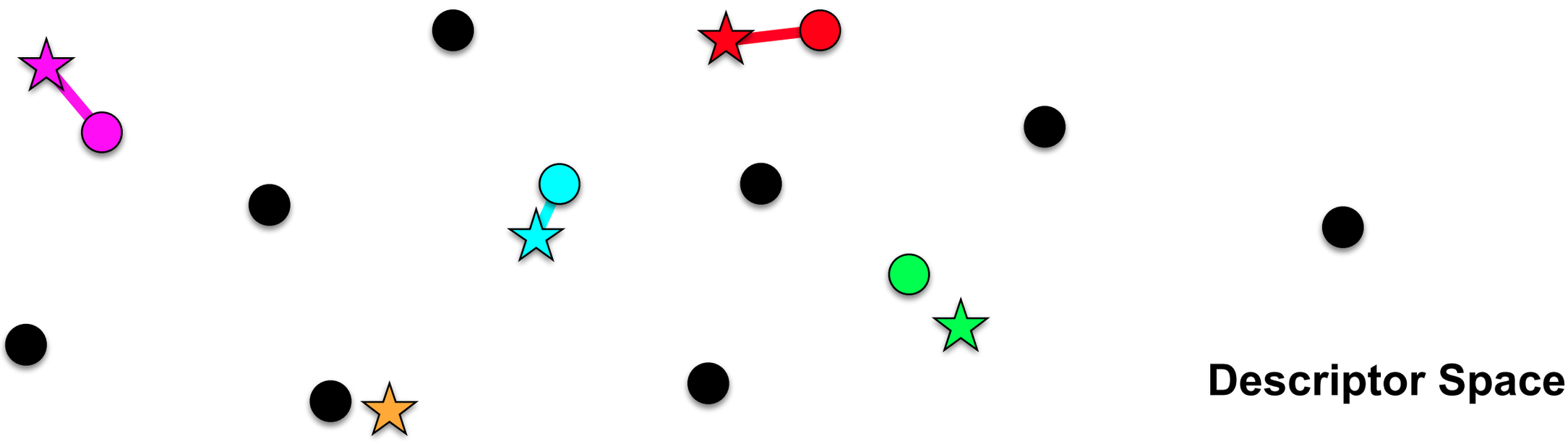
Matching = Nearest Neighbor Search



Query Image

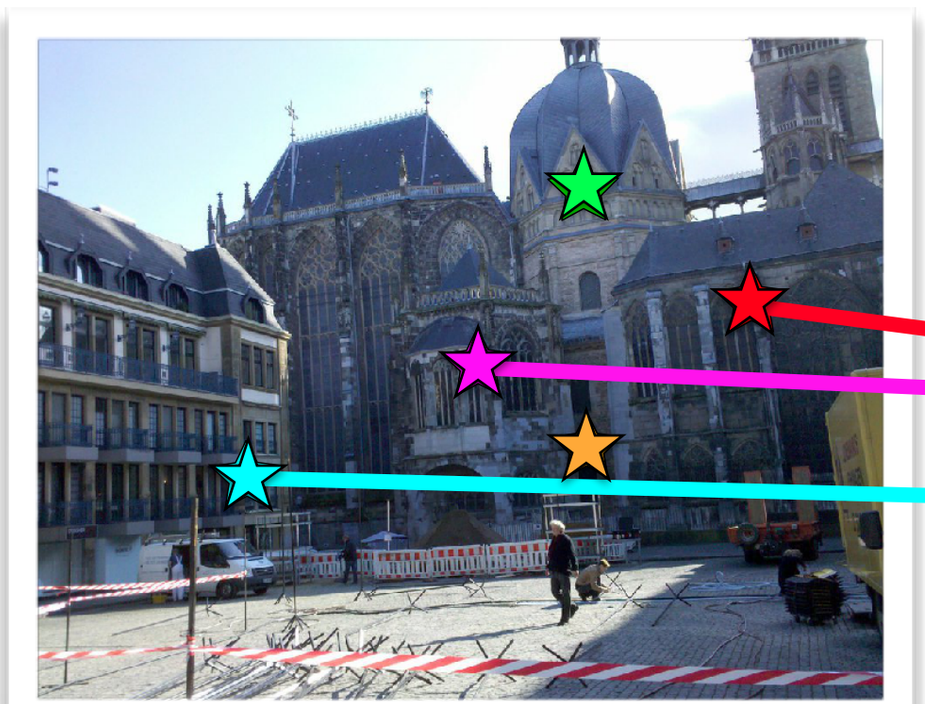


3D Model

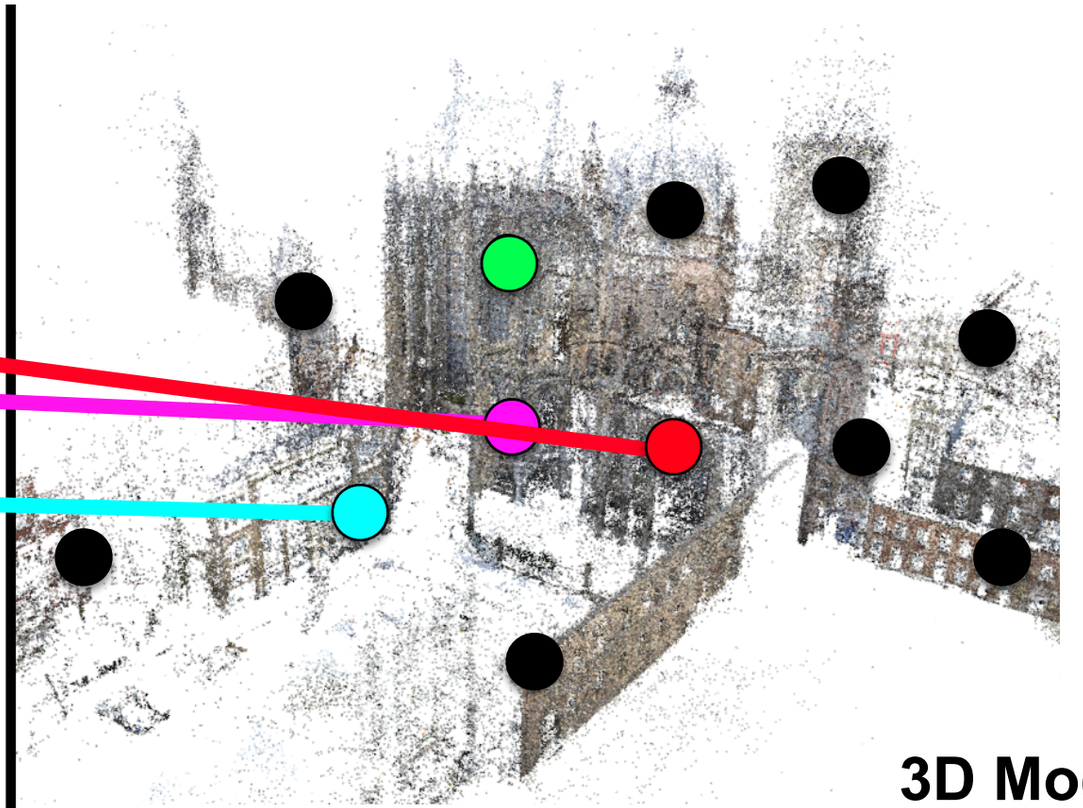


Descriptor Space

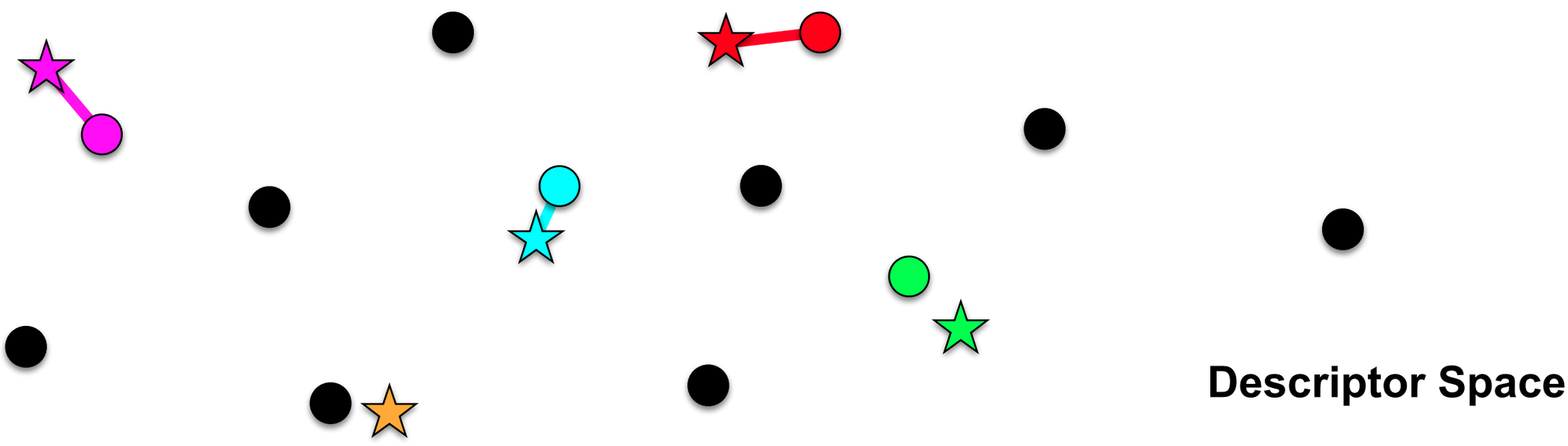
Matching = Nearest Neighbor Search



Query Image

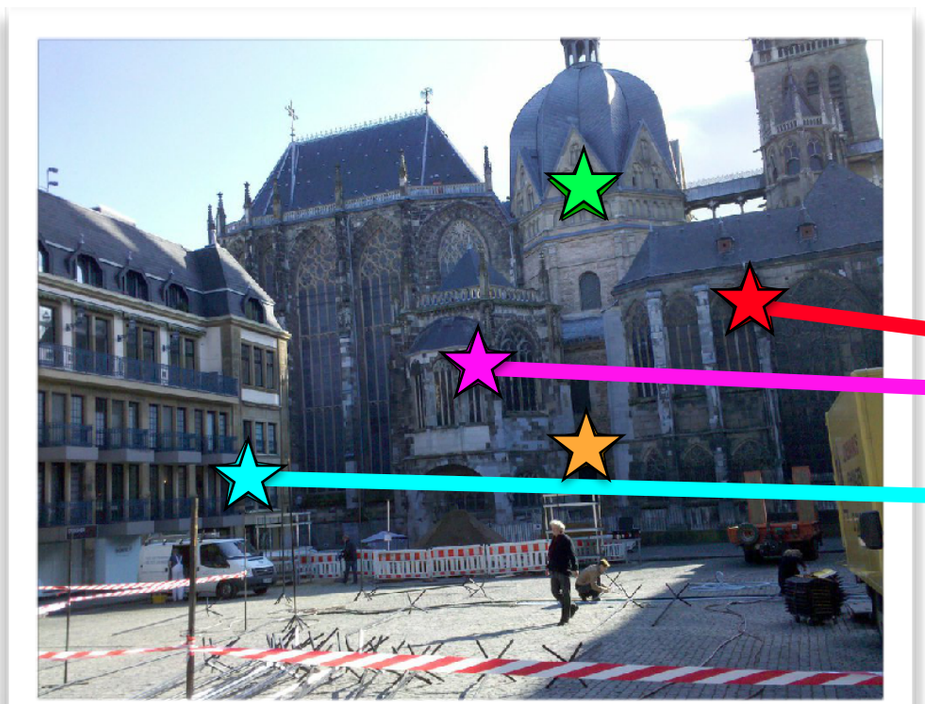


3D Model

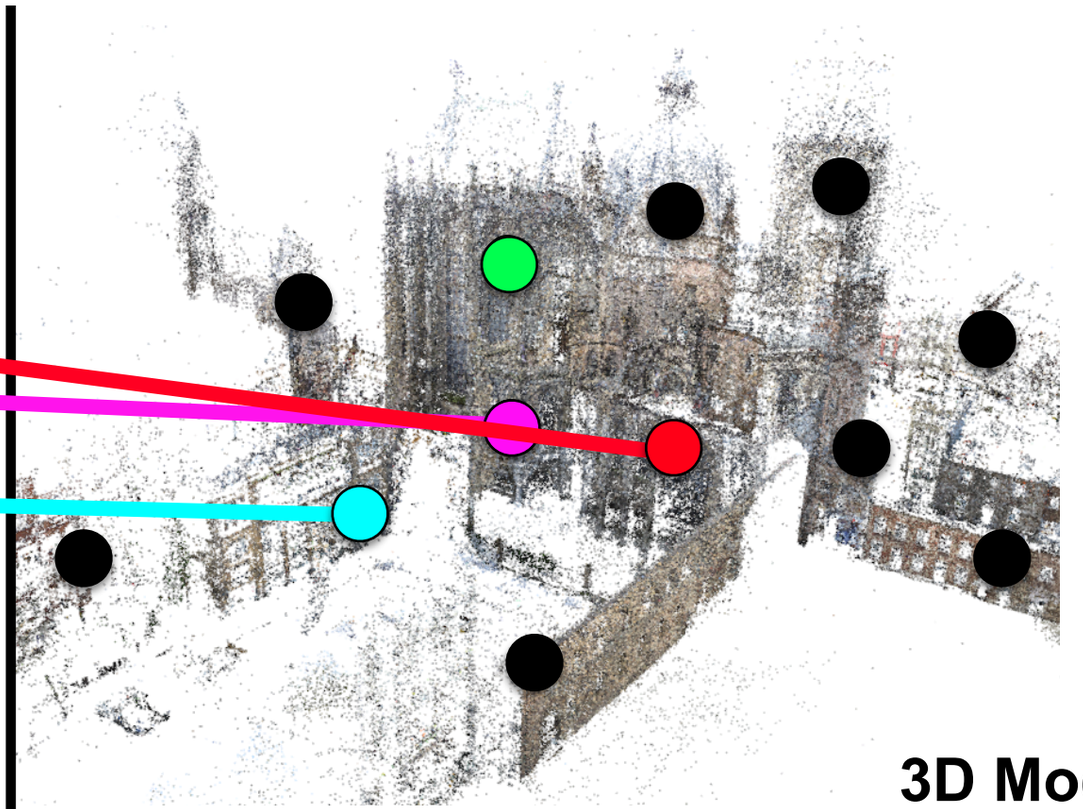


Descriptor Space

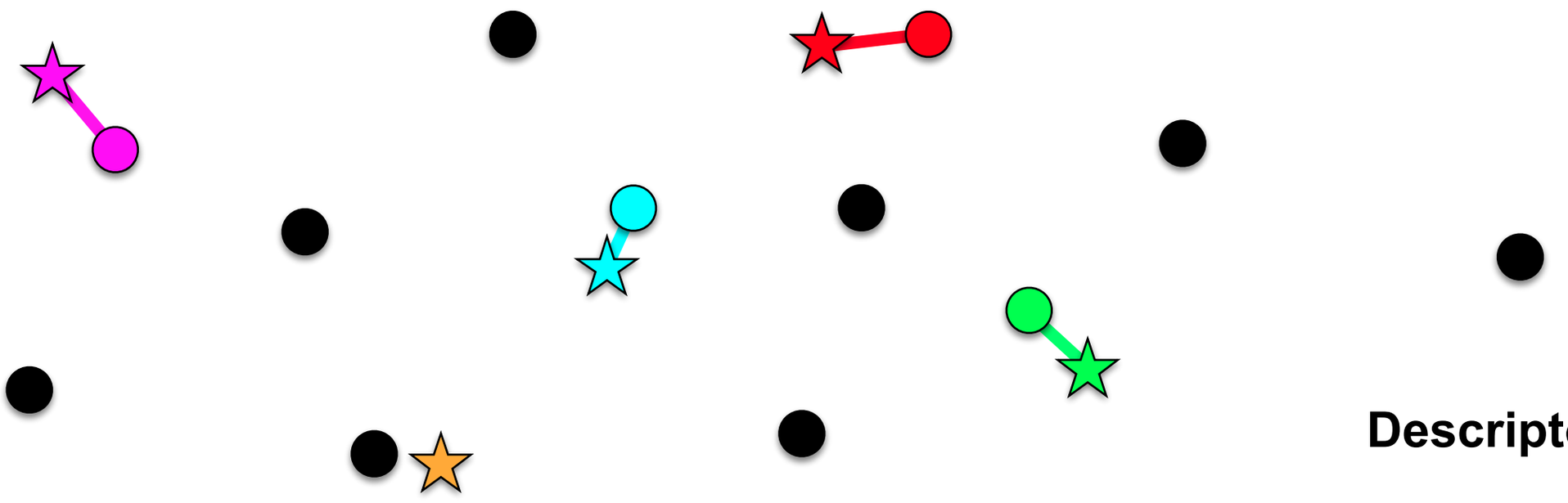
Matching = Nearest Neighbor Search



Query Image

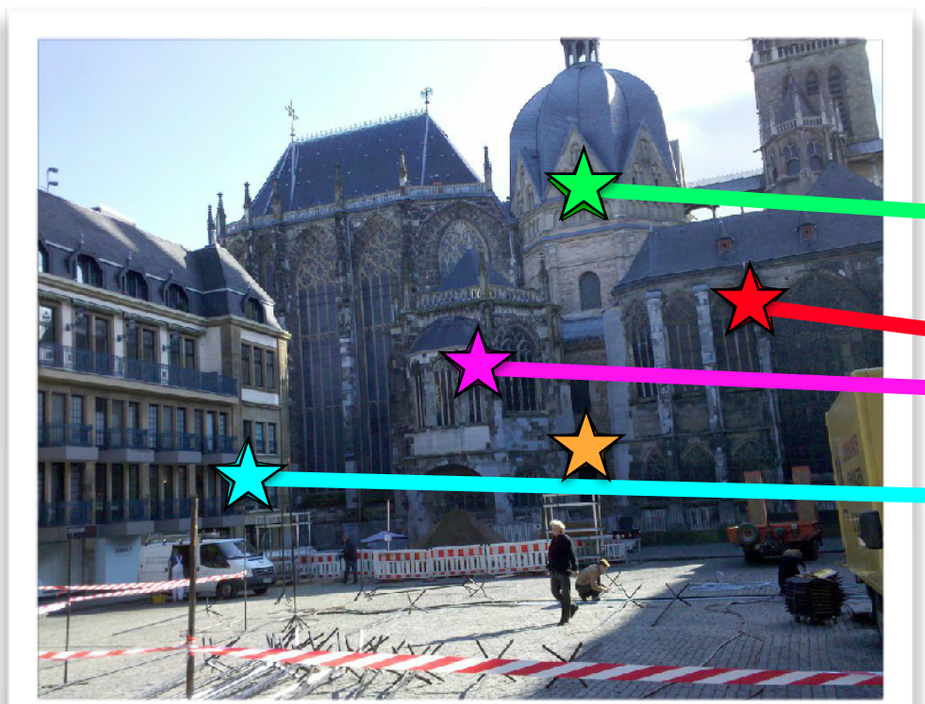


3D Model

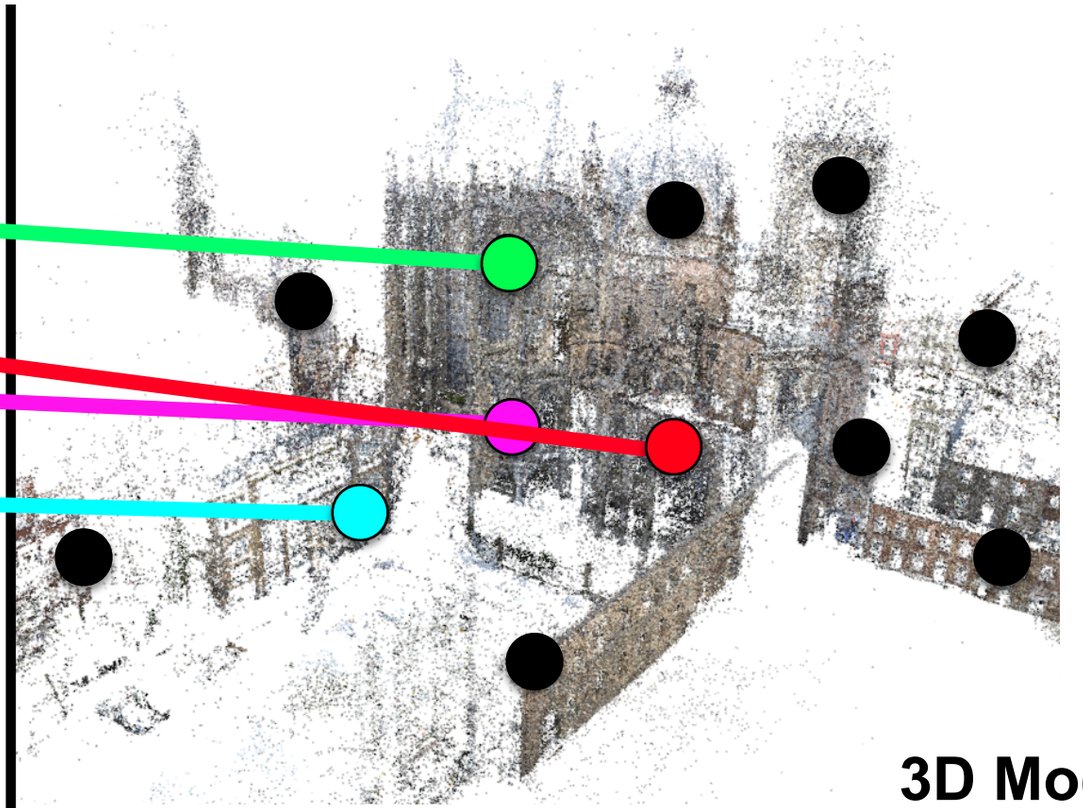


Descriptor Space

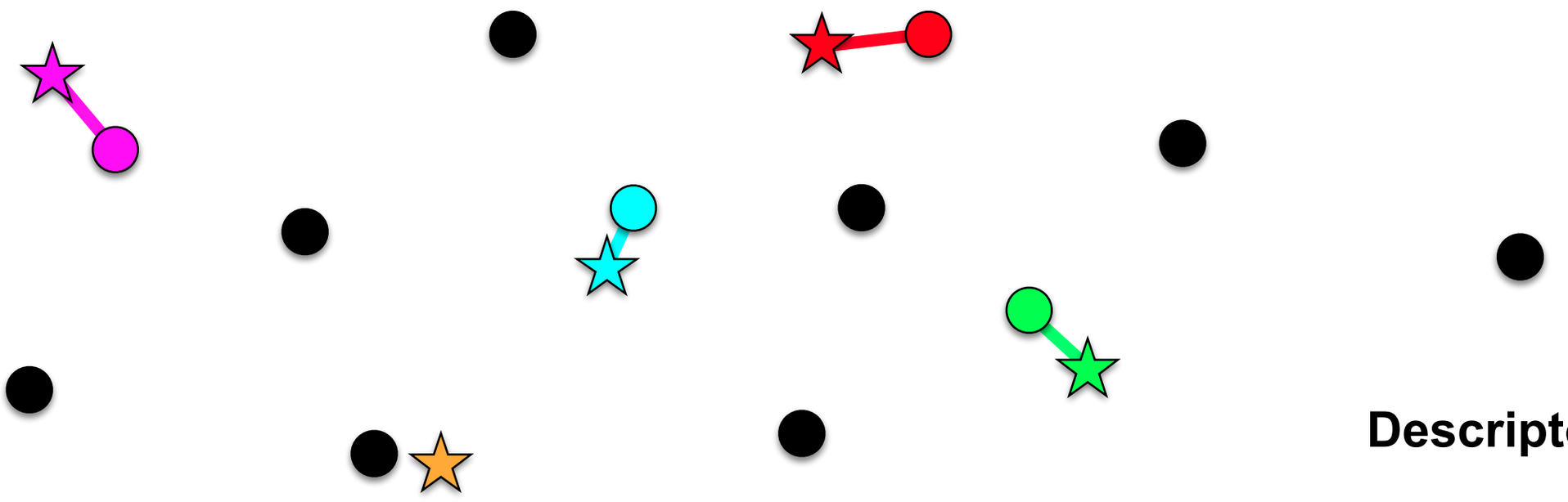
Matching = Nearest Neighbor Search



Query Image

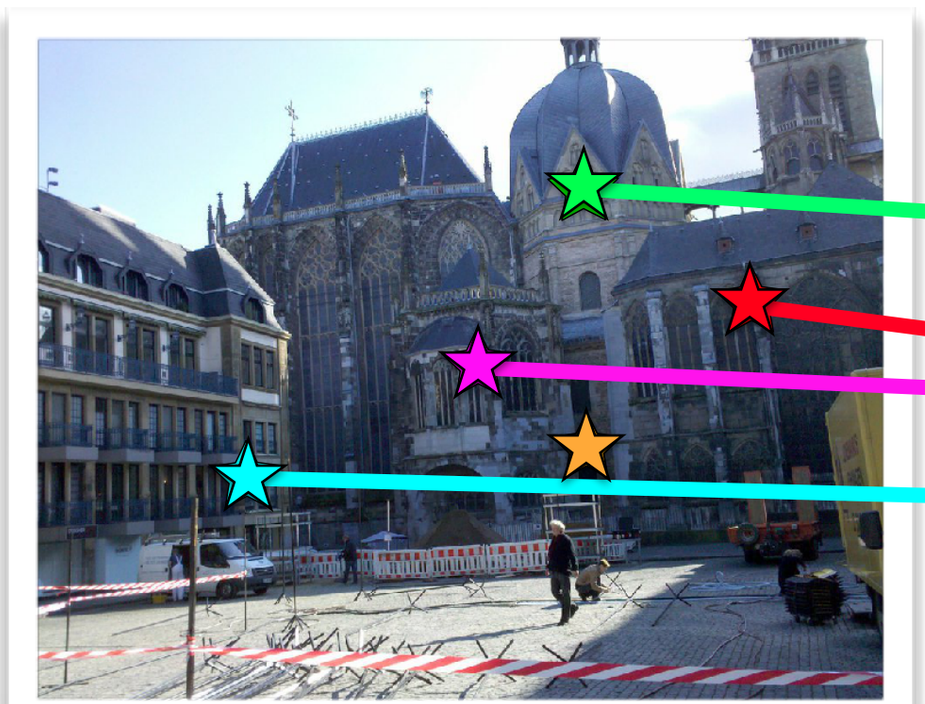


3D Model

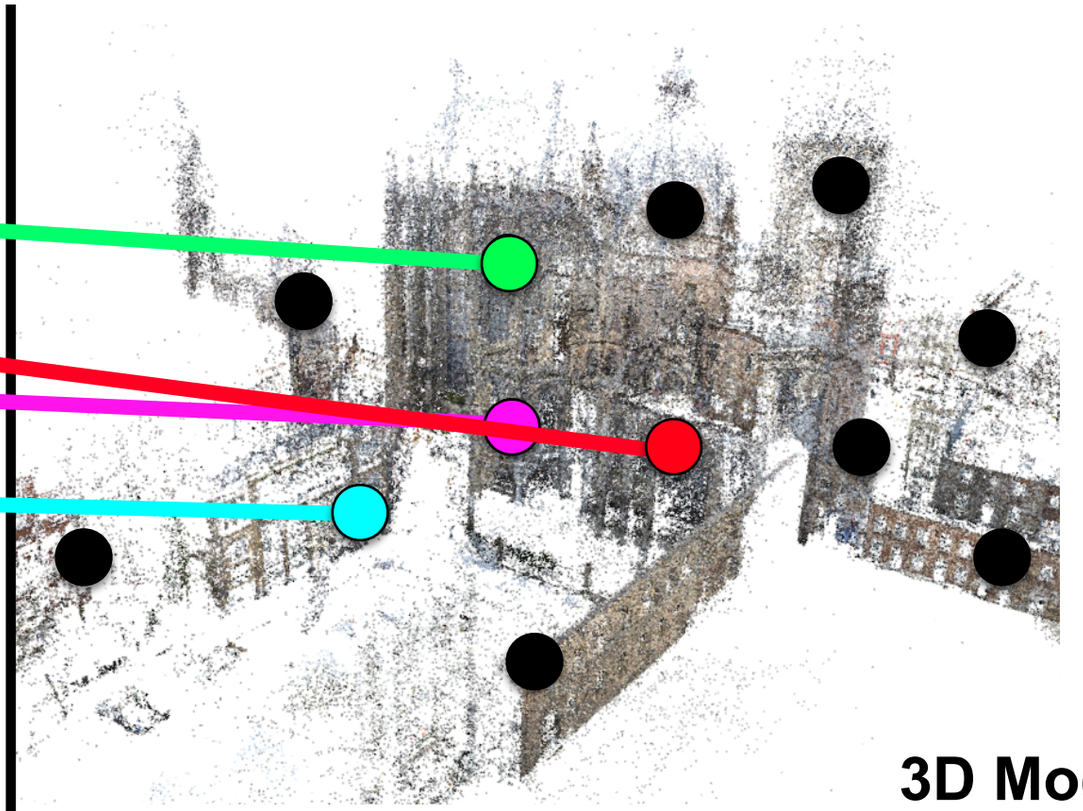


Descriptor Space

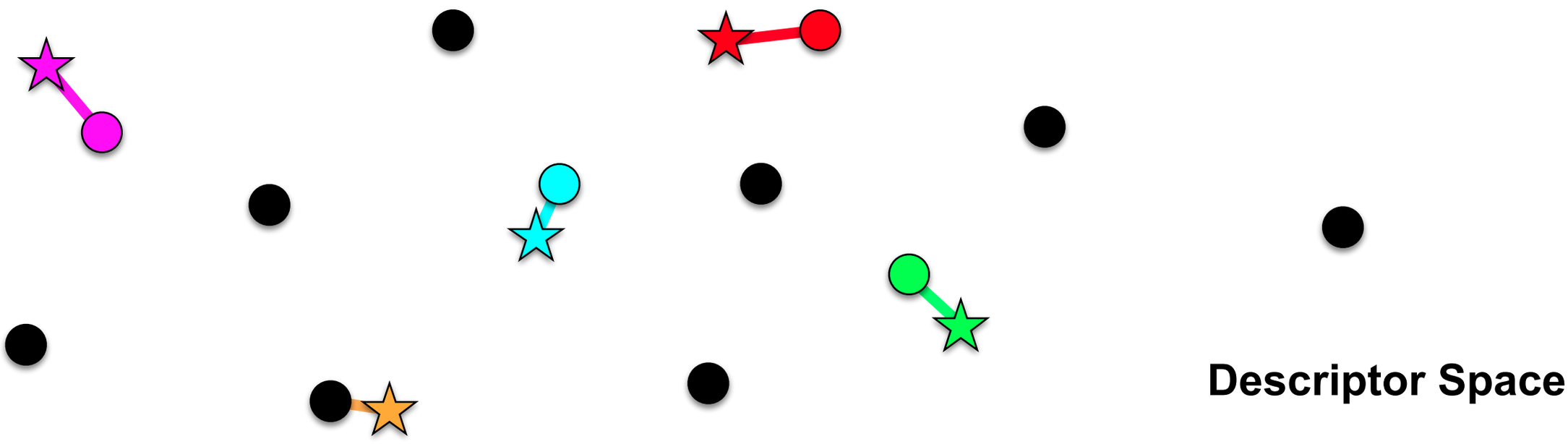
Matching = Nearest Neighbor Search



Query Image

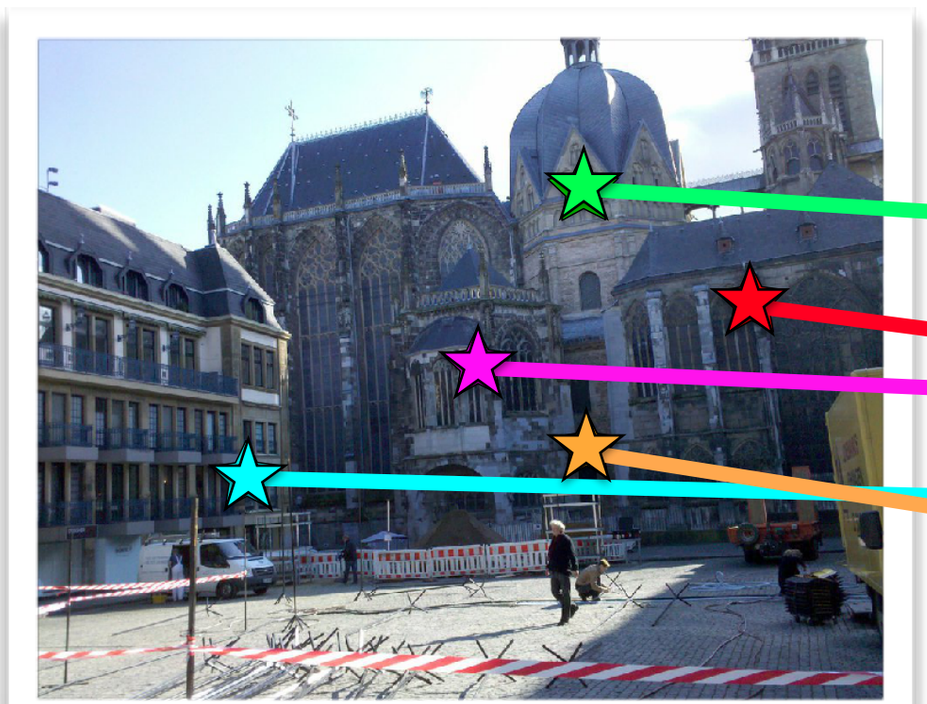


3D Model

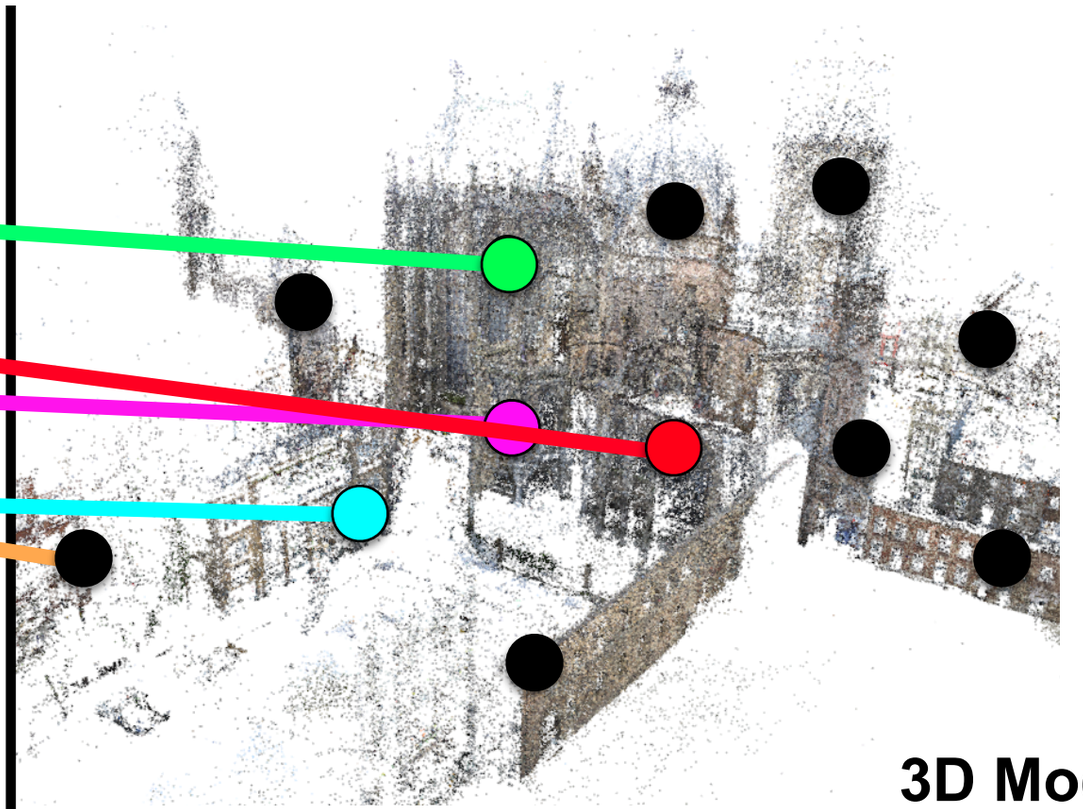


Descriptor Space

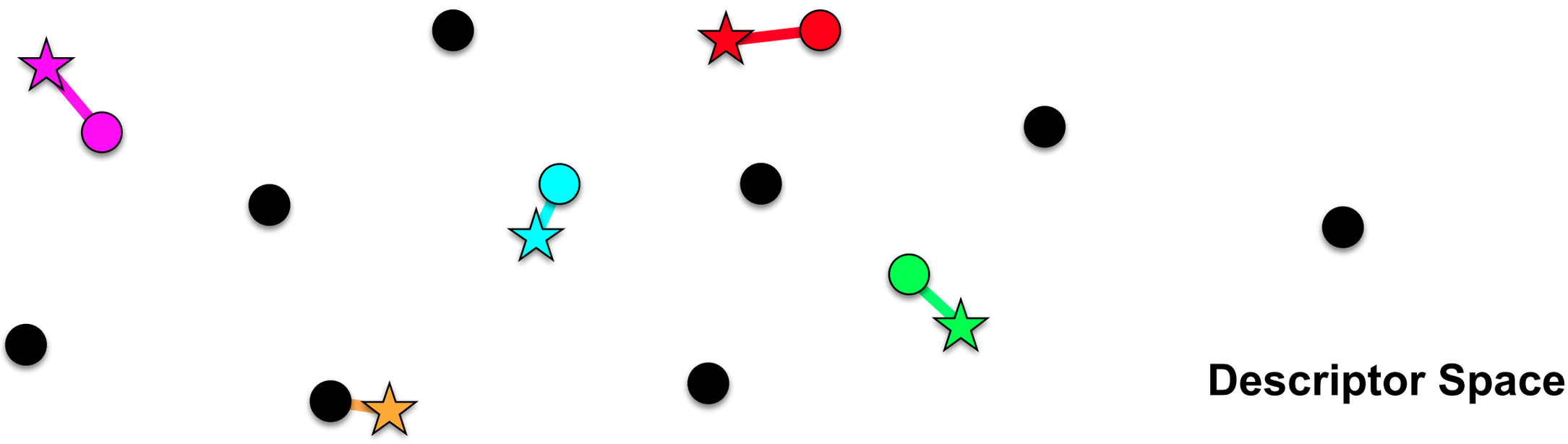
Matching = Nearest Neighbor Search



Query Image

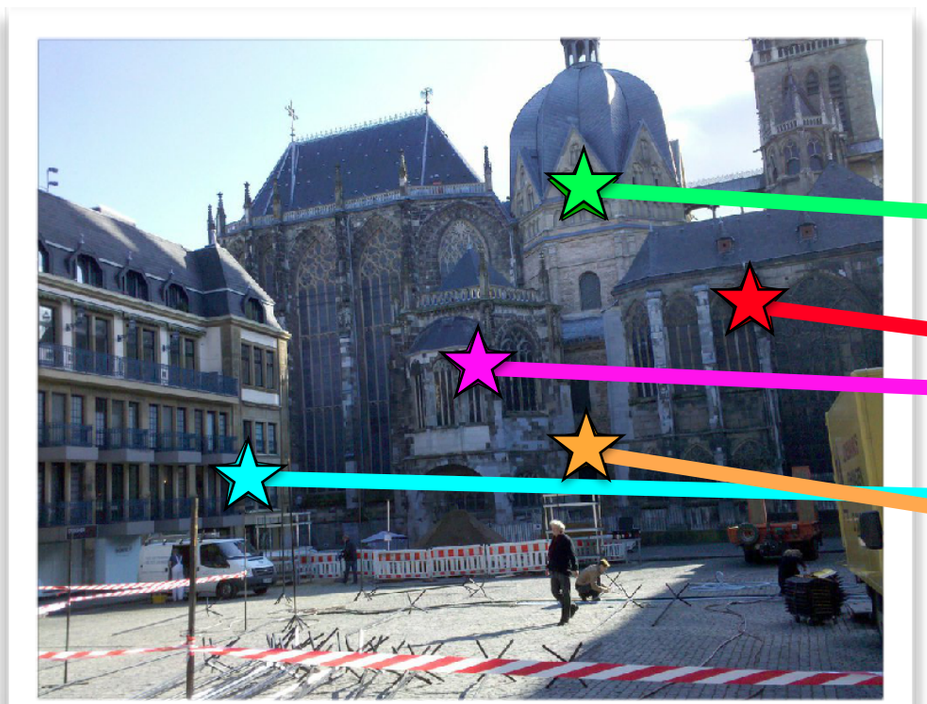


3D Model

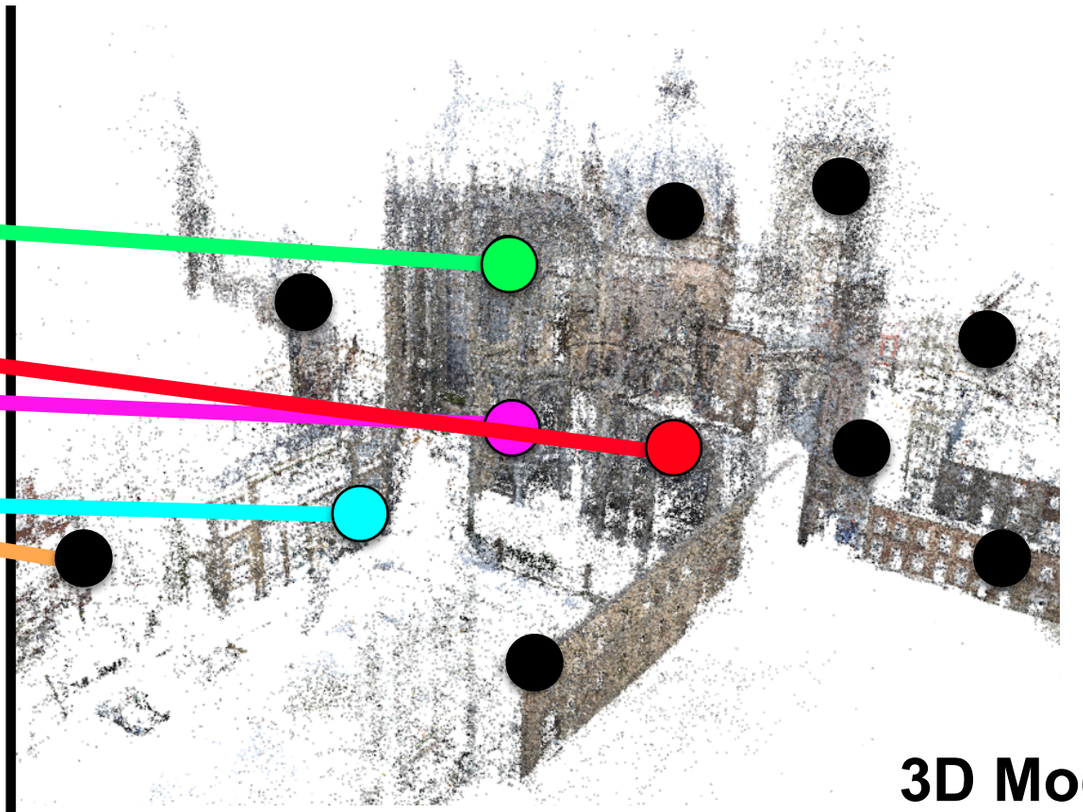


Descriptor Space

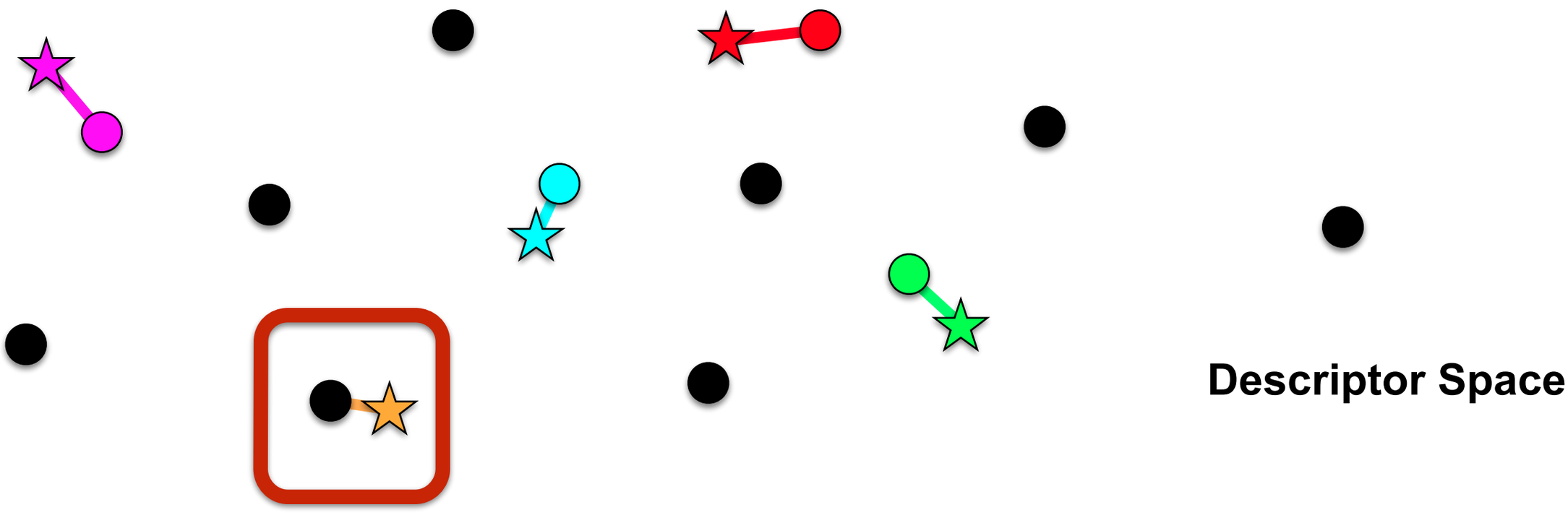
Matching = Nearest Neighbor Search



Query Image

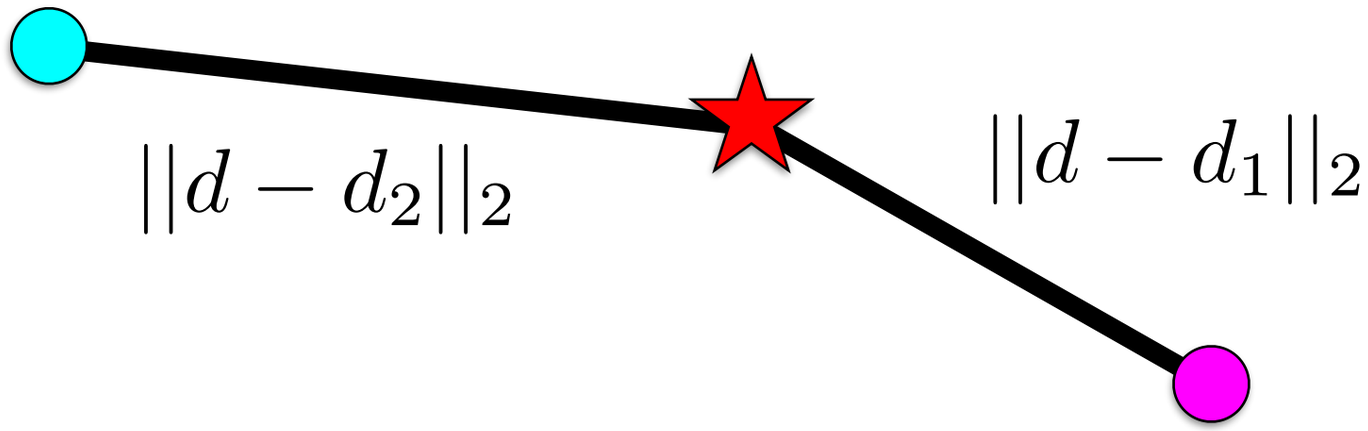


3D Model



Descriptor Space

Lowe's Ratio Test



- No every nearest neighbor is correct
- Use ratio test to reject wrong / ambiguous matches [Lowe, IJCV'04]
- Only accept match if

$$\frac{||d - d_1||_2}{||d - d_2||_2} < 0.8$$

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

- Multiple fast approximate nearest neighbor search methods:

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

- Multiple fast approximate nearest neighbor search methods:
  - **kd-trees** [Muja & Lowe, PAMI'14] [code]

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

- Multiple fast approximate nearest neighbor search methods:
  - **kd-trees** [Muja & Lowe, PAMI'14] [code]
  - **Hierarchical k-means trees** [Muja & Lowe, PAMI'14] [code]

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

- Multiple fast approximate nearest neighbor search methods:
  - **kd-trees** [Muja & Lowe, PAMI'14] [code]
  - **Hierarchical k-means trees** [Muja & Lowe, PAMI'14] [code]
  - Product quantization [Jégou et al., PAMI'11] [code] [Kalantidis & Avrithis, CVPR'14]

Nearest Neighbor Search

- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

- Multiple fast approximate nearest neighbor search methods:
  - **kd-trees** [Muja & Lowe, PAMI'14] [code]
  - **Hierarchical k-means trees** [Muja & Lowe, PAMI'14] [code]
  - Product quantization [Jégou et al., PAMI'11] [code] [Kalantidis & Avrithis, CVPR'14]
  - Diverse hashing techniques

Nearest Neighbor Search

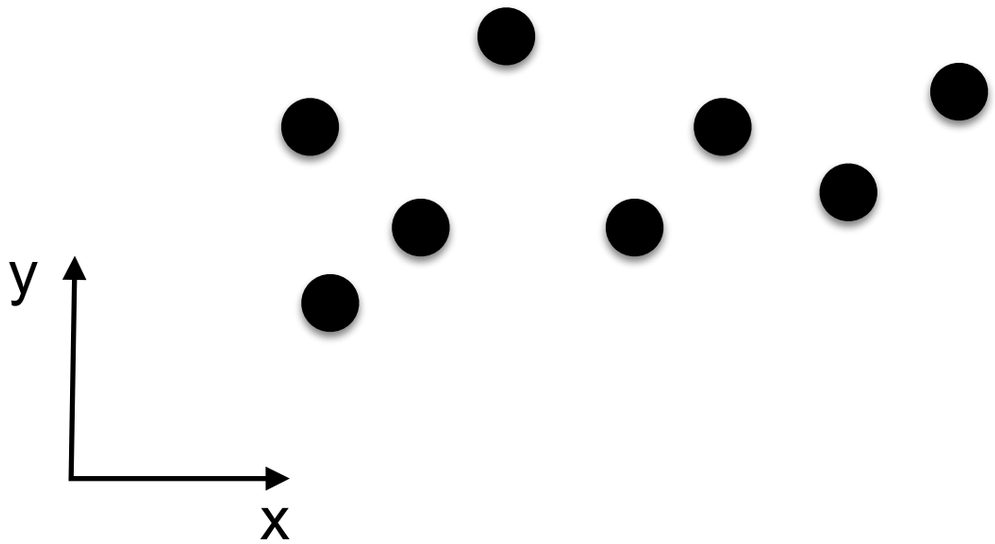
- Typical datasets: 3-10k features, >1M points

➡ Exhaustive (linear) nearest neighbor search is prohibitive

- **Curse of dimensionality**: No exact search method that is faster than linear search

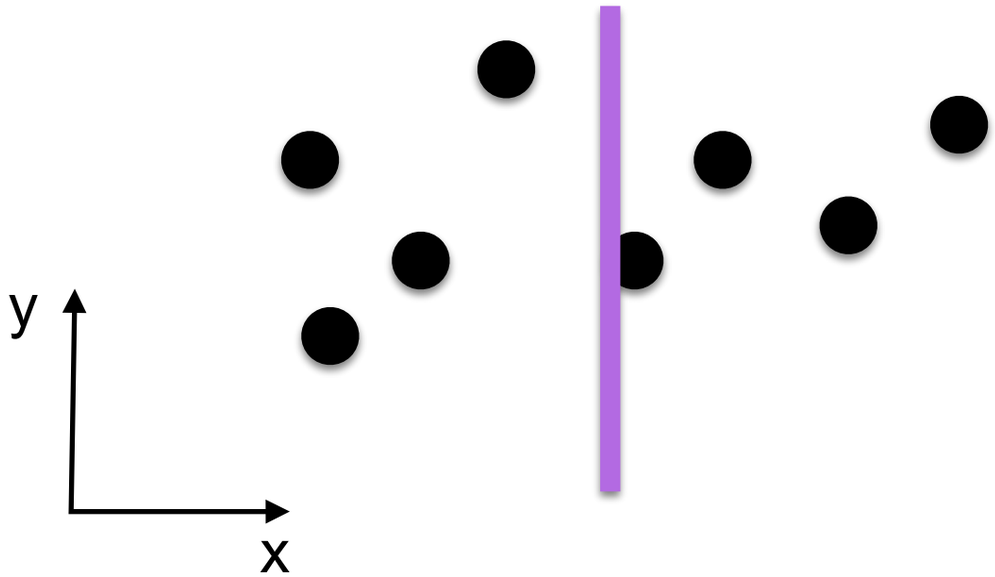
- Multiple fast approximate nearest neighbor search methods:
  - **kd-trees** [Muja & Lowe, PAMI'14] [code]
  - **Hierarchical k-means trees** [Muja & Lowe, PAMI'14] [code]
  - Product quantization [Jégou et al., PAMI'11] [code] [Kalantidis & Avrithis, CVPR'14]
  - Diverse hashing techniques
  - …

kd-tree Construction & Search



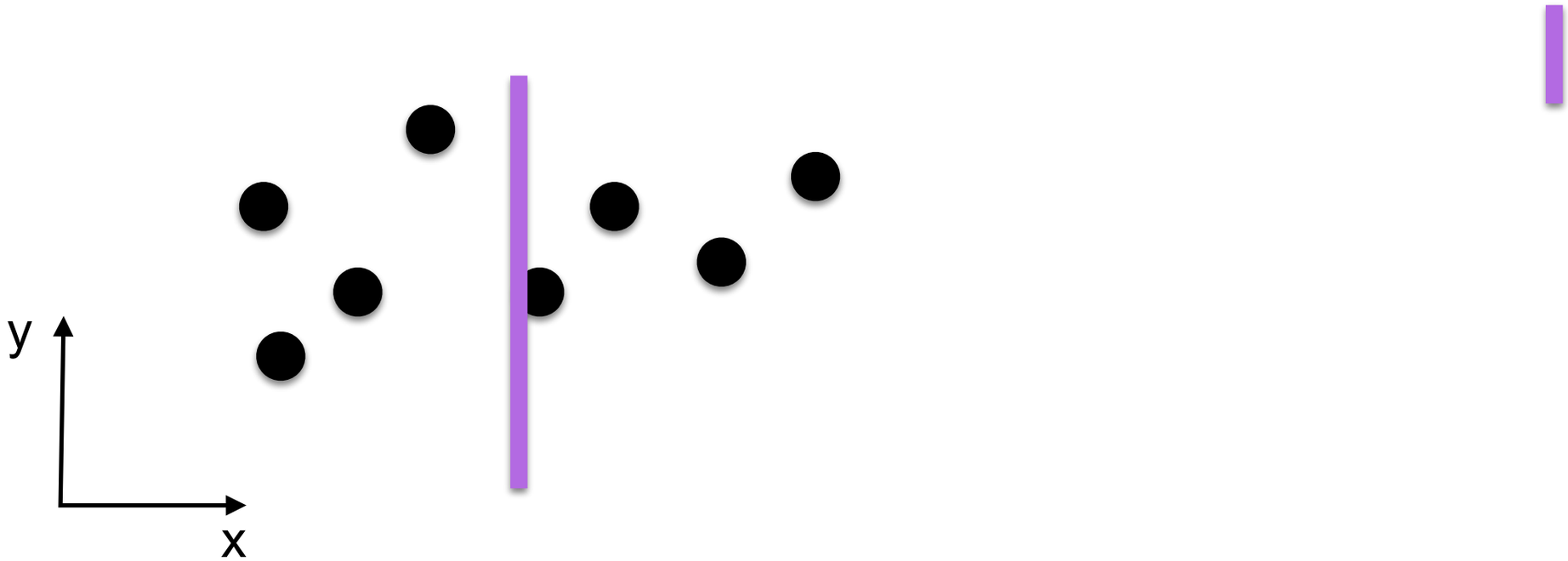
- Iteratively split dimension with largest variance at median

kd-tree Construction & Search



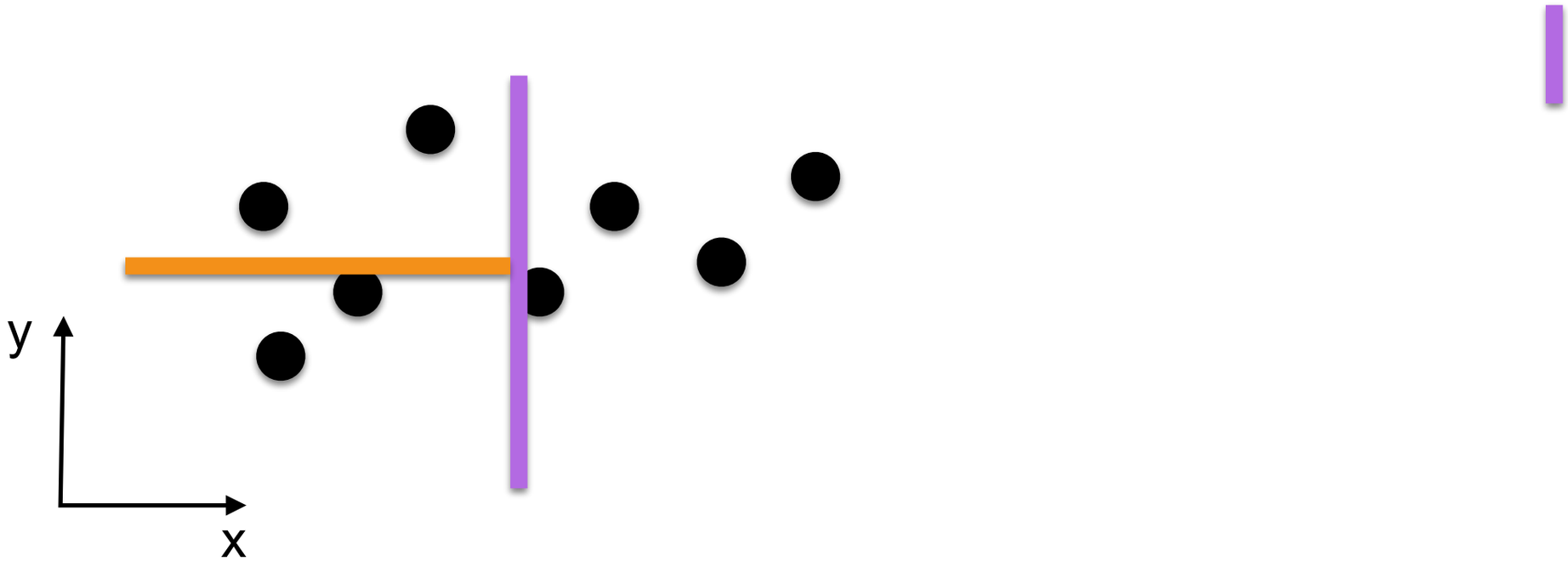
- Iteratively split dimension with largest variance at median

kd-tree Construction & Search



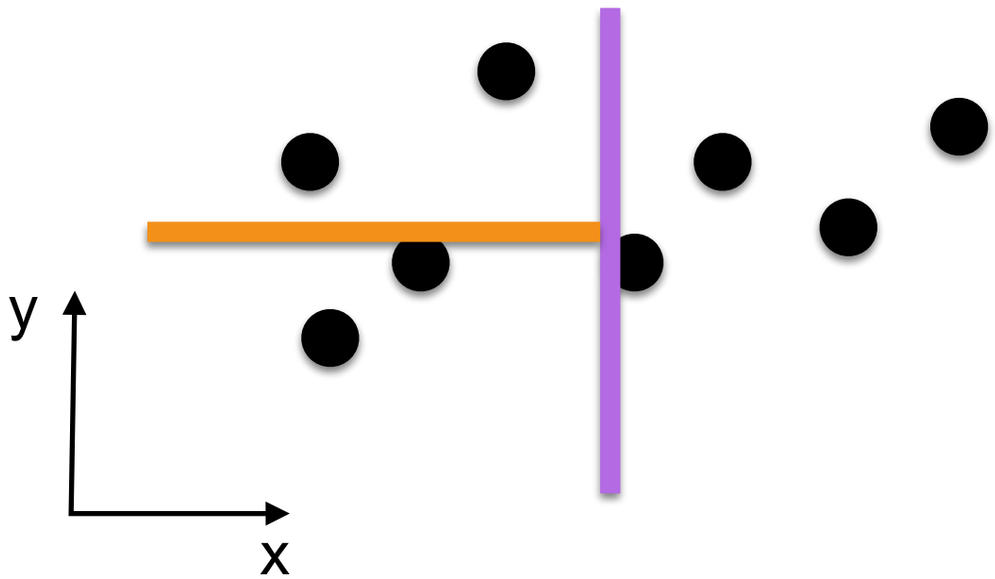
- Iteratively split dimension with largest variance at median

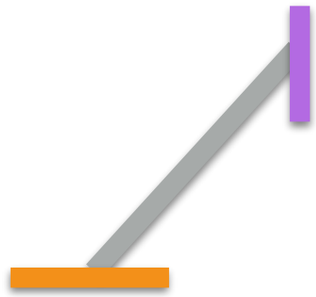
kd-tree Construction & Search



- Iteratively split dimension with largest variance at median

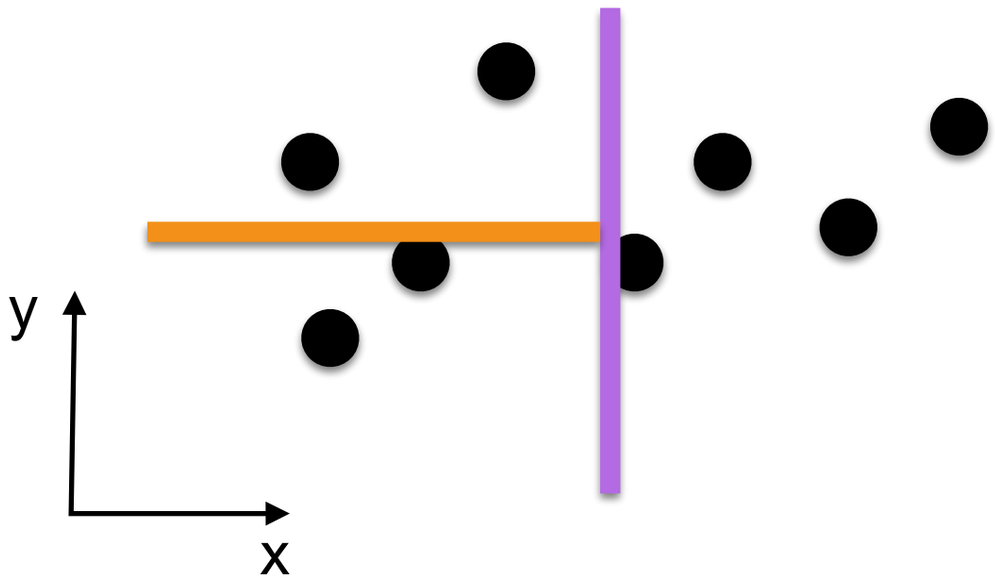
kd-tree Construction & Search

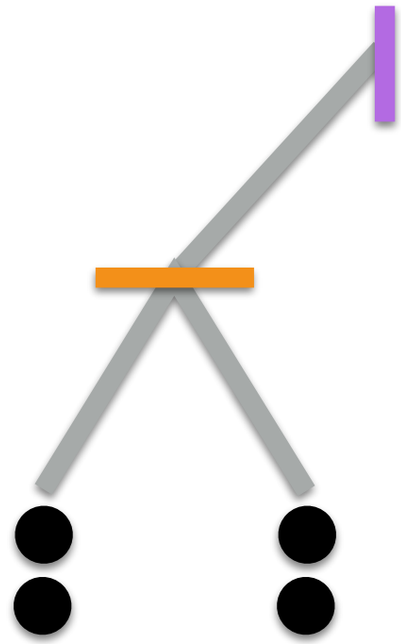




- Iteratively split dimension with largest variance at median

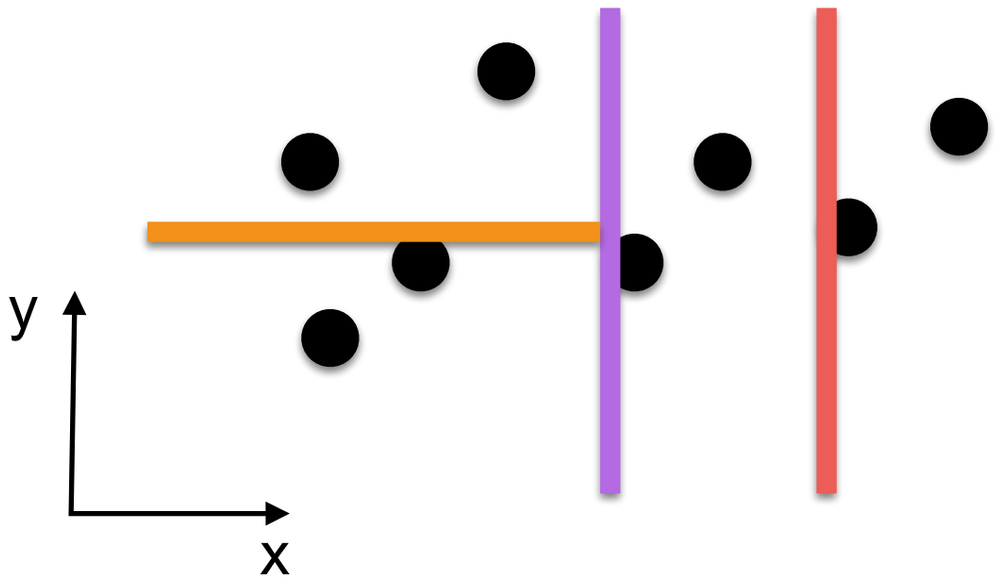
kd-tree Construction & Search

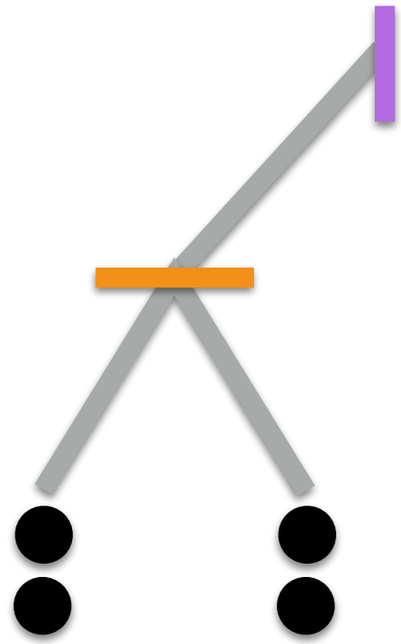




- Iteratively split dimension with largest variance at median

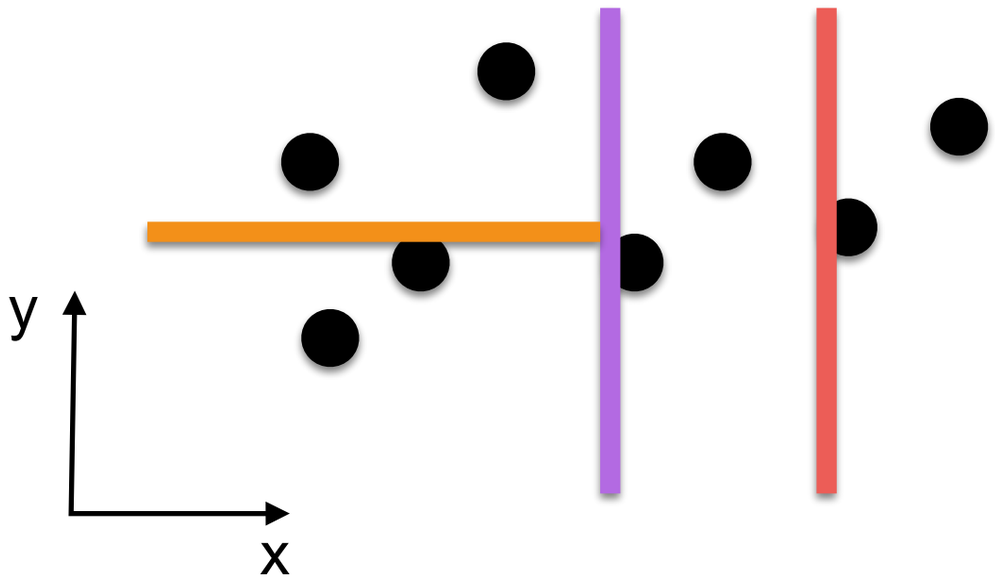
kd-tree Construction & Search

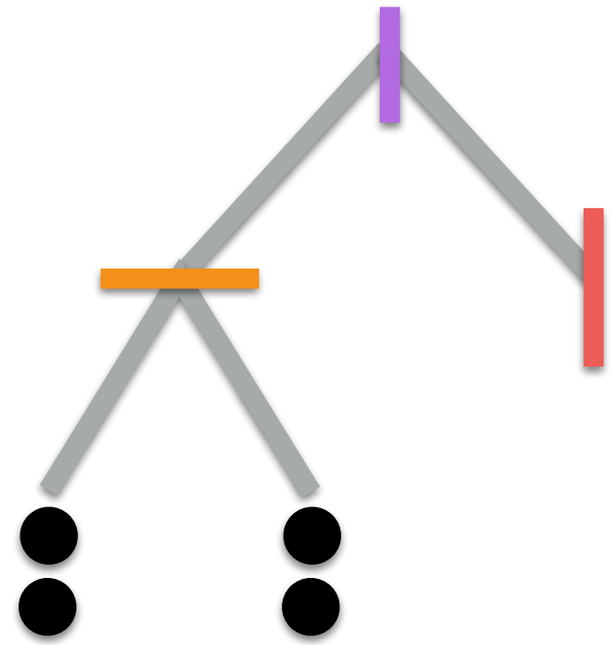




- Iteratively split dimension with largest variance at median

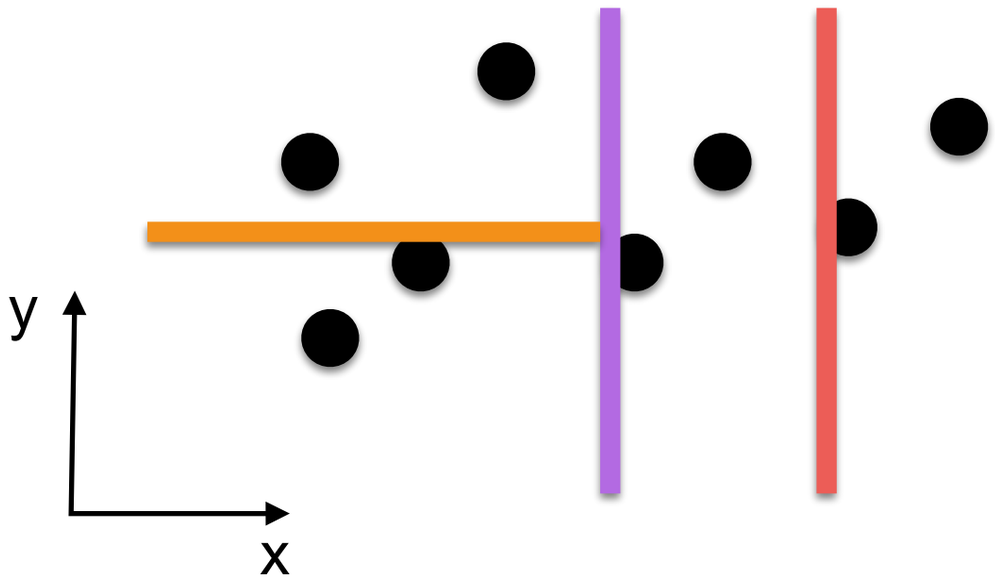
kd-tree Construction & Search

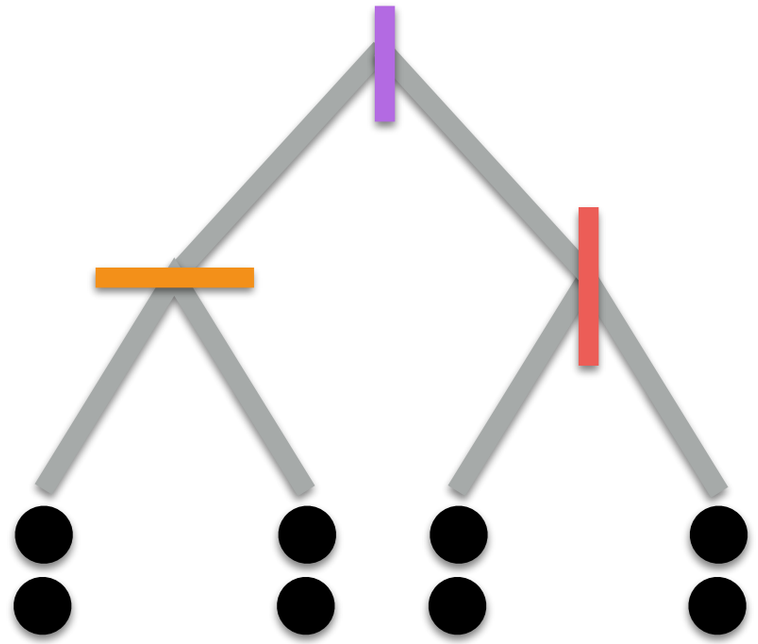




- Iteratively split dimension with largest variance at median

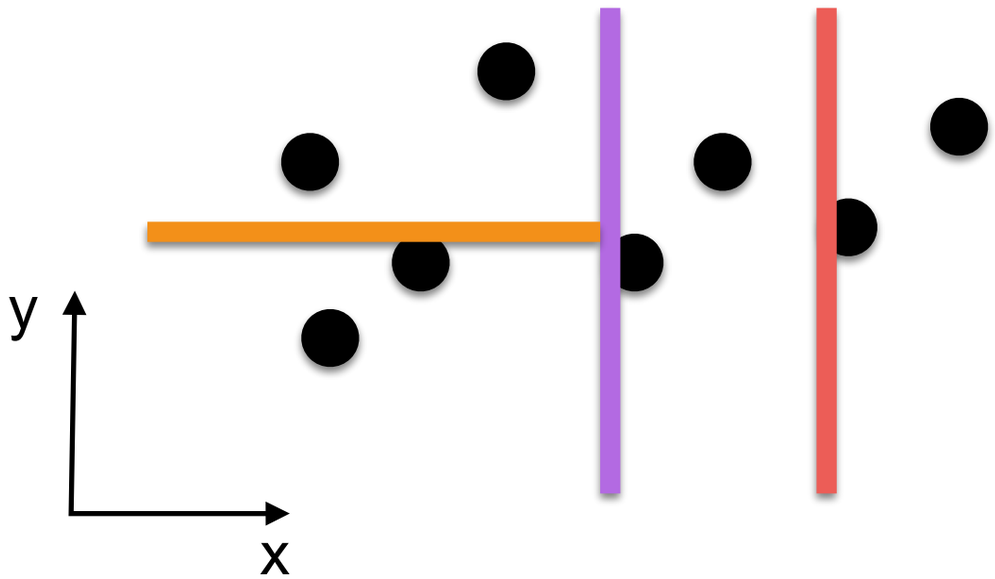
kd-tree Construction & Search

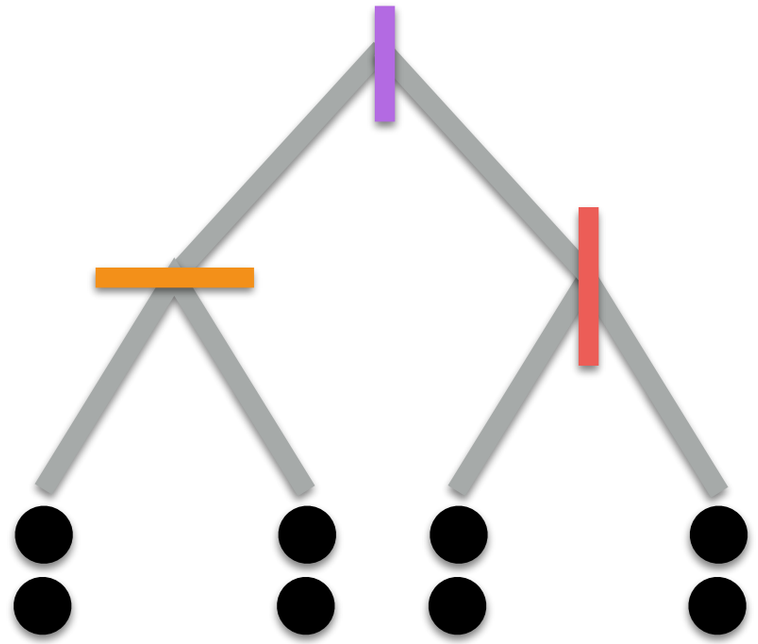




- Iteratively split dimension with largest variance at median

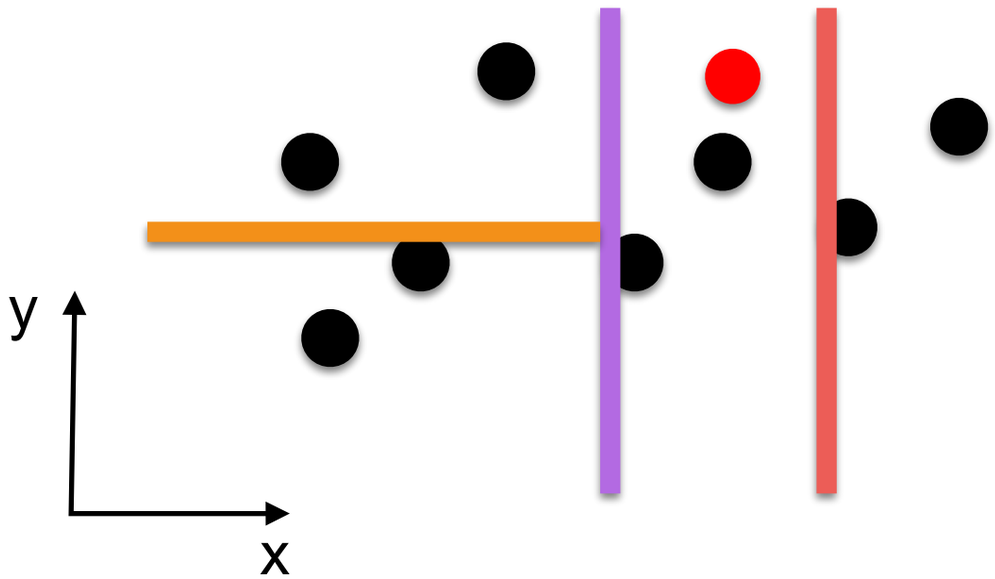
kd-tree Construction & Search

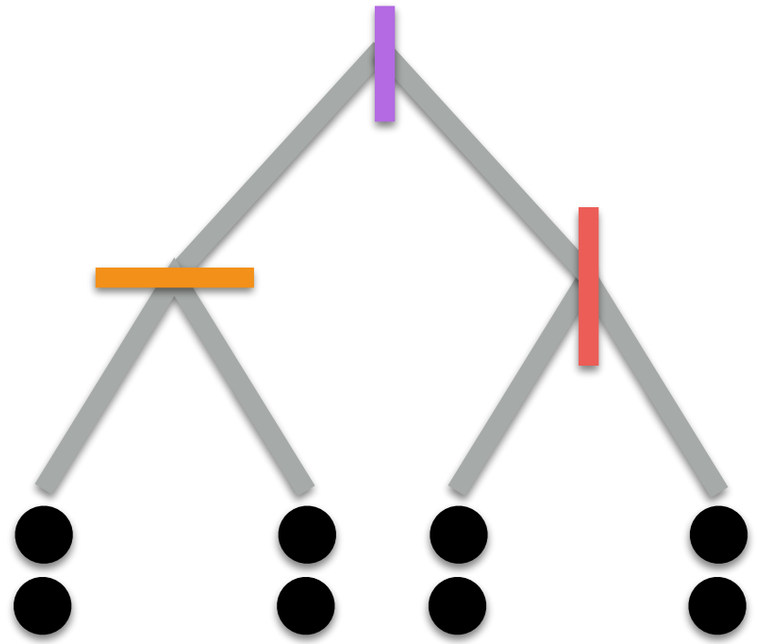




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

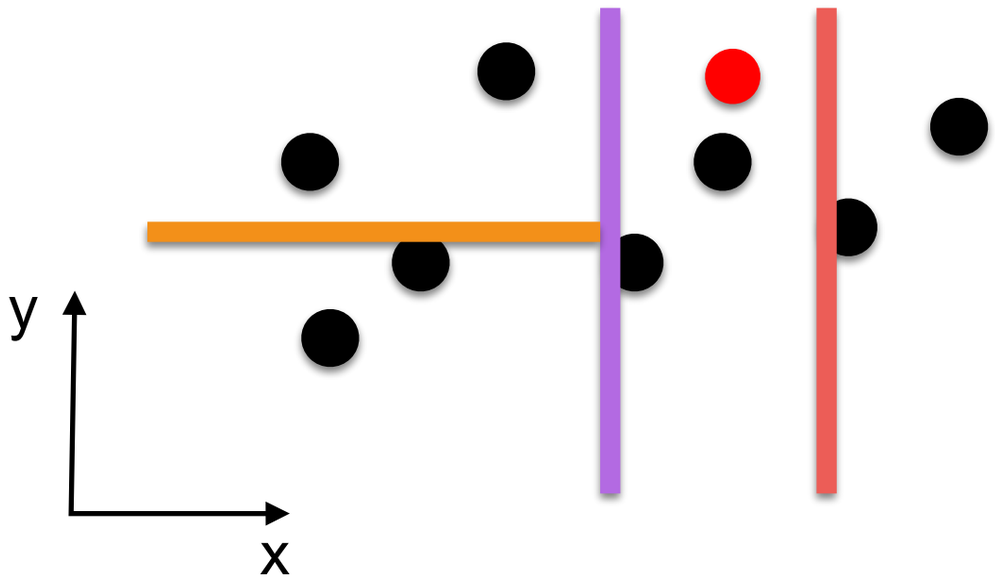
kd-tree Construction & Search

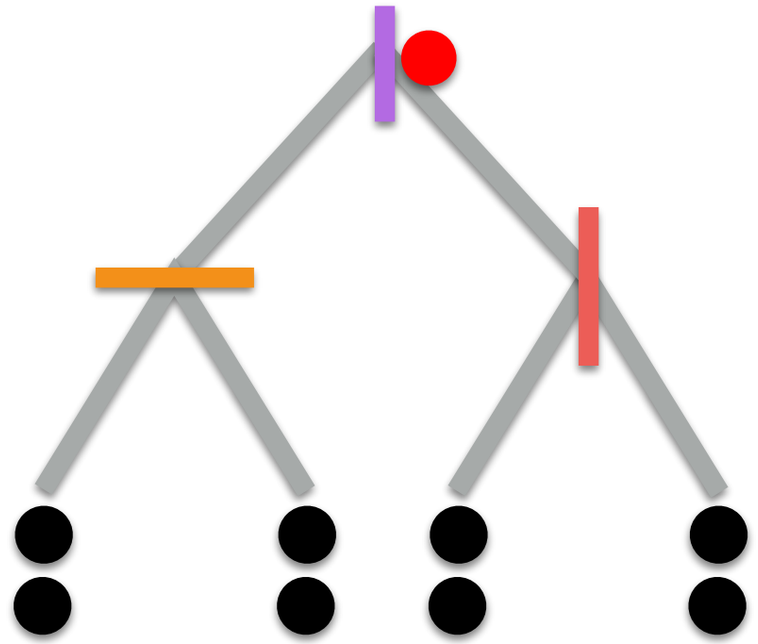




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

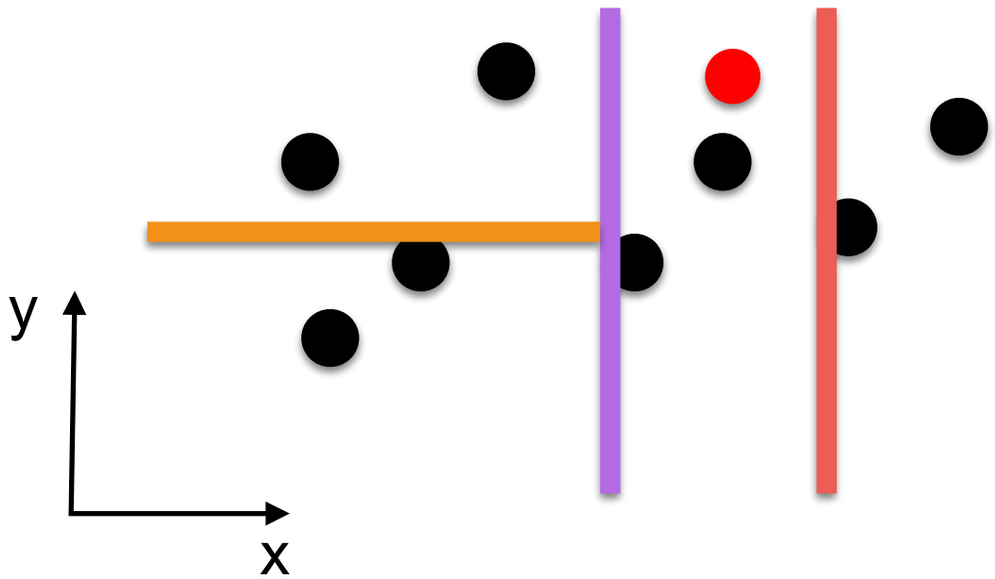
kd-tree Construction & Search

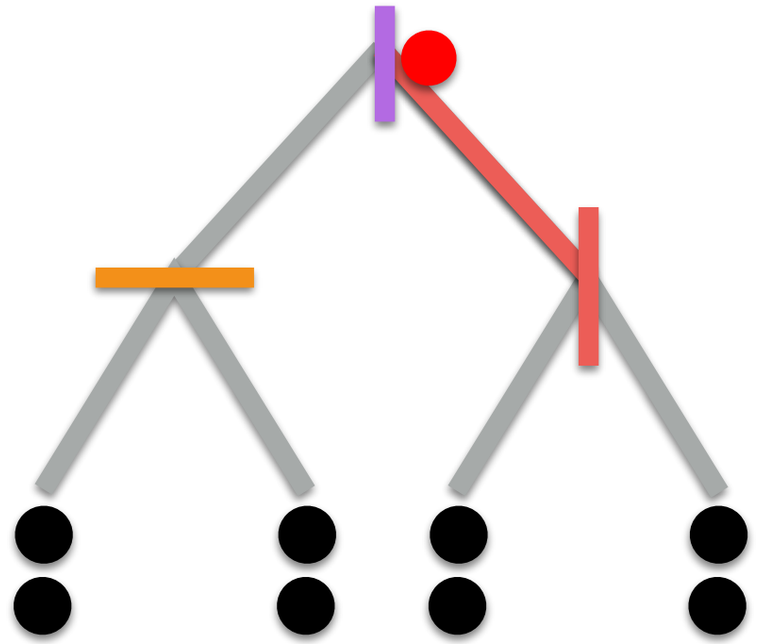




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

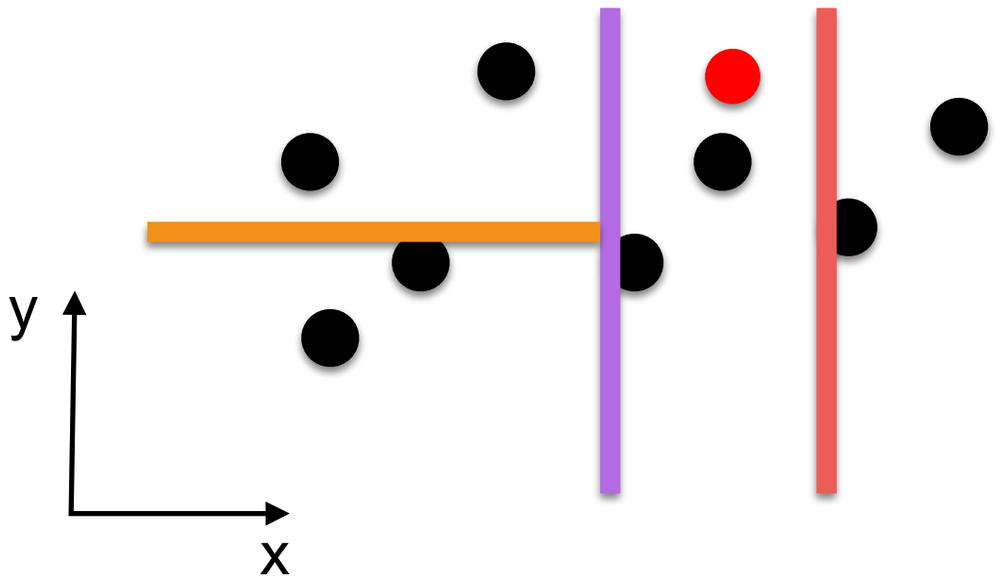
kd-tree Construction & Search

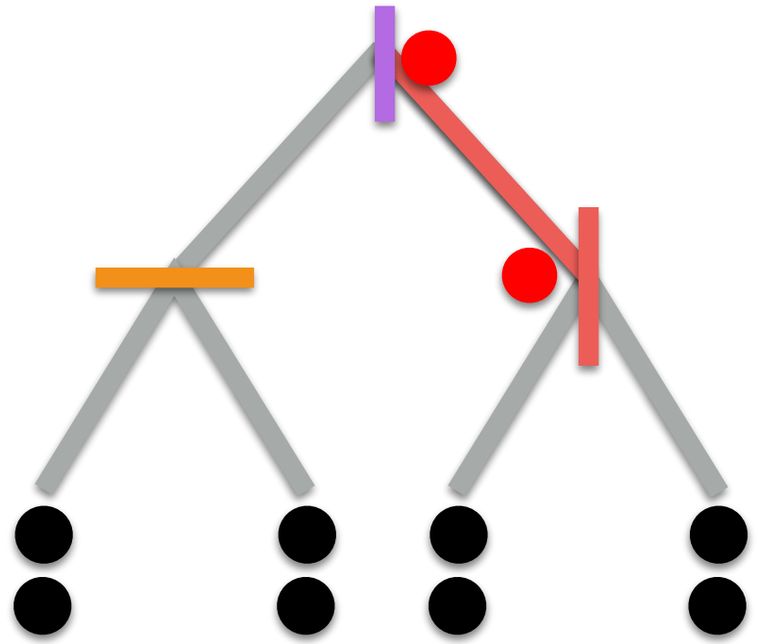




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

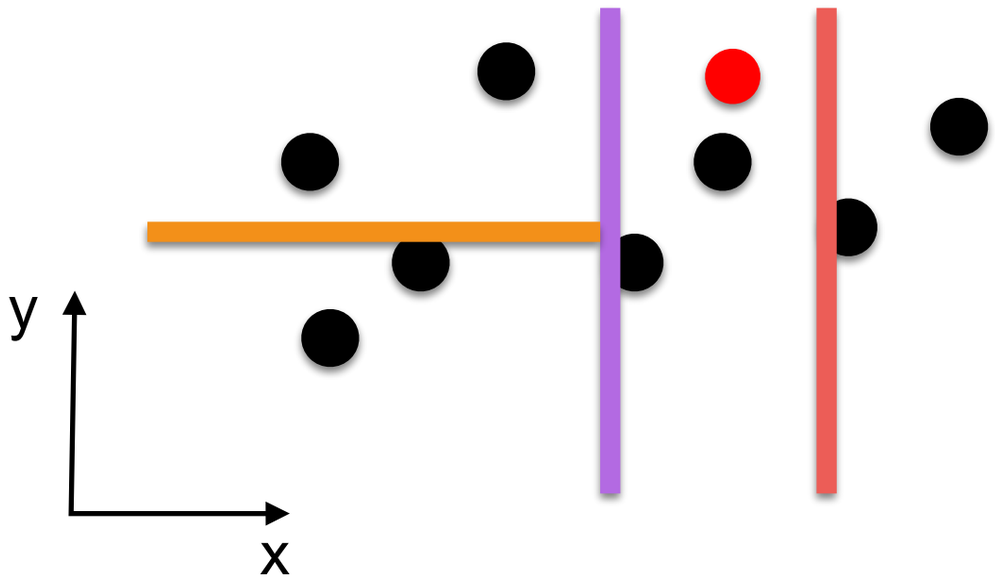
kd-tree Construction & Search

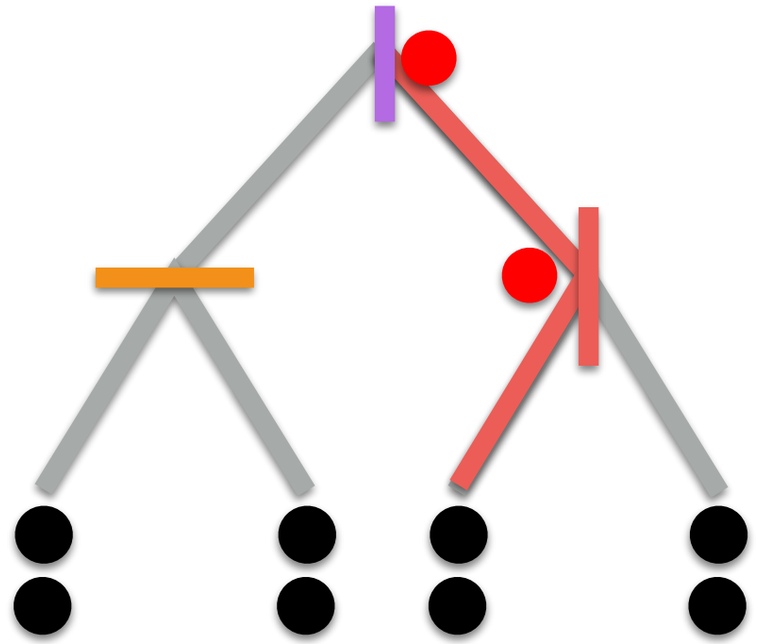




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

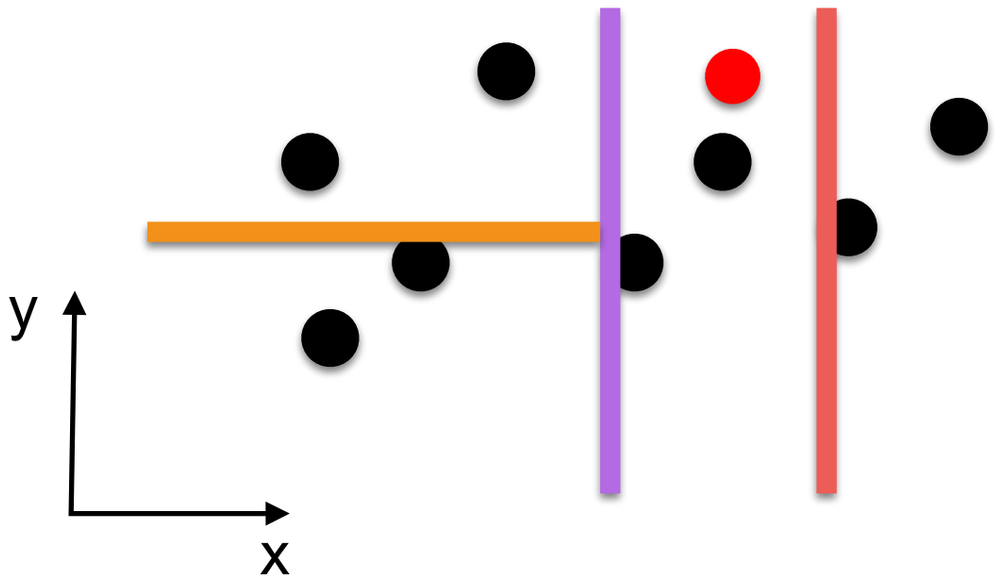
kd-tree Construction & Search

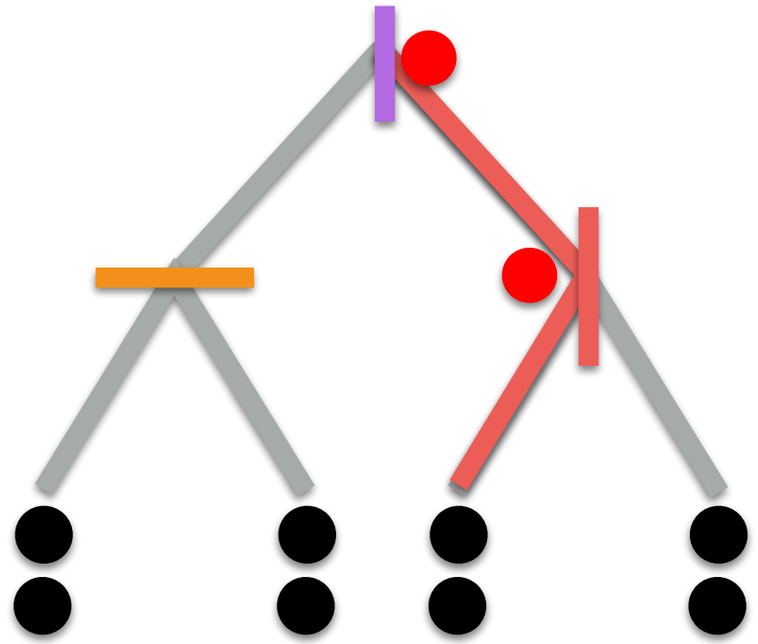




- Iteratively split dimension with largest variance at median
- Traversal based on side of split

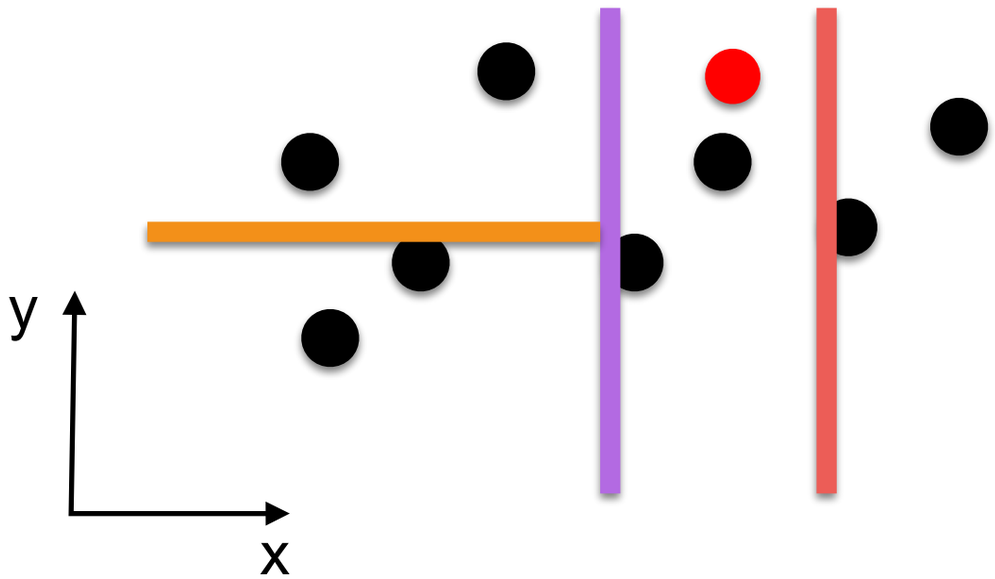
kd-tree Construction & Search

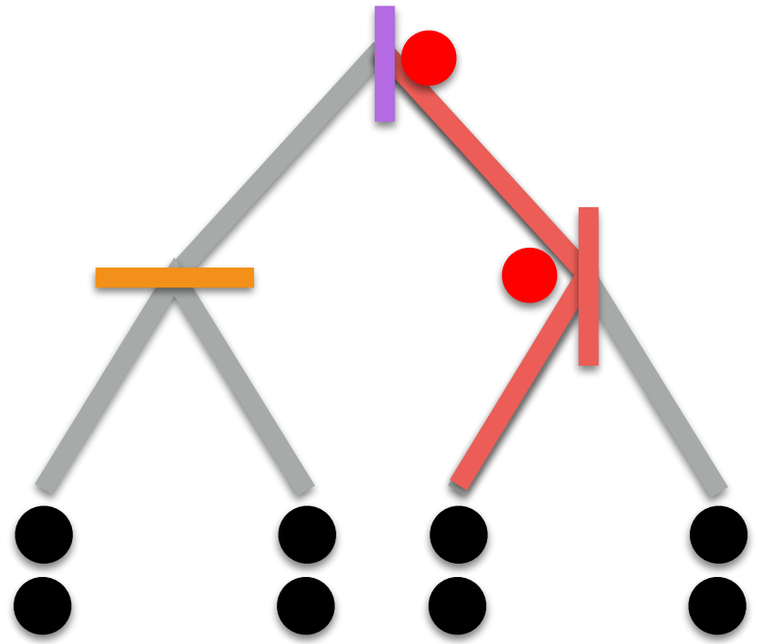




- Iteratively split dimension with largest variance at median
- Traversal based on side of split
- 1B SIFT descriptors (128D) → Only 30 dimensions considered!

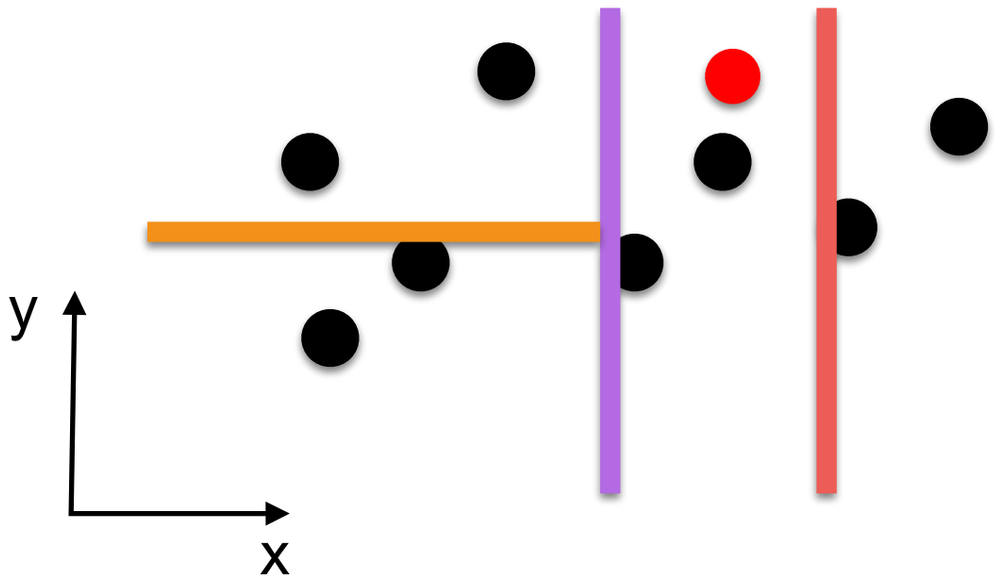
kd-tree Construction & Search

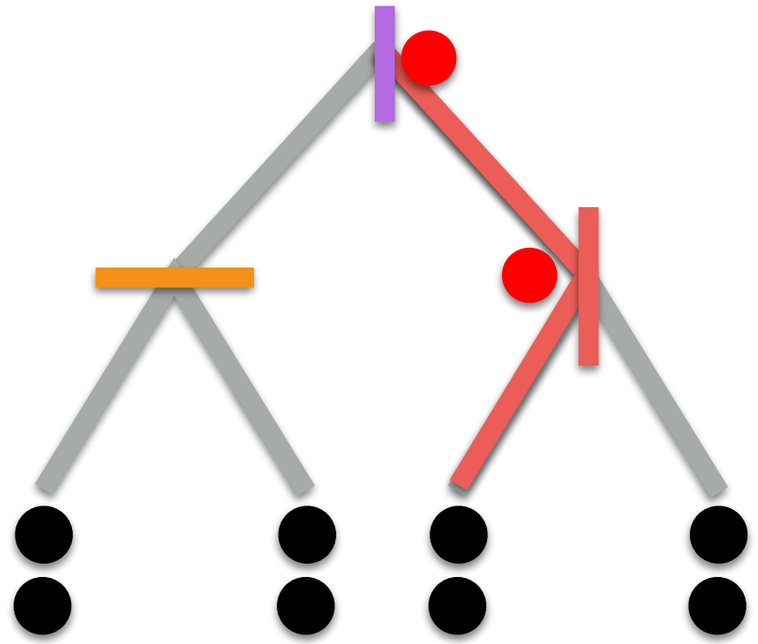




- Iteratively split dimension with largest variance at median
- Traversal based on side of split
- 1B SIFT descriptors (128D) → Only 30 dimensions considered!
  - Curse of dimensionality: Need to visit all leaves!

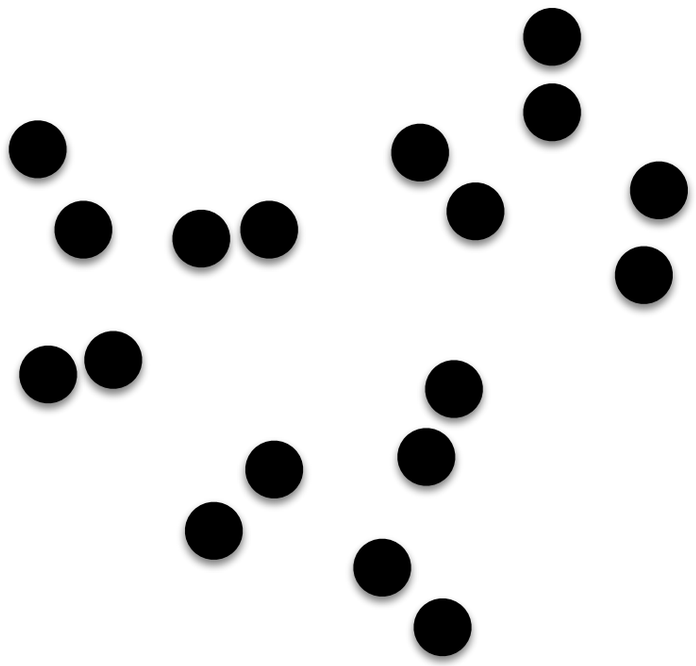
kd-tree Construction & Search





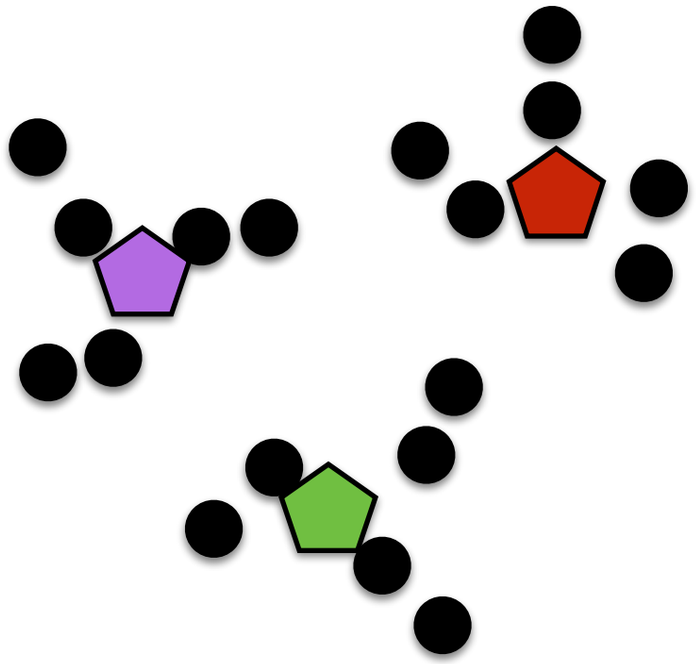
- Iteratively split dimension with largest variance at median
- Traversal based on side of split
- 1B SIFT descriptors (128D) → Only 30 dimensions considered!
  - Curse of dimensionality: Need to visit all leaves!
  - Approximate search: Visit `N` leaf nodes

Hierarchical k-Means Tree



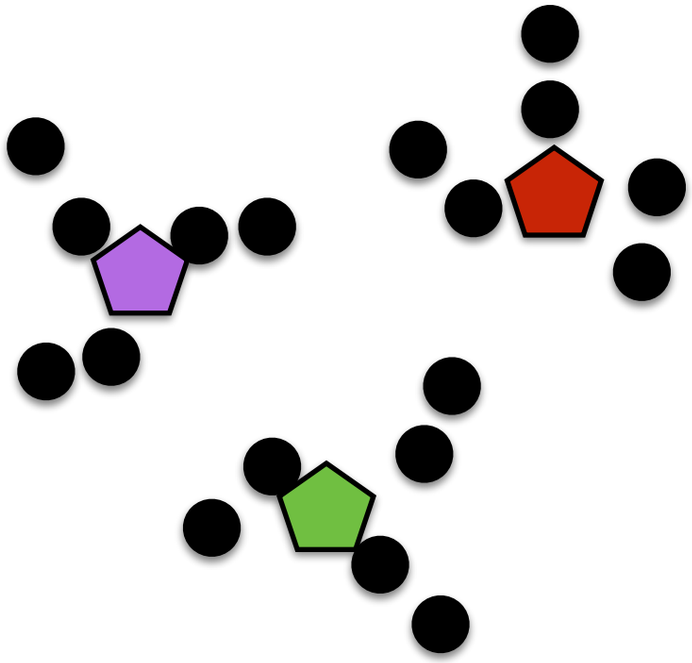
- Iteratively apply k-means clustering

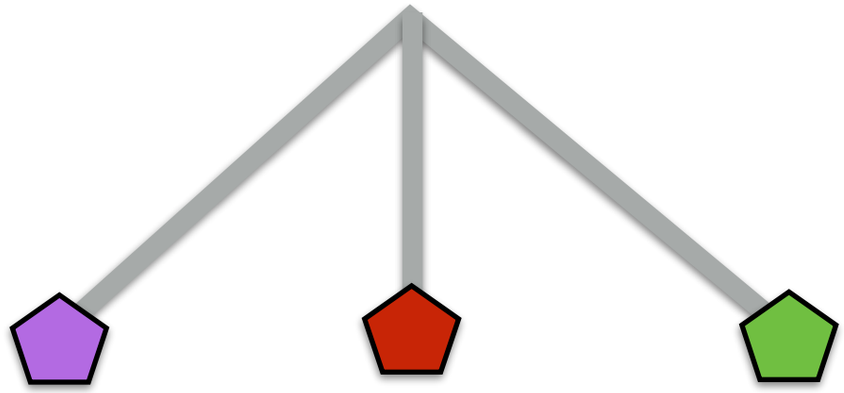
Hierarchical k-Means Tree



- Iteratively apply k-means clustering

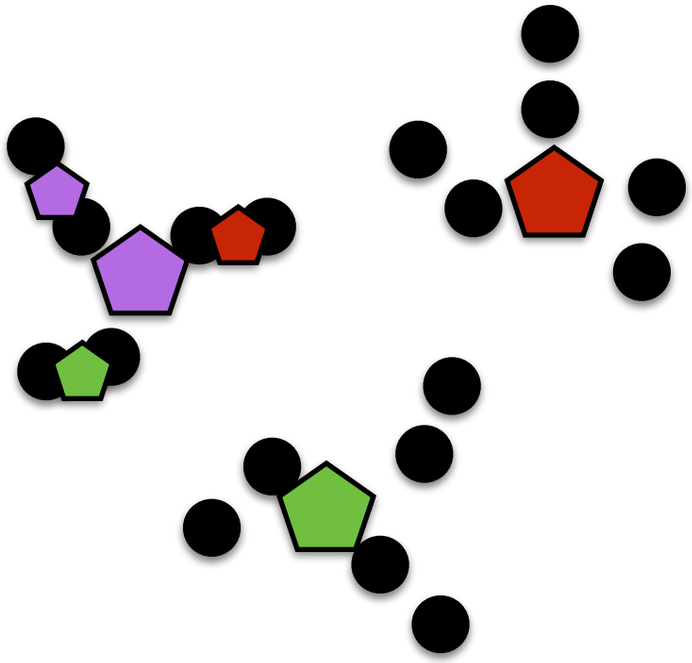
Hierarchical k-Means Tree

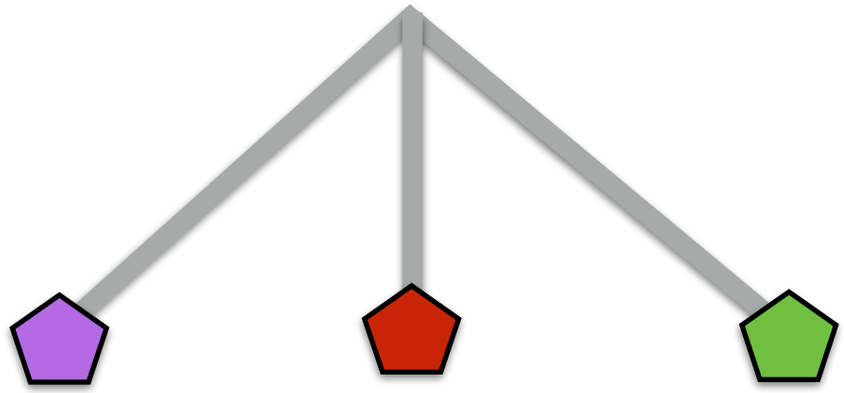




- Iteratively apply k-means clustering

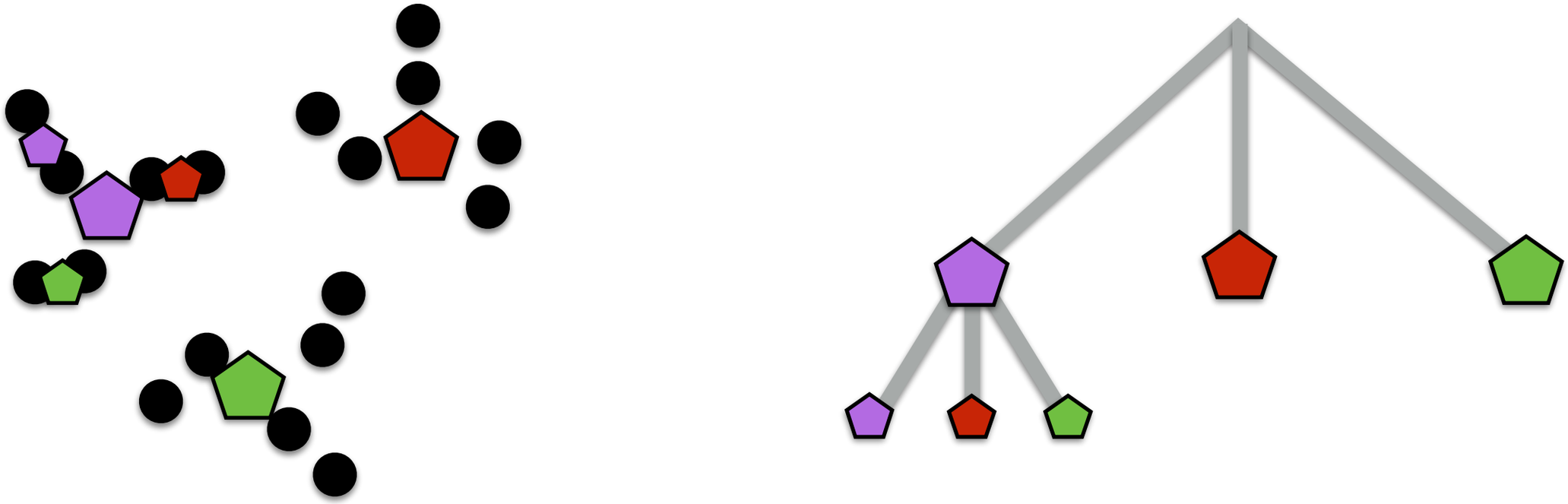
Hierarchical k-Means Tree





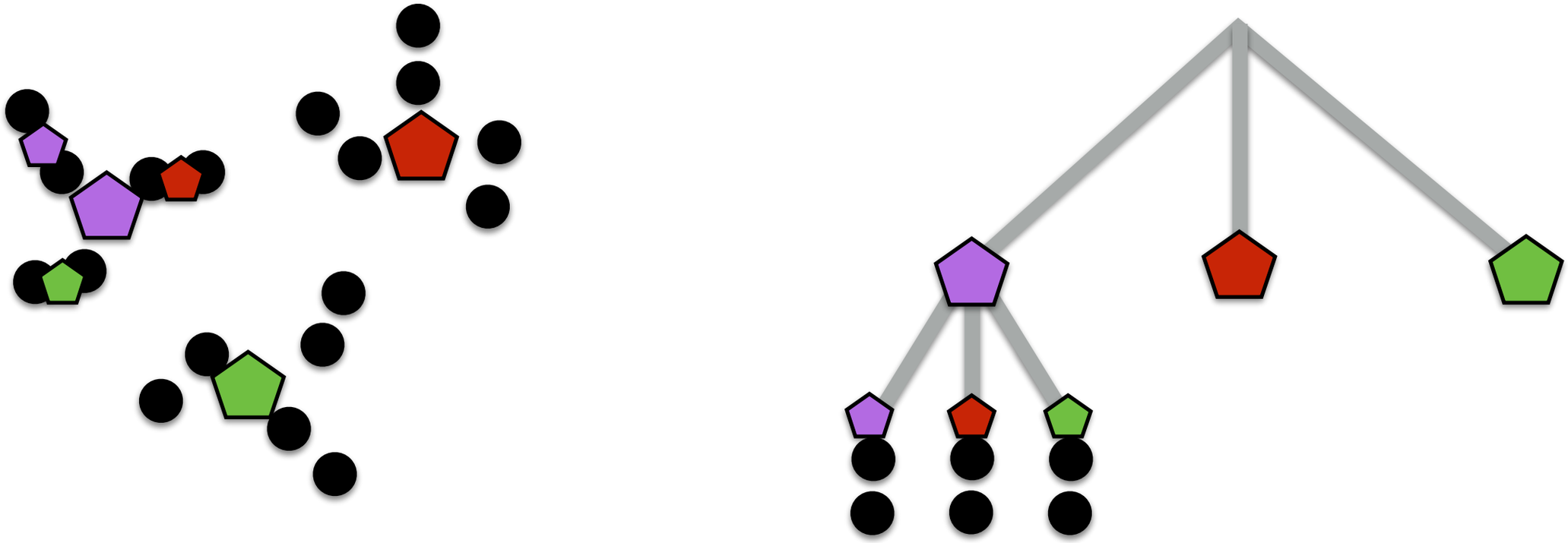
- Iteratively apply k-means clustering

Hierarchical k-Means Tree



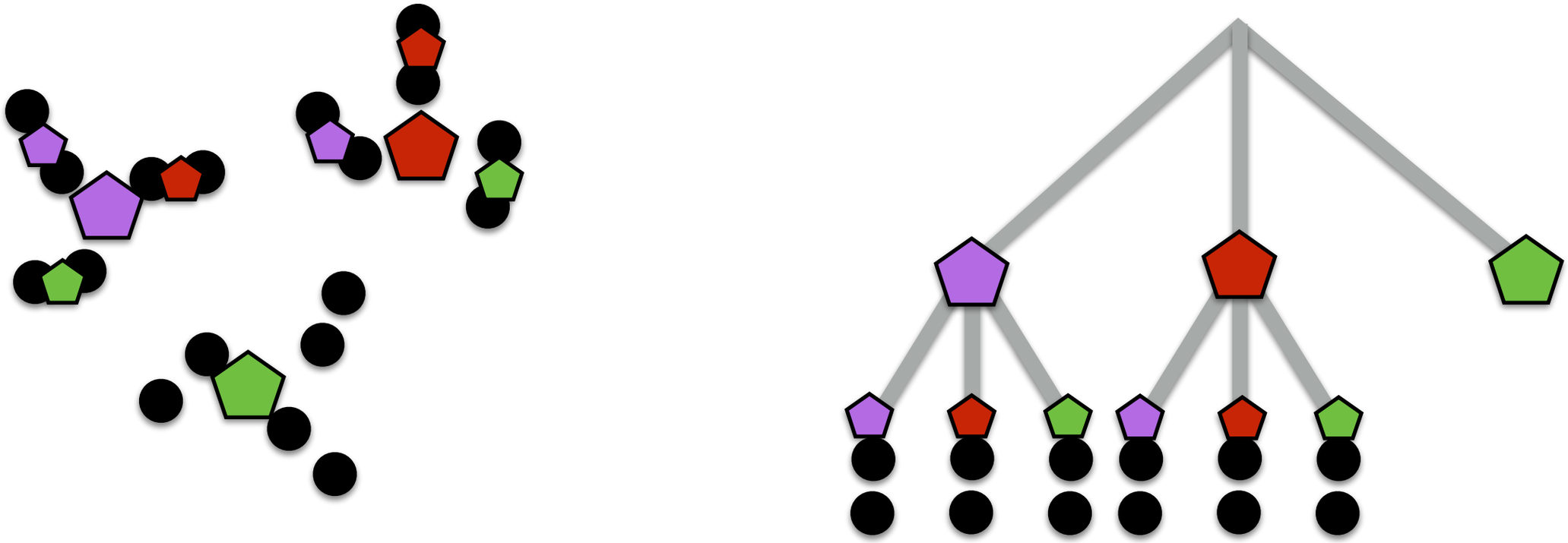
- Iteratively apply k-means clustering

Hierarchical k-Means Tree



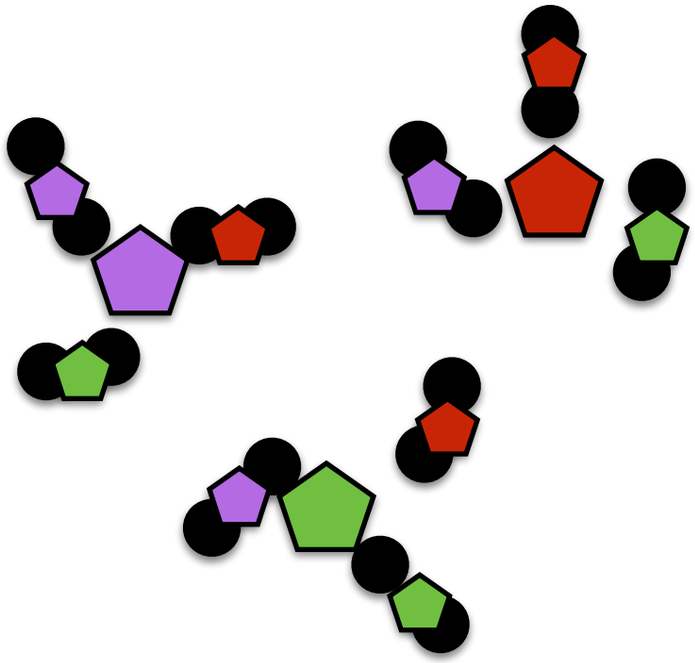
- Iteratively apply k-means clustering

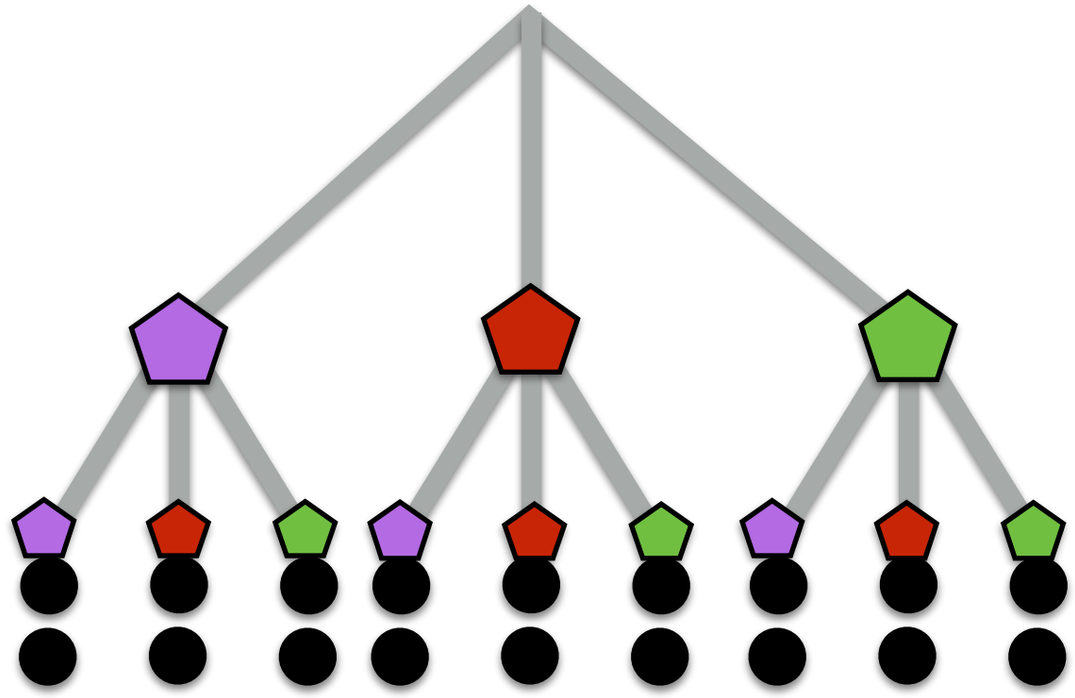
Hierarchical k-Means Tree



- Iteratively apply k-means clustering

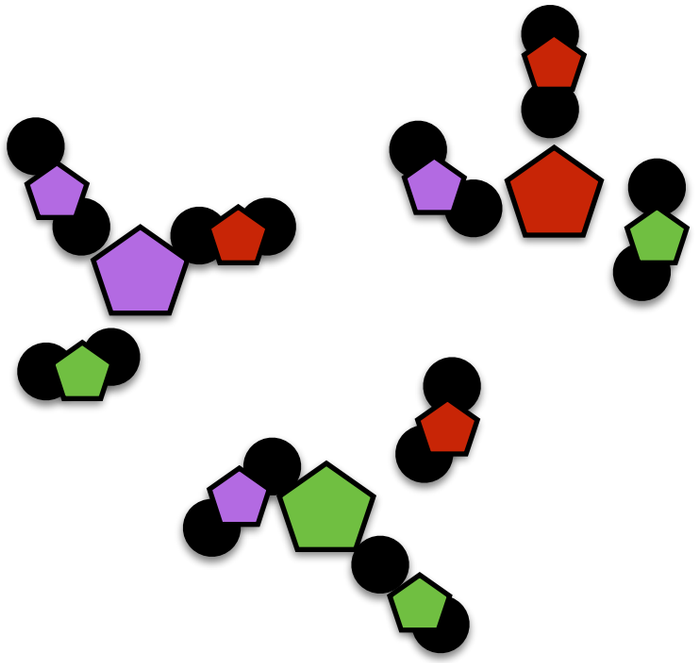
Hierarchical k-Means Tree

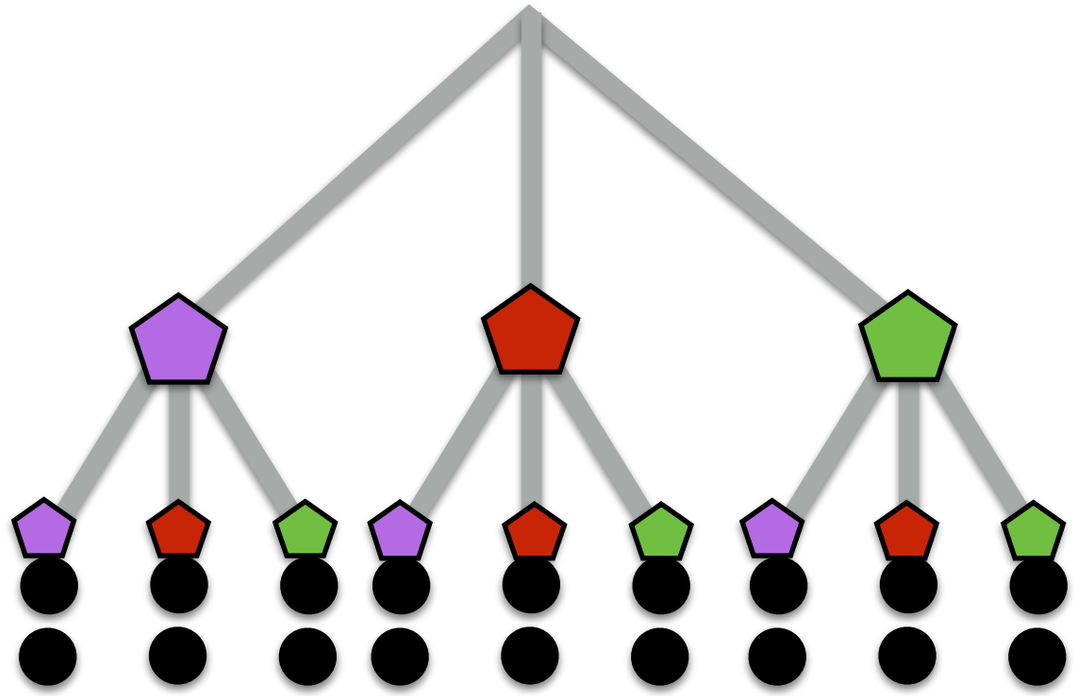




- Iteratively apply k-means clustering

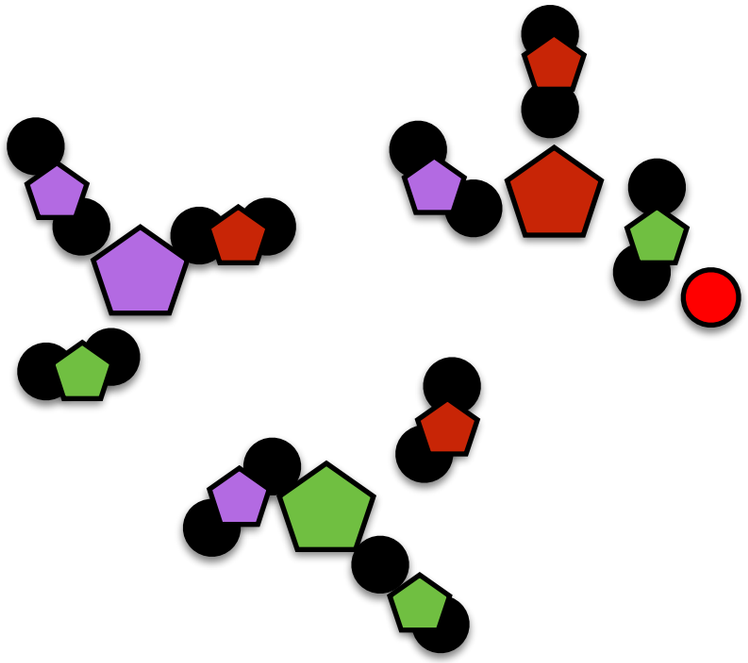
Hierarchical k-Means Tree

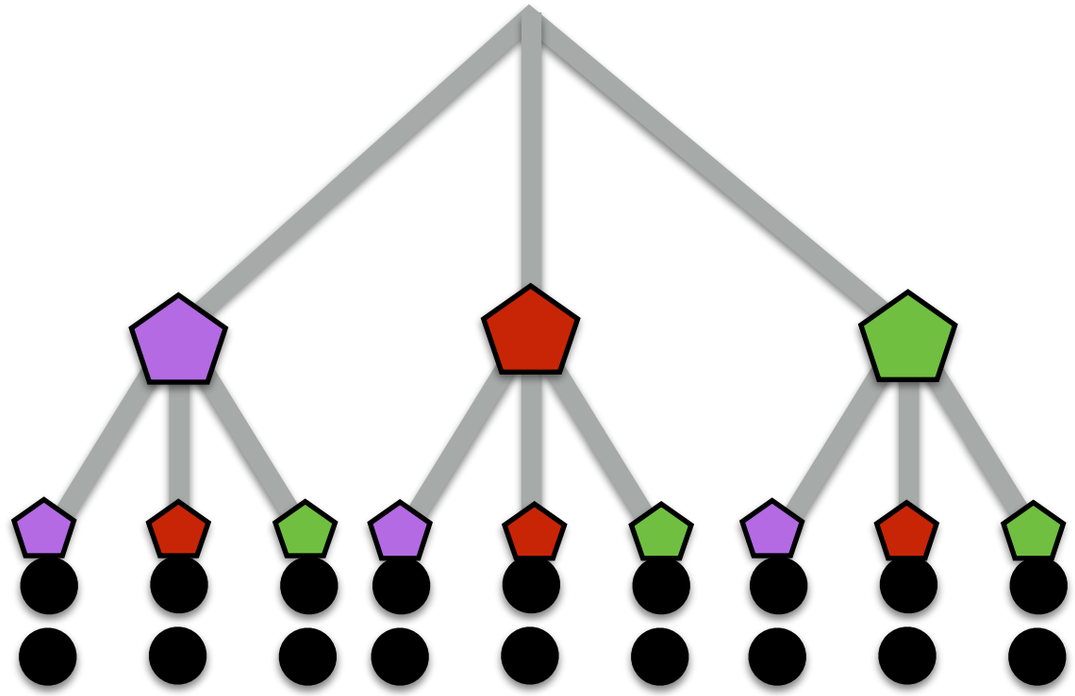




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

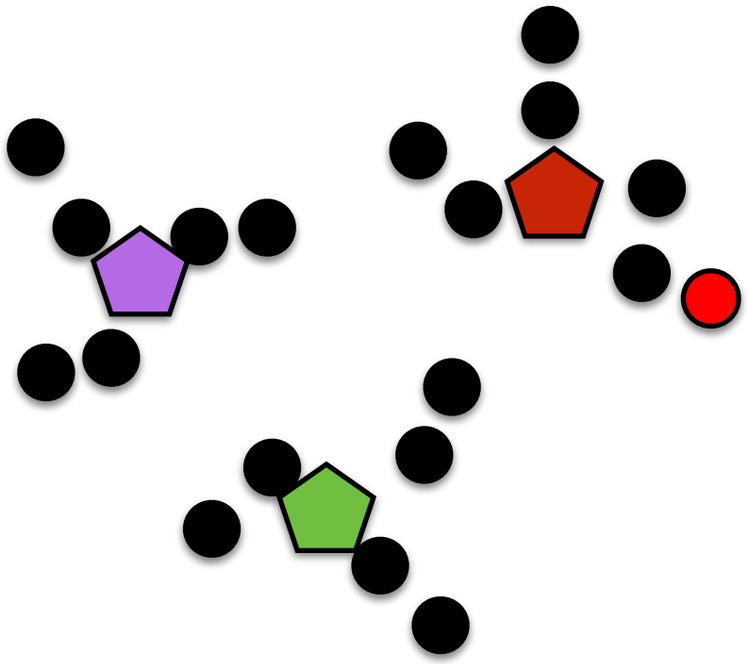
Hierarchical k-Means Tree

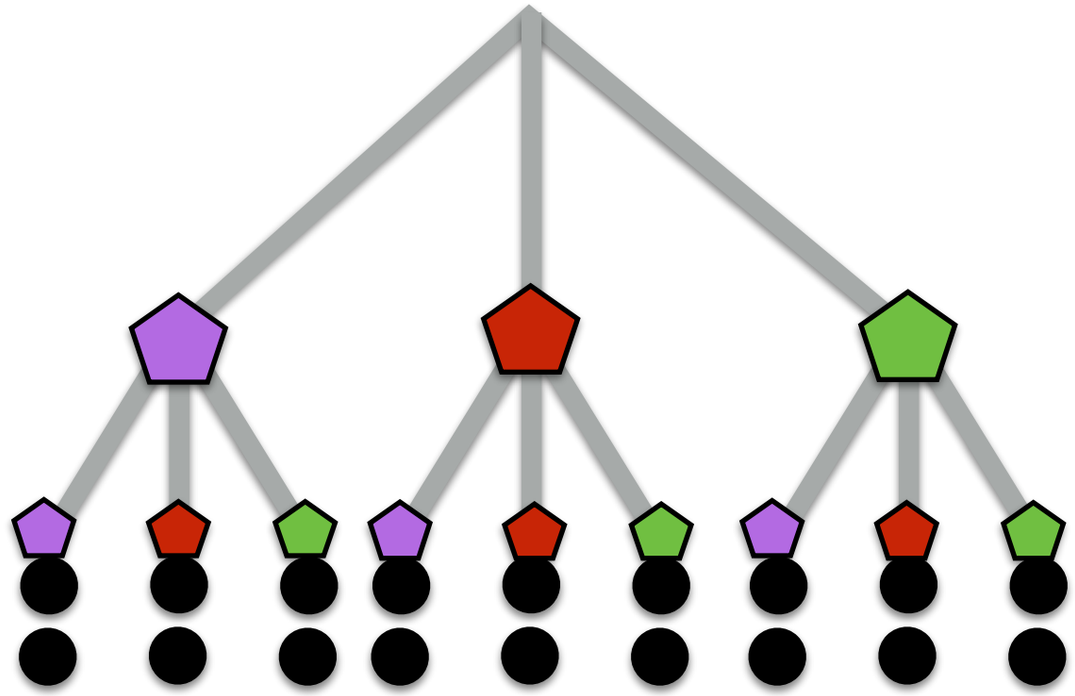




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

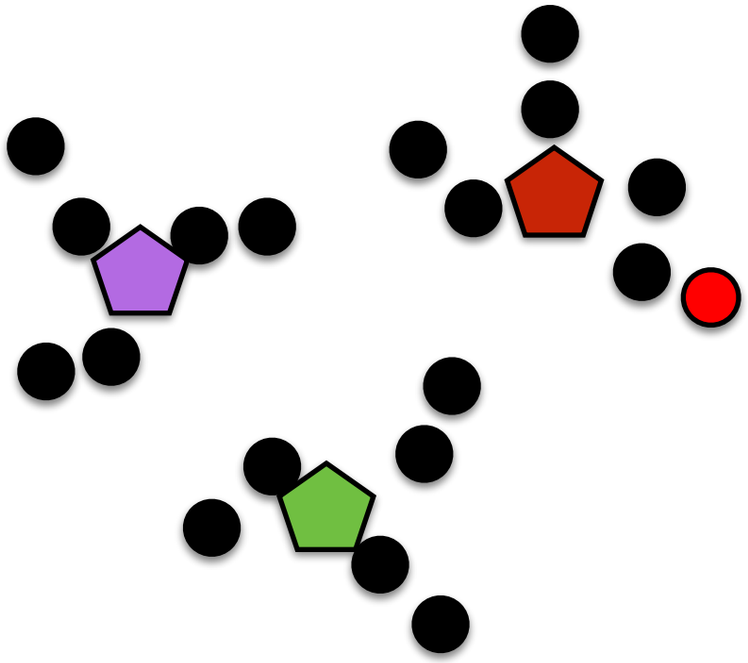
Hierarchical k-Means Tree

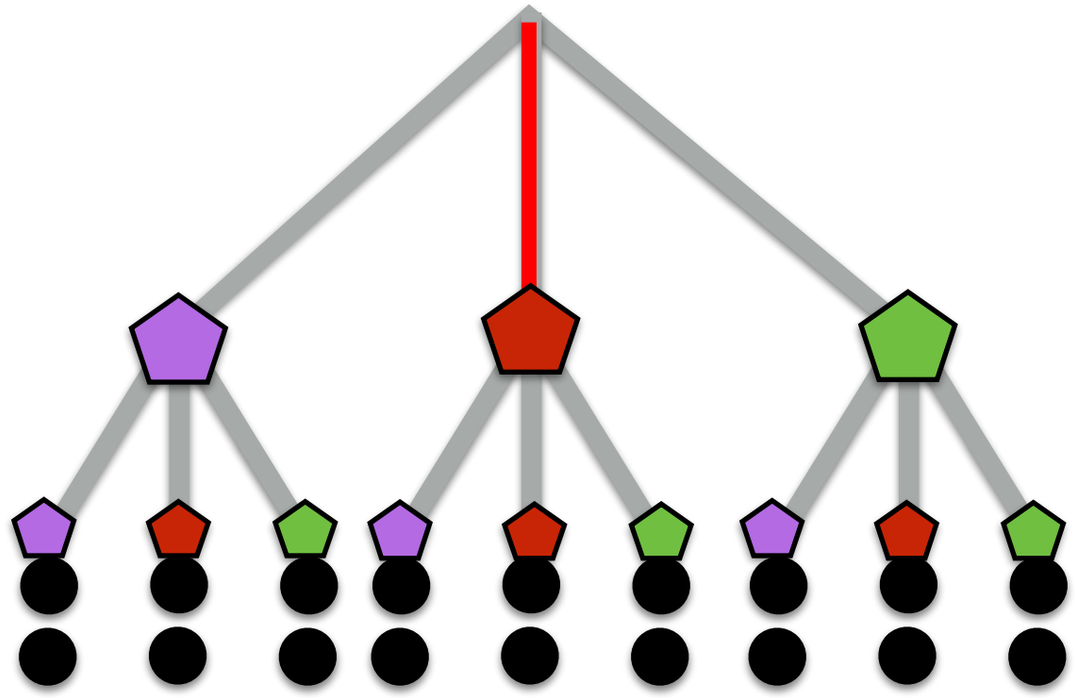




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

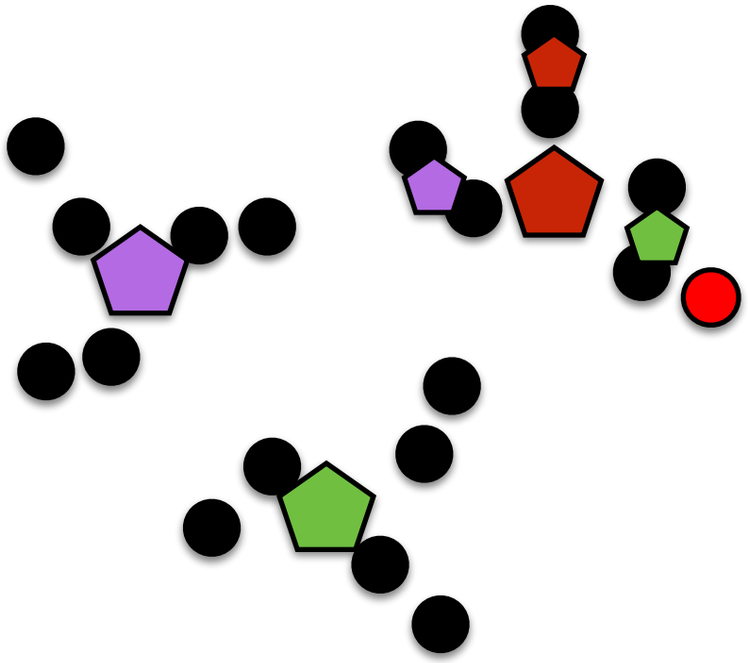
Hierarchical k-Means Tree

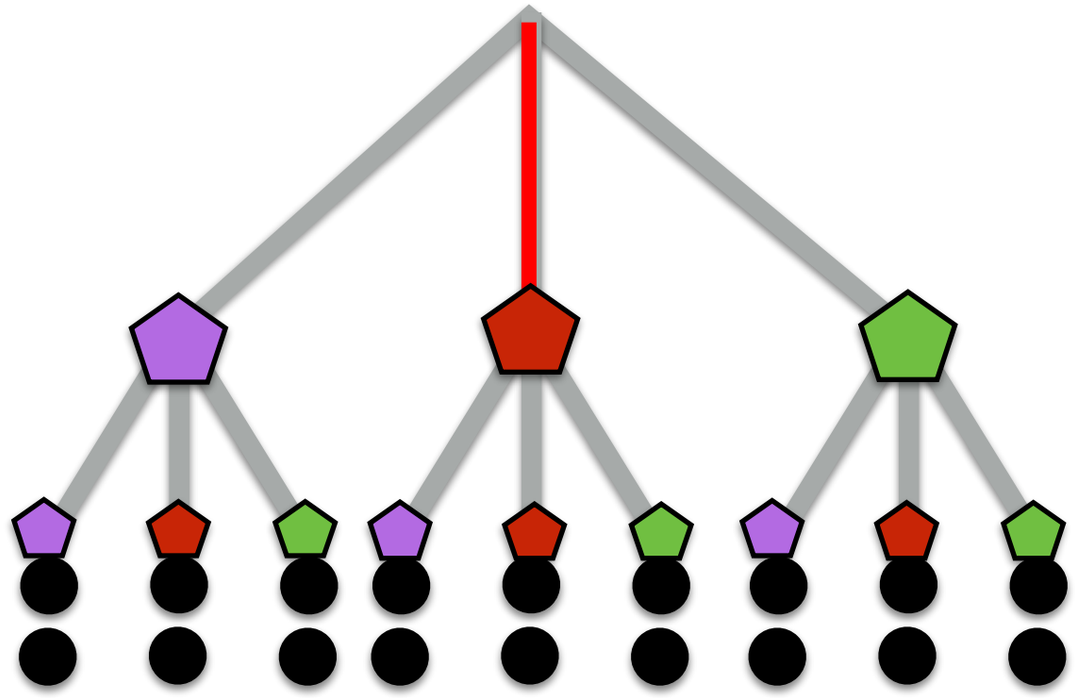




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

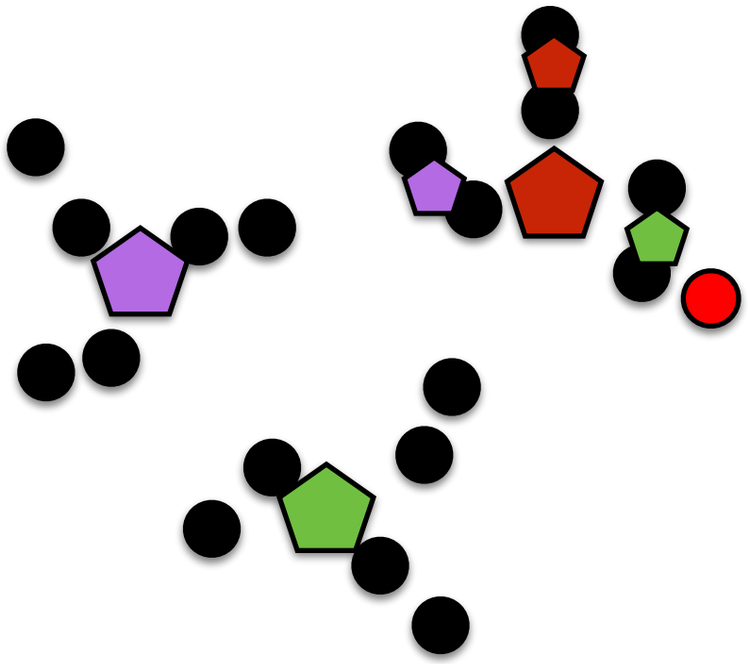
Hierarchical k-Means Tree

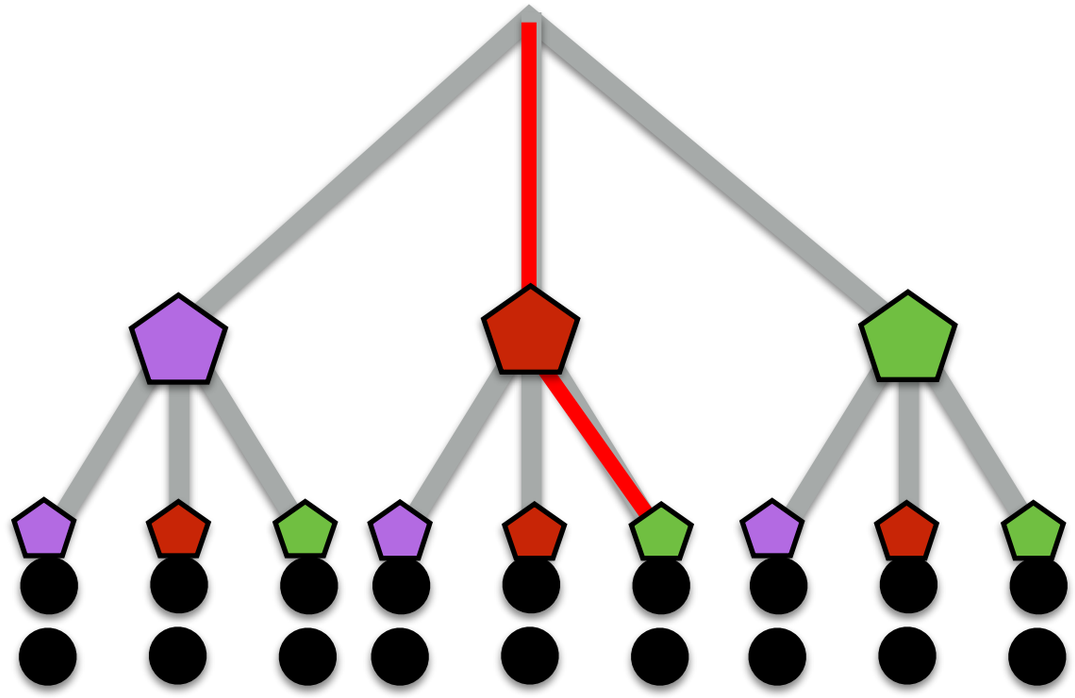




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

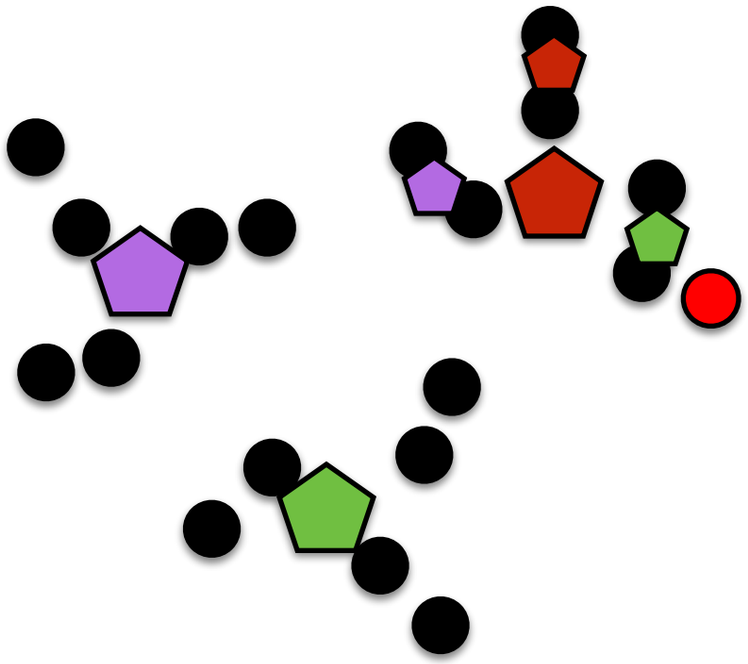
Hierarchical k-Means Tree

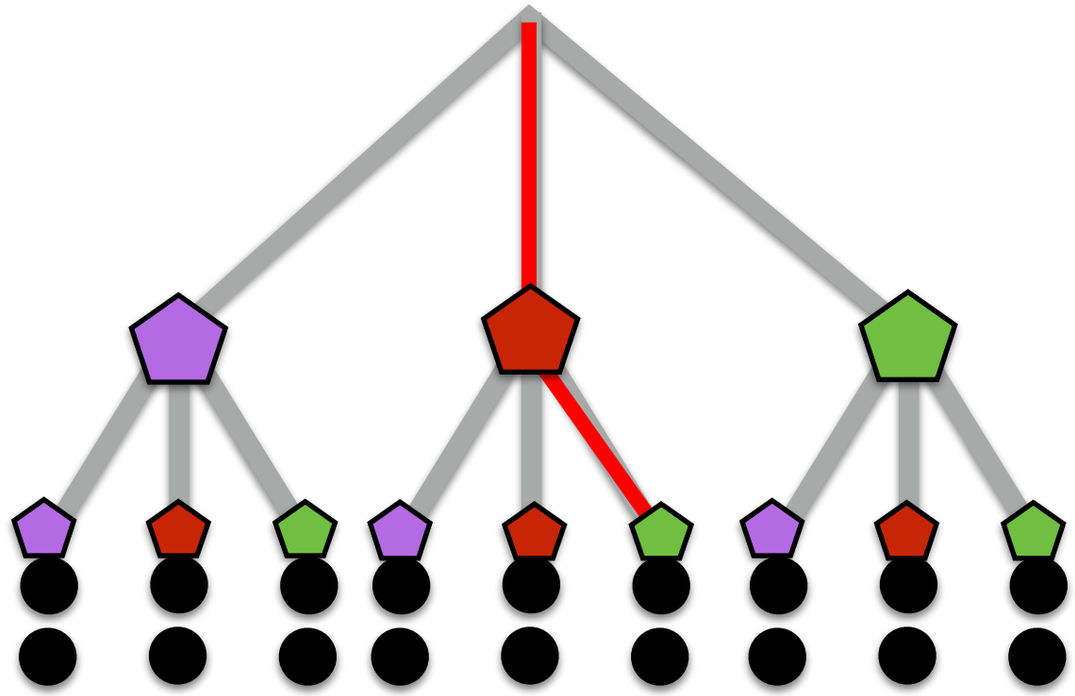




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster

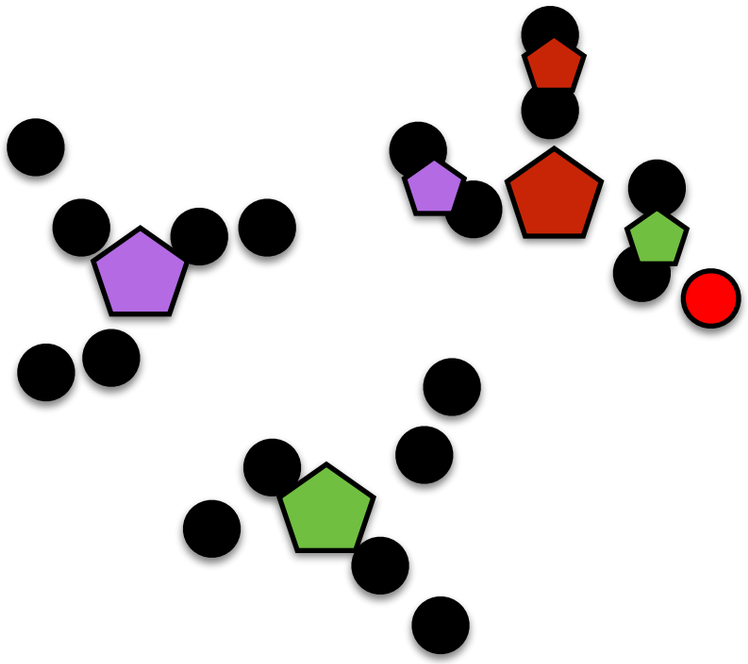
Hierarchical k-Means Tree

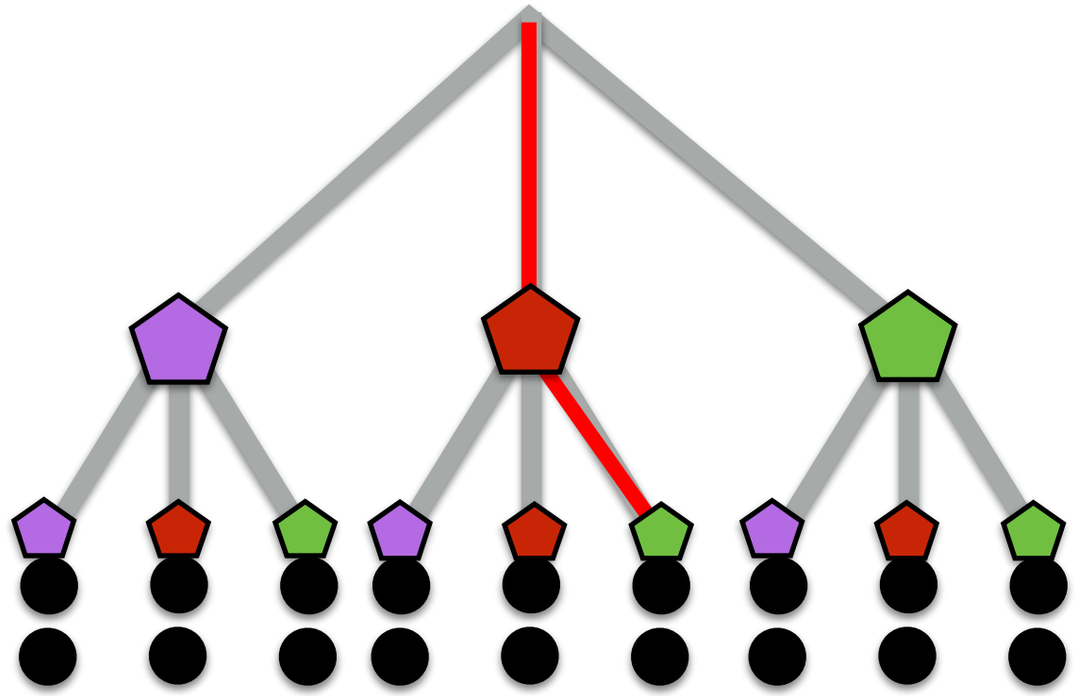




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster
- Approximate search: Visit `N` leaf nodes

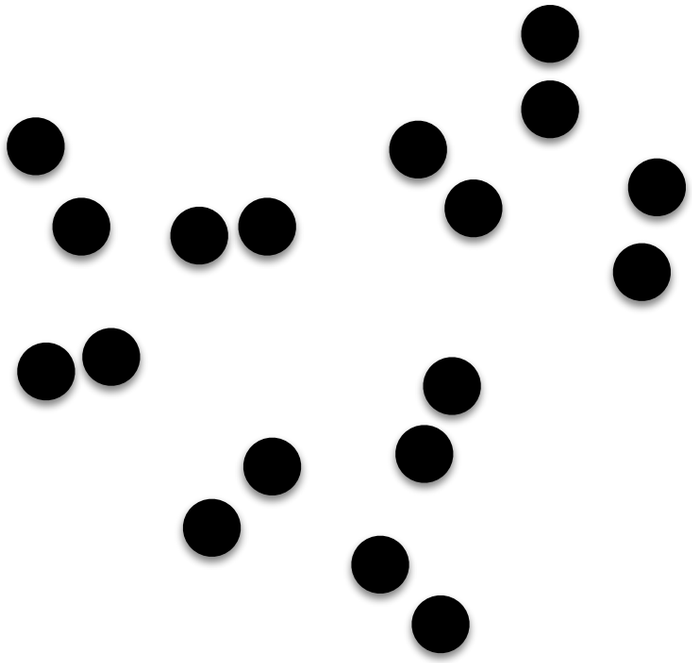
Hierarchical k-Means Tree

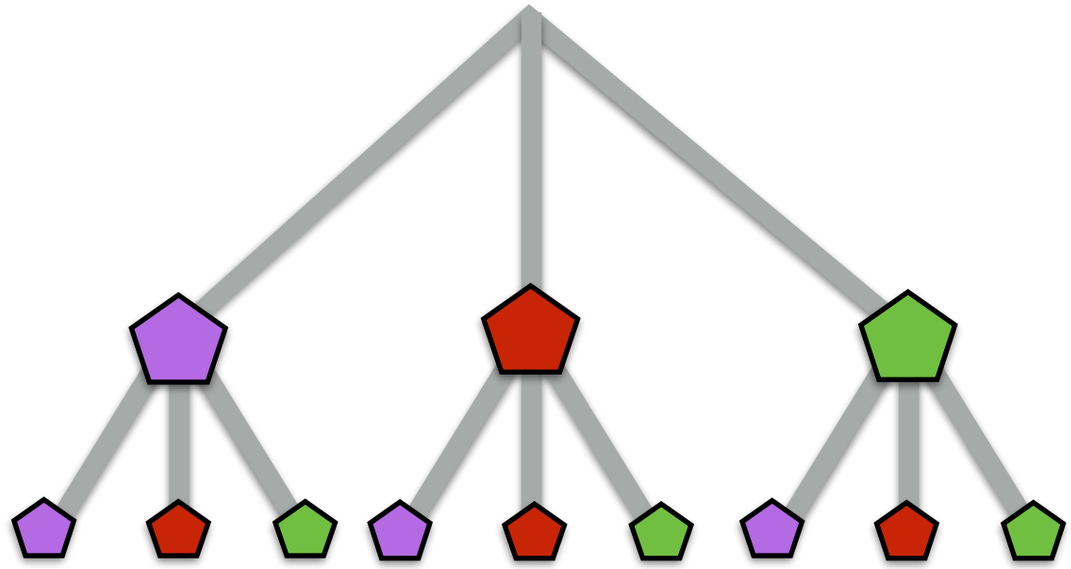




- Iteratively apply k-means clustering
- Traversal based on nearest neighboring cluster
- Approximate search: Visit `N` leaf nodes
- Performs (slightly) worse than kd-tree

Vocabulary Trees

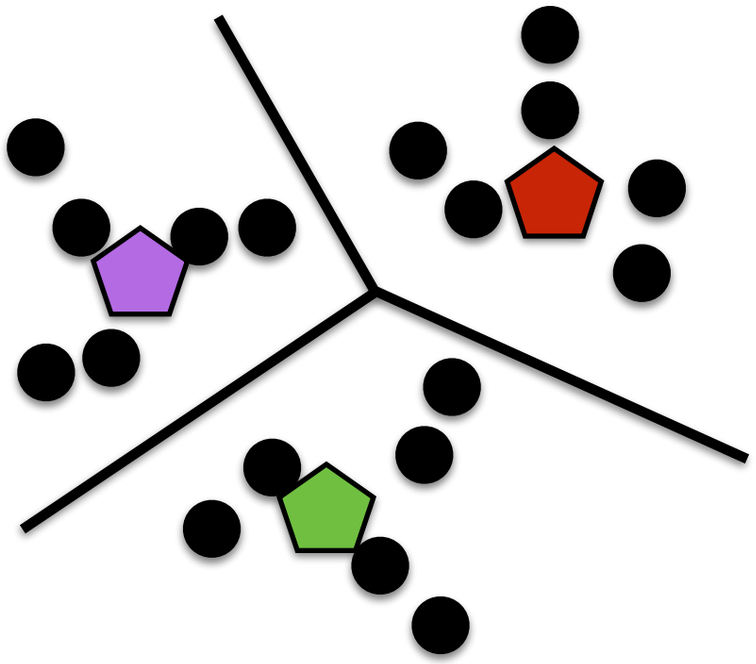


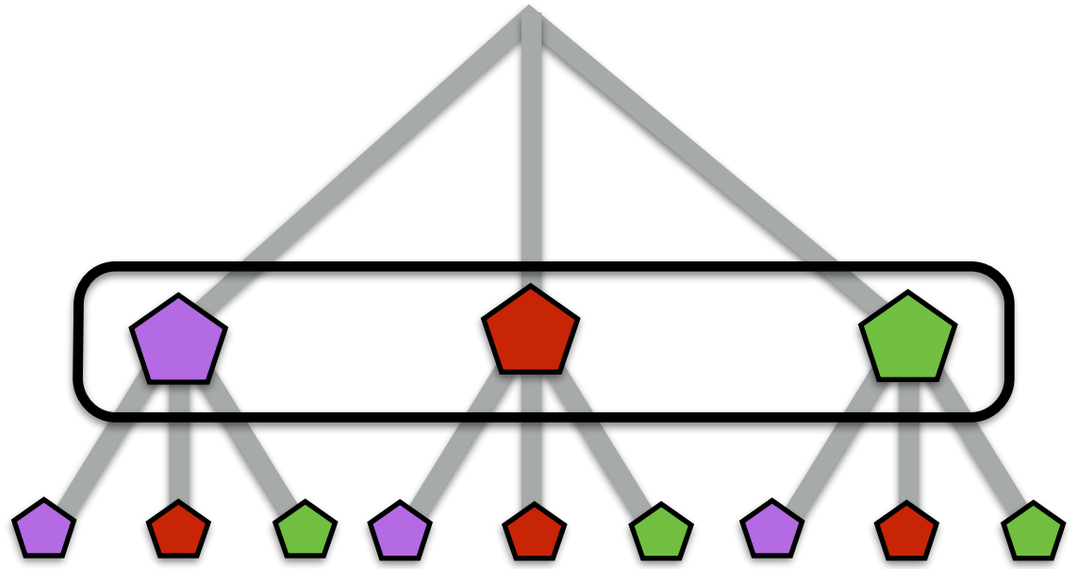


- Each level in a hierarchical k-means tree defines a quantization of the descriptor space (**visual vocabulary**)
- Hierarchical k-means trees also known as **Vocabulary Trees**

  [Nister & Stewenius, CVPR'06]

Vocabulary Trees

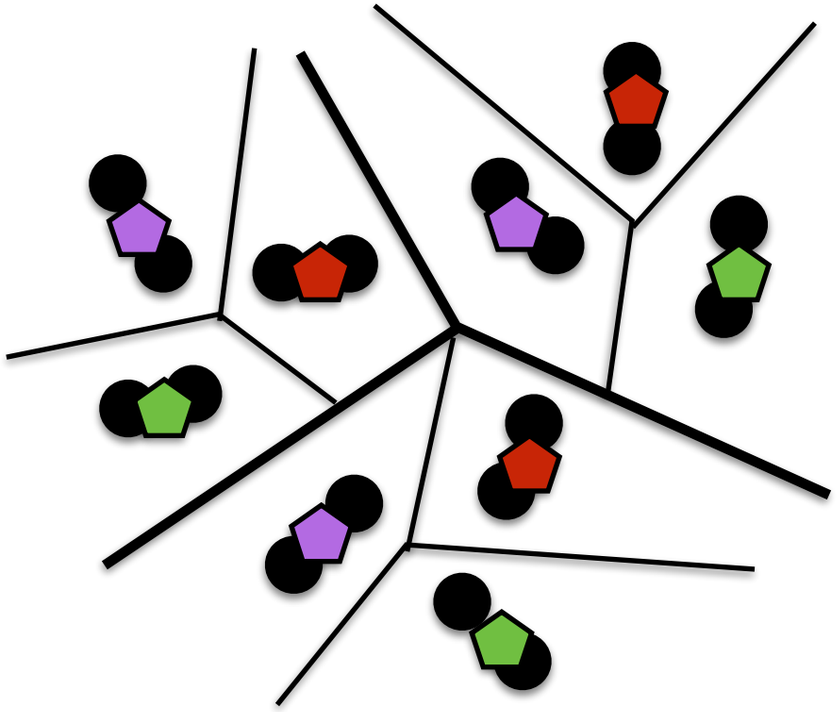


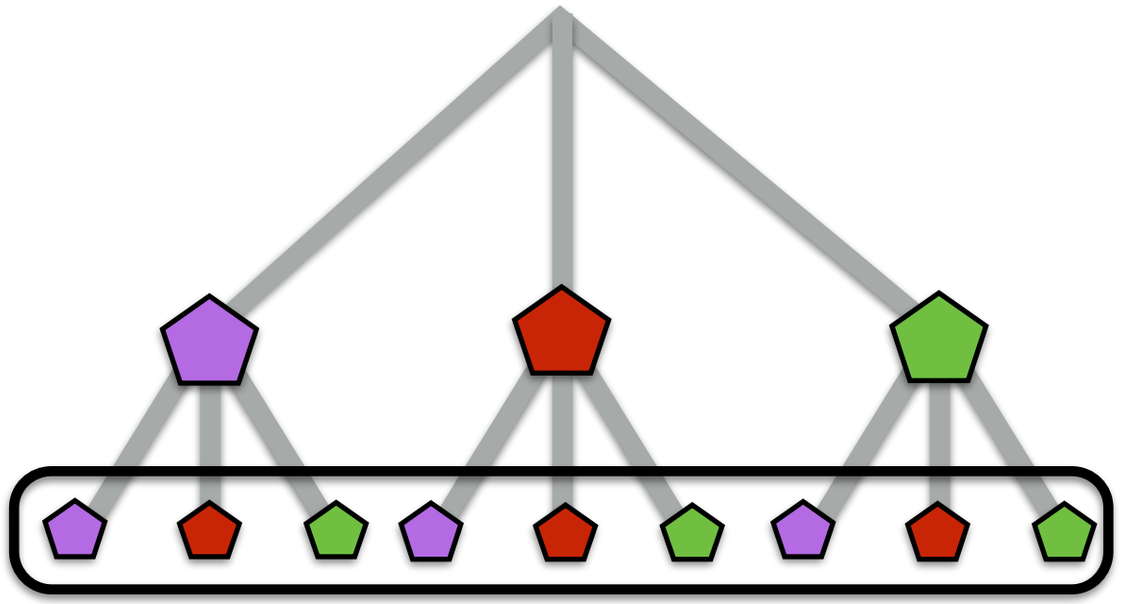


- Each level in a hierarchical k-means tree defines a quantization of the descriptor space (**visual vocabulary**)
- Hierarchical k-means trees also known as **Vocabulary Trees**

  [Nister & Stewenius, CVPR'06]

Vocabulary Trees





- Each level in a hierarchical k-means tree defines a quantization of the descriptor space (**visual vocabulary**)
- Hierarchical k-means trees also known as **Vocabulary Trees**

[Nister & Stewenius, CVPR'06]

Fundamentals

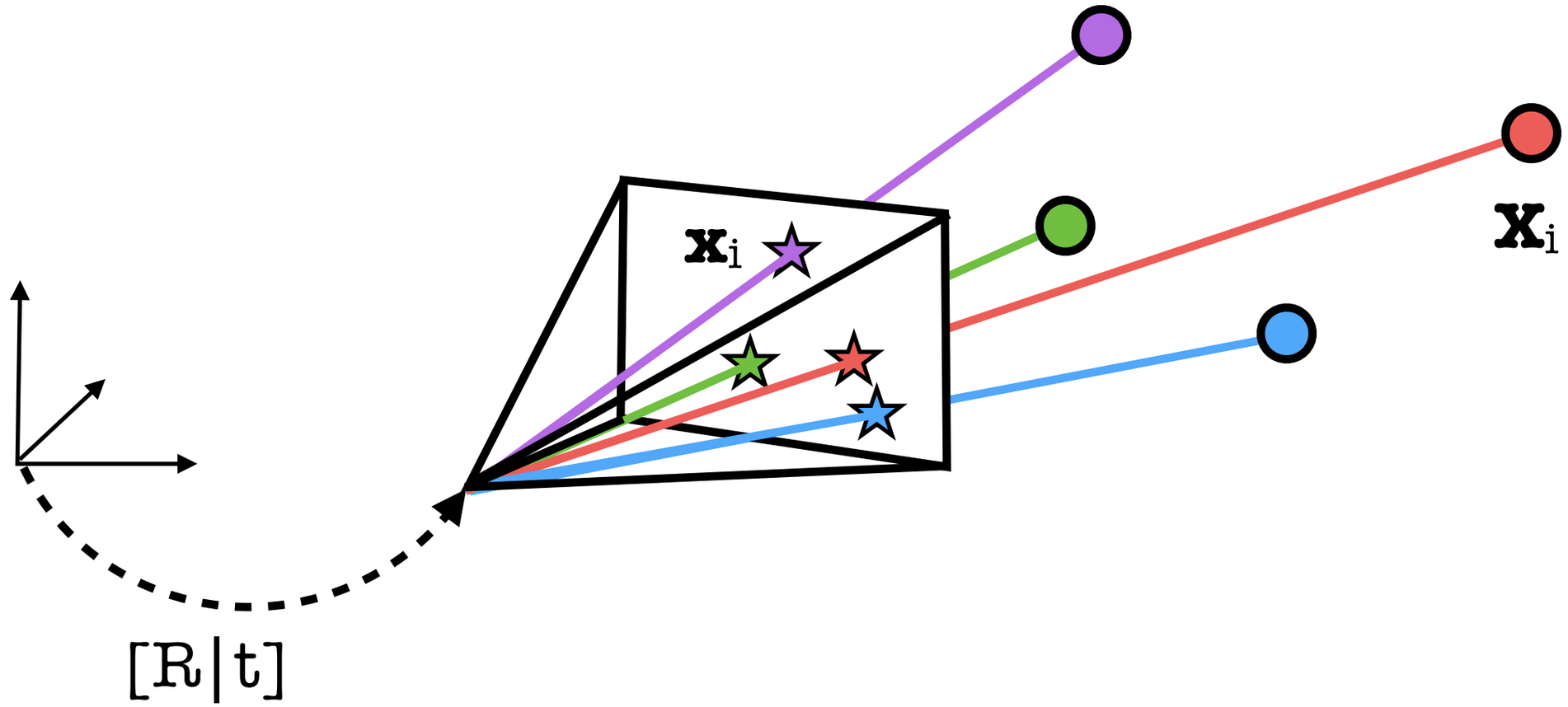
Structure-from-Motion

Establish 2D-3D Matches

Camera Pose Estimation:
RANSAC + n-Point-Pose Algorithm

(This is meant as a short overview, no in-depth discussion)

n-Point Pose Problem (PnP)



- Given: *n* 2D-3D correspondences ($\mathbf{x}_i$, $\mathbf{X}_i$)
- Compute pose $[\mathtt{R}|\mathtt{t}]$ s.t. $\mathtt{K}[\mathtt{R}|\mathtt{t}]\mathbf{X}_i = \alpha_i\mathbf{x}_i$, $\alpha_i>0$
- Optionally: Also estimate internal calibration matrix $\mathtt{K}$
  - In form of individual parameters
  - In form of projection matrix $\mathtt{P} = \mathtt{K}[\mathtt{R}|\mathtt{t}]$

RANdom SAmple Consensus

- Wrong matches cause problems for pose estimation
- RANdom SAmple Consensus (RANSAC) robustly handles outlier data [Fischler, Bolles, CACM'91]

RANdom SAmple Consensus

- Wrong matches cause problems for pose estimation
- RANdom SAmple Consensus (RANSAC) robustly handles outlier data [Fischler, Bolles, CACM'91]

While probability of missing correct model >η

Estimate model from *n* random data points

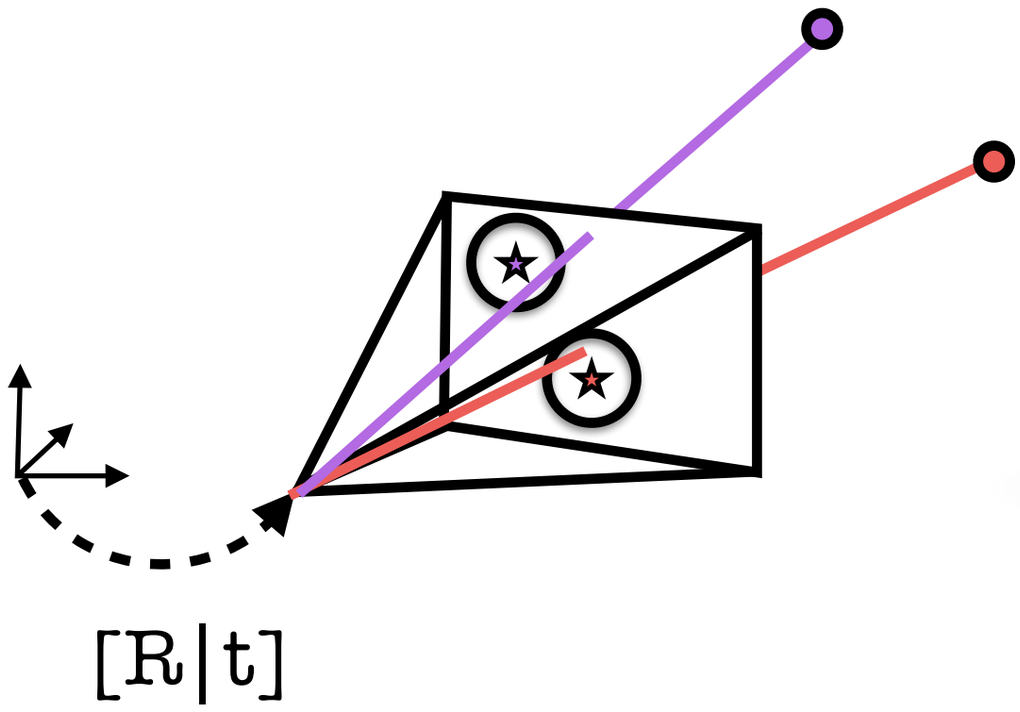
Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

RANdom SAmple Consensus

- Wrong matches cause problems for pose estimation
- RANdom SAmple Consensus (RANSAC) robustly handles outlier data [Fischler, Bolles, CACM'91]



While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

$\mathtt{I}$ = # inliers

$\mathtt{M}$ = # matches

$\varepsilon = \mathtt{I} / \mathtt{M}$ (**inlier ratio**)

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= **#inliers**) of model

Update best model

Return: Model with most inliers

$\mathtt{I}$ = # inliers

$\mathtt{M}$ = # matches

$\varepsilon = \mathtt{I} / \mathtt{M}$ (**inlier ratio**)

- Assumption: Every all-inlier sample leads to correct model

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

$\mathtt{I}$ = # inliers

$\mathtt{M}$ = # matches

$\varepsilon = \mathtt{I} / \mathtt{M}$ (**inlier ratio**)

- Assumption: Every all-inlier sample leads to correct model
- Probability of not drawing all-inlier sample in $\mathtt{k}$ iterations

$$(1 - \varepsilon^n)^k$$

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= **#inliers**) of model

Update best model

Return: Model with most inliers

$\mathtt{I}$ = # inliers

$\mathtt{M}$ = # matches

$\varepsilon = \mathtt{I} / \mathtt{M}$ (**inlier ratio**)

- Assumption: Every all-inlier sample leads to correct model
- Probability of not drawing all-inlier sample in $\mathtt{k}$ iterations

$$(1 - \varepsilon^n)^k$$

- Number of required iterations for uncertainty η

$$k_{max} = \log(\eta) / \log(1-\varepsilon^n)$$

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= **#inliers**) of model

Update best model

Return: Model with most inliers

$\mathtt{I}$ = # inliers

$\mathtt{M}$ = # matches

$\varepsilon = \mathtt{I} / \mathtt{M}$ (**inlier ratio**)

- Assumption: Every all-inlier sample leads to correct model
- Probability of not drawing all-inlier sample in $\mathtt{k}$ iterations

$$(1 - \varepsilon^n)^k$$

- Number of required iterations for uncertainty η

$$k_{max} = \log(\eta) / \log(1-\varepsilon^n)$$

- In practice: ε not known, but iteratively estimated

RANSAC - Termination Criterion

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= **#inliers**) of model

Update best model

Return: Model with most inliers

I = # inliers

M = # matches

$\varepsilon = I / M$ (**inlier ratio**)

- Assumption: Every all-inlier sample leads to correct model
- Probability of not drawing all-inlier sample in k iterations

$$(1 - \varepsilon^n)^k$$

- Number of required iterations for uncertainty η

$$k_{max} = \log(\eta) / \log(1-\varepsilon^n)$$

- In practice: ε not known, but iteratively estimated
  - Initialize $k_{max} = \log(\eta) / \log(1-\varepsilon_{min}^n)$

$T_{d,d}$ Test

While probability of missing correct model >η

Estimate model from *n* random data points

If đ randomly selected points are inliers to model

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

- **Motivation**: Avoid evaluating bad models

$T_{d,d}$ Test

While probability of missing correct model >η

Estimate model from *n* random data points

If d randomly selected points are inliers to model

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

- **Motivation**: Avoid evaluating bad models
- **$T_{d,d}$ Test**: Fully evaluate model only if d random matches are inliers (d=1 in practice)

$T_{d,d}$ Test

While probability of missing correct model >η

Estimate model from *n* random data points

If d randomly selected points are inliers to model

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

- **Motivation**: Avoid evaluating bad models
- **$T_{d,d}$ Test**: Fully evaluate model only if d random matches are inliers (d=1 in practice)
  - Requires more iterations: $k_{max} = \log(\eta) / \log(1-\varepsilon^{n+d})$

$T_{d,d}$ Test

While probability of missing correct model >η

Estimate model from *n* random data points

If d randomly selected points are inliers to model

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

- **Motivation**: Avoid evaluating bad models
- **$T_{d,d}$ Test**: Fully evaluate model only if d random matches are inliers ($d=1$ in practice)
  - Requires more iterations: $k_{max} = \log(\eta) / \log(1-\varepsilon^{n+d})$
  - But around 2x speed-up due to early model rejection

$T_{d,d}$ Test

While probability of missing correct model >η

Estimate model from *n* random data points

If d randomly selected points are inliers to model

Estimate support (= #**inliers**) of model

Update best model

Return: Model with most inliers

- **Motivation**: Avoid evaluating bad models
- **$T_{d,d}$ Test**: Fully evaluate model only if d random matches are inliers ($d=1$ in practice)
  - Requires more iterations: $k_{max} = \log(\eta) / \log(1-\varepsilon^{n+d})$
  - But around 2x speed-up due to early model rejection
- Better early model rejection: [Chum & Matas, PAMI'08]

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

➡ More samples needed than predicted

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

  ➡ More samples needed than predicted

- **LO-RANSAC**: Inner RANSAC using non-minimal samples

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

  ➡ More samples needed than predicted

- **LO-RANSAC**: Inner RANSAC using non-minimal samples
  - On average executed only $\log(k_{max})+1$ times

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

  ➡ More samples needed than predicted

- **LO-RANSAC**: Inner RANSAC using non-minimal samples
  - On average executed only $\log(k_{max})+1$ times
  - Required number iterations close to theoretical prediction

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

  ➡ More samples needed than predicted

- **LO-RANSAC**: Inner RANSAC using non-minimal samples
  - On average executed only $\log(k_{max})+1$ times
  - Required number iterations close to theoretical prediction
  - Better model quality

RANSAC with Local Optimizaton

While probability of missing correct model >η

Estimate model from *n* random data points

Estimate support (= #**inliers**) of model

If new best model

Perform Local Optimization (LO) & update best model

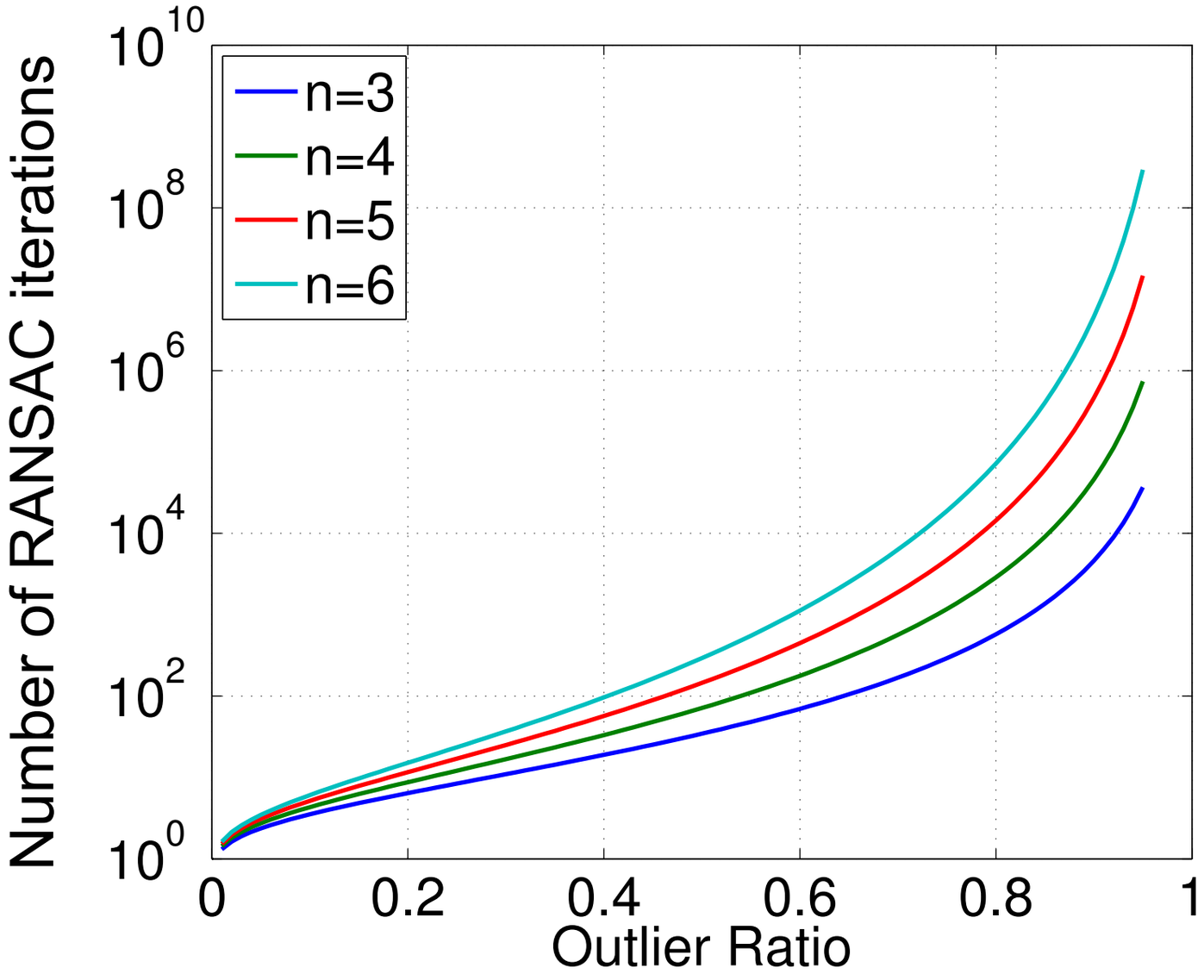
Return: Model with most inliers

- Due to noise: Not every all-inlier sample results in correct model

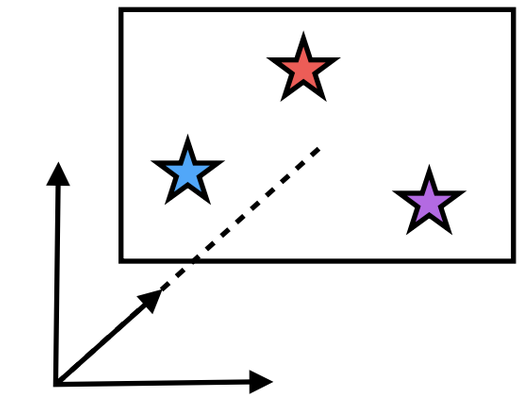
  ➡ More samples needed than predicted

- **LO-RANSAC**: Inner RANSAC using non-minimal samples
  - On average executed only $\log(k_{max})+1$ times
  - Required number iterations close to theoretical prediction
  - Better model quality
  - **Highly recommended!**

The Need for Minimal Subsets

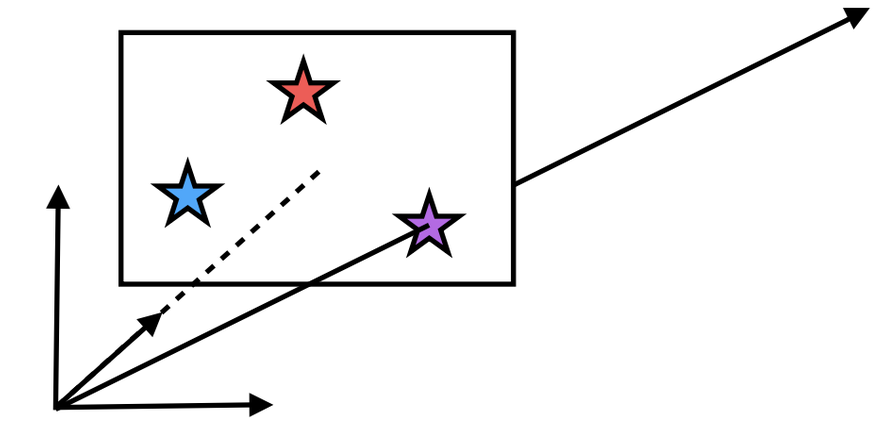


3-Point Pose Problem (P3P)



- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

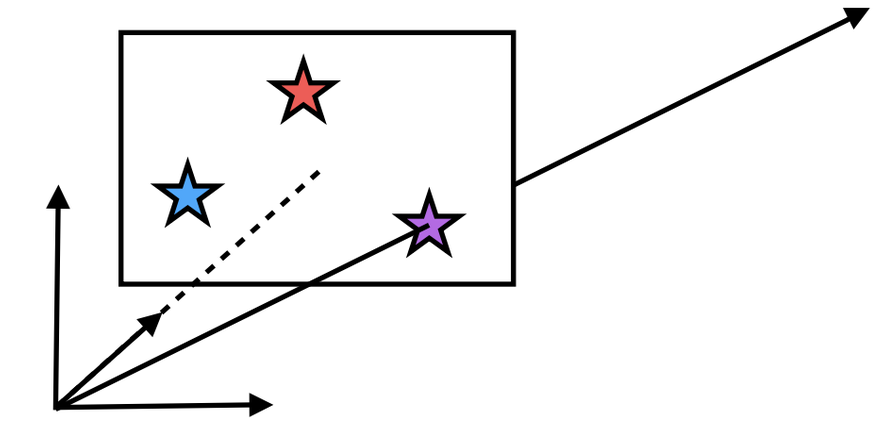
3-Point Pose Problem (P3P)



camera coordinates

- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

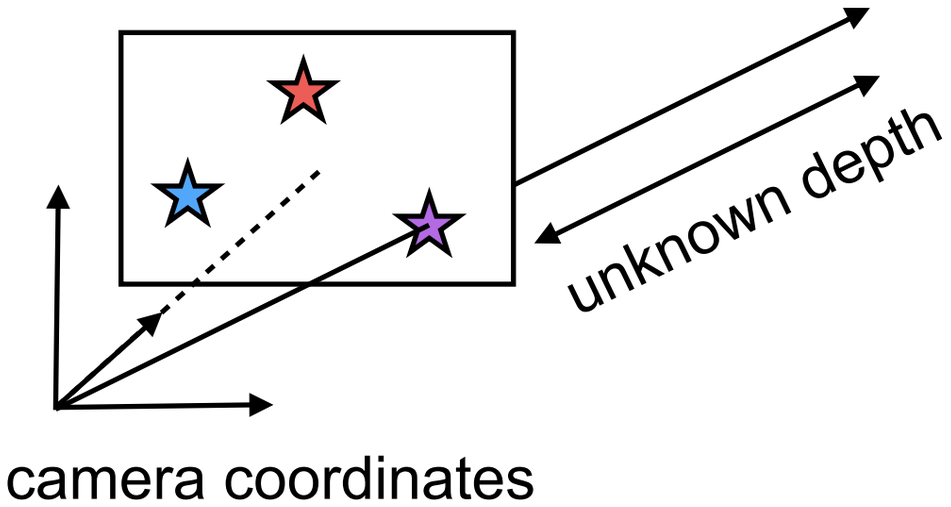
3-Point Pose Problem (P3P)



camera coordinates

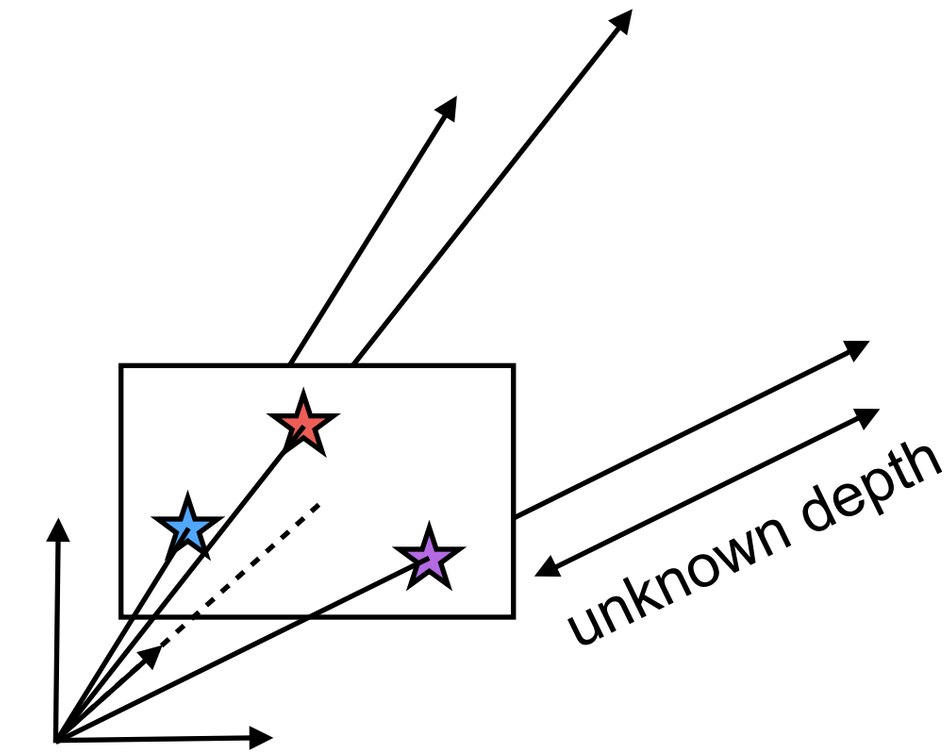
- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

3-Point Pose Problem (P3P)



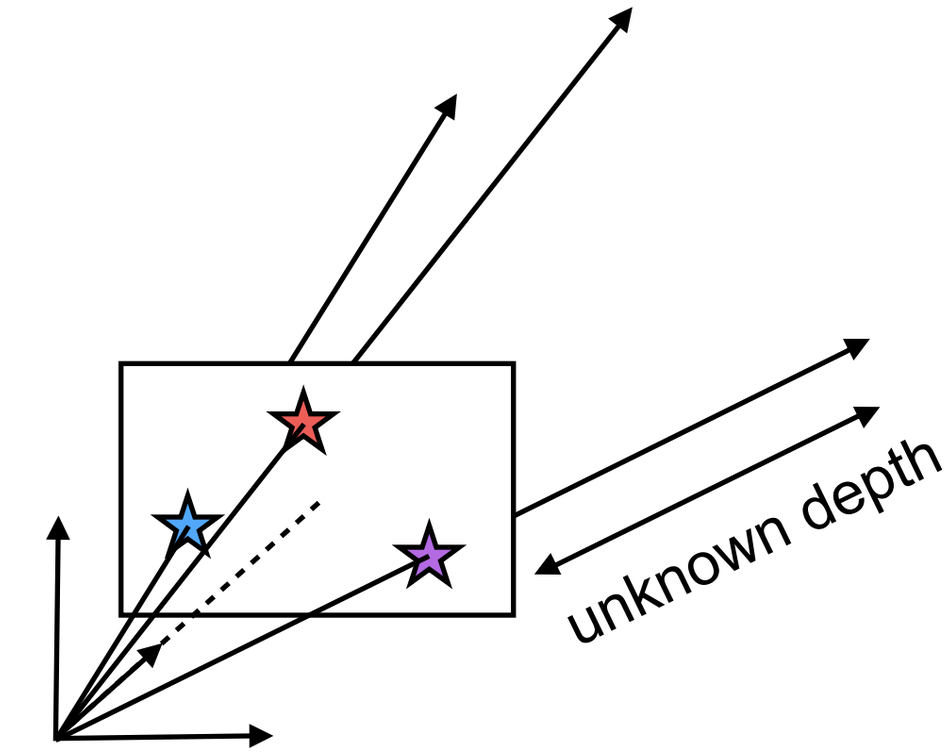
- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

3-Point Pose Problem (P3P)

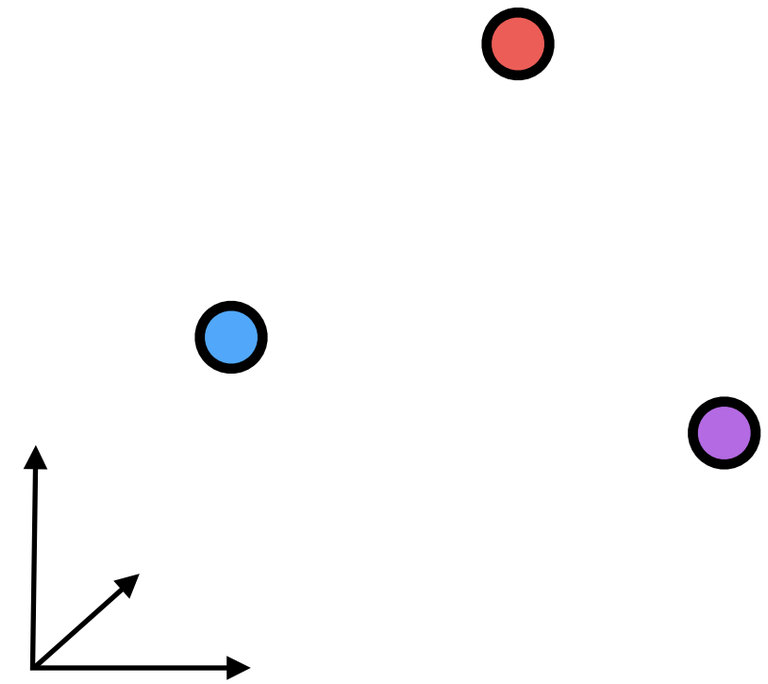


- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

3-Point Pose Problem (P3P)



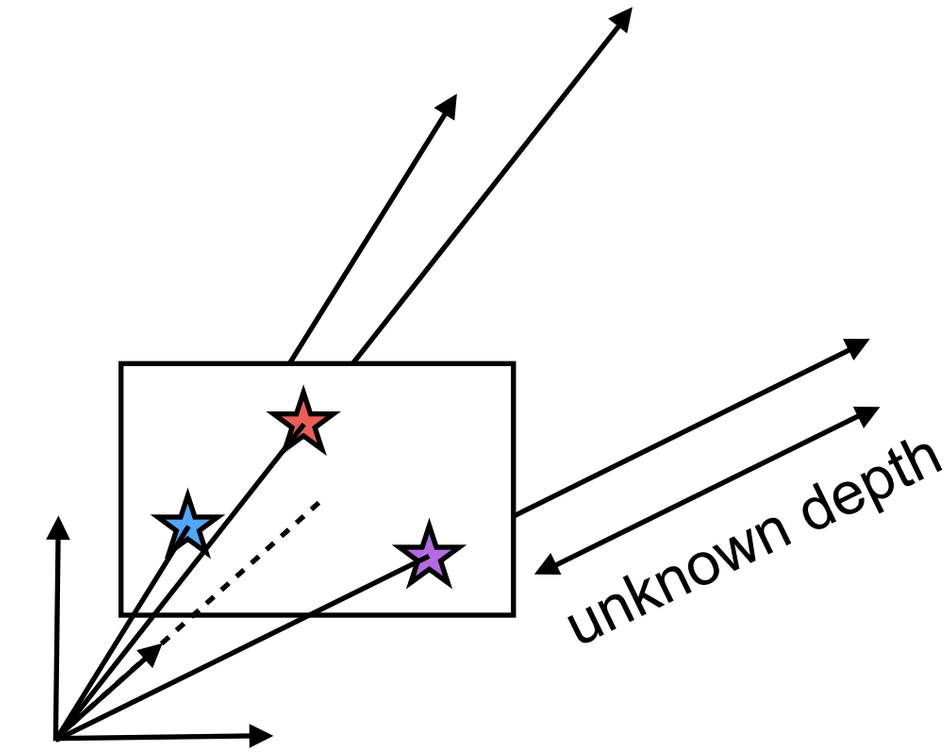
camera coordinates

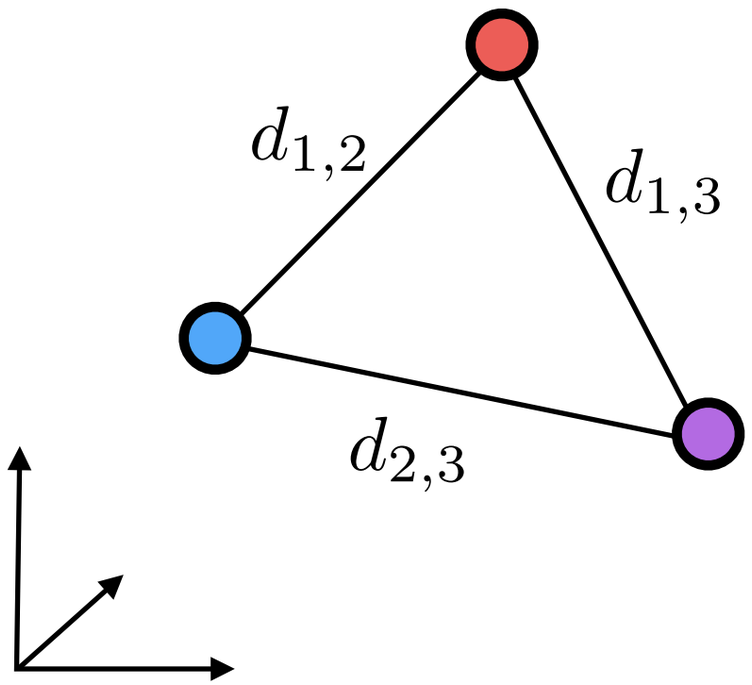


world coordinates

- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

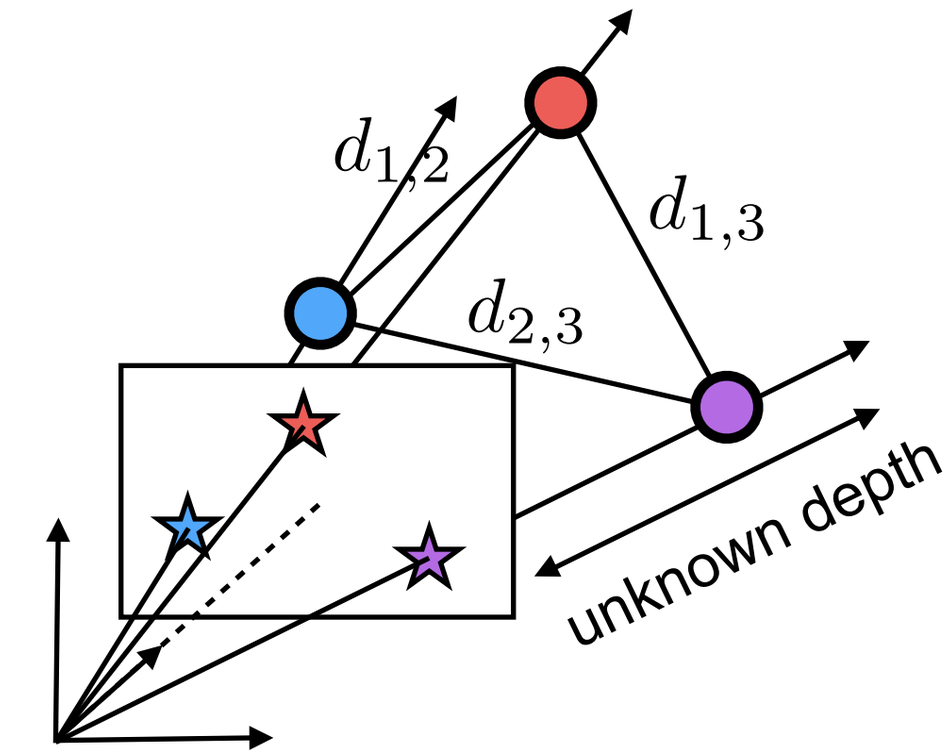
3-Point Pose Problem (P3P)



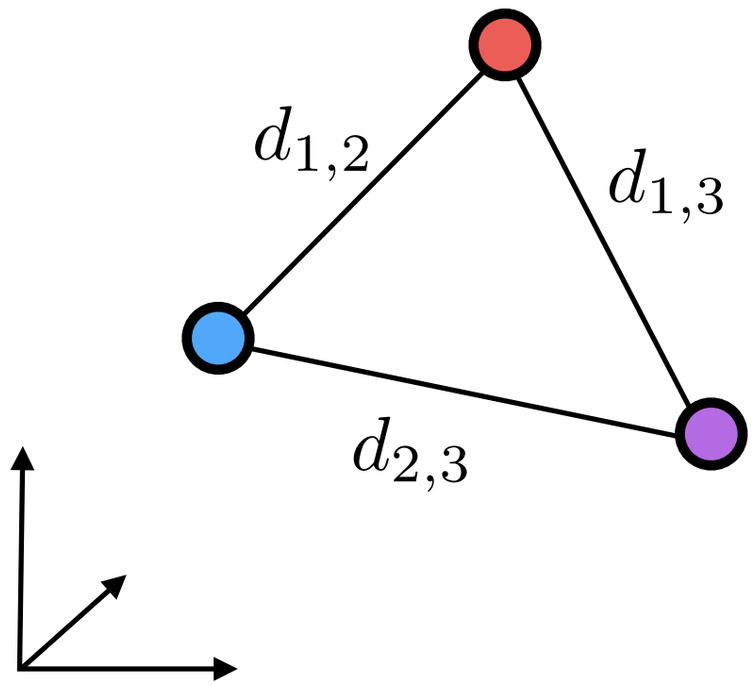


- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

3-Point Pose Problem (P3P)



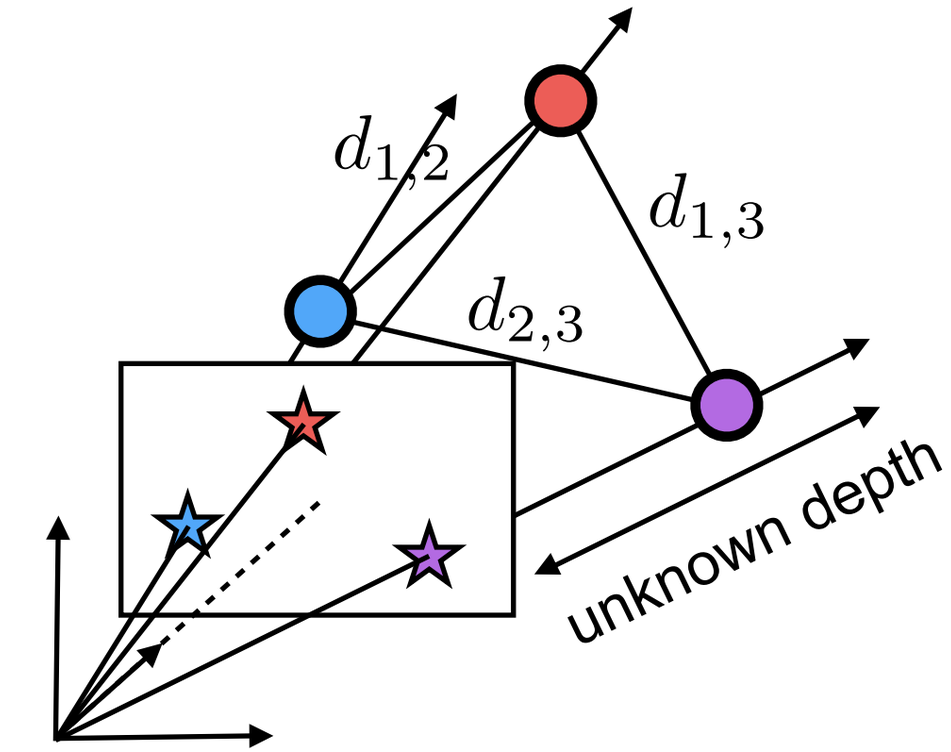
camera coordinates



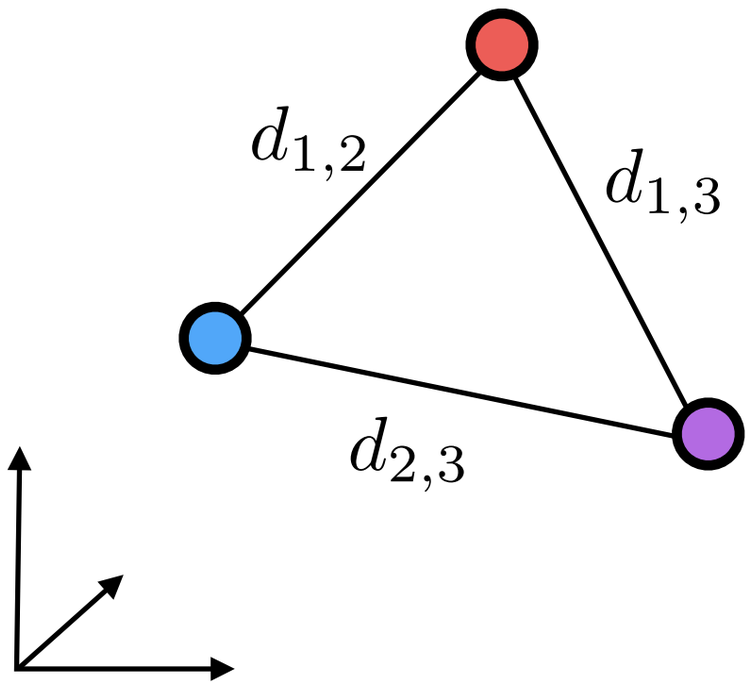
world coordinates

- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]

3-Point Pose Problem (P3P)



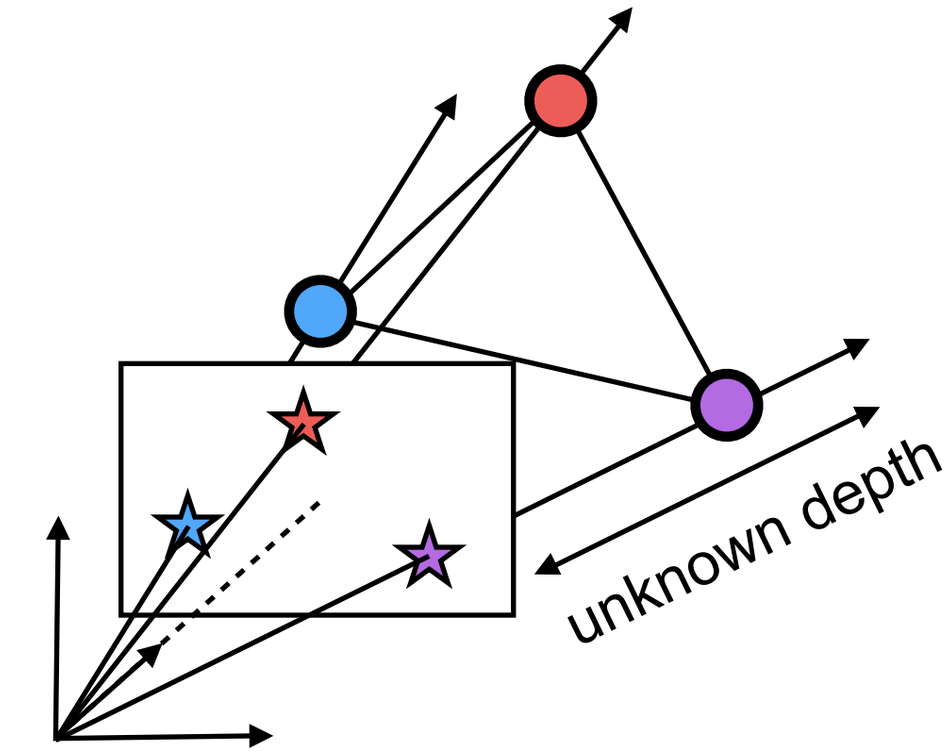
camera coordinates



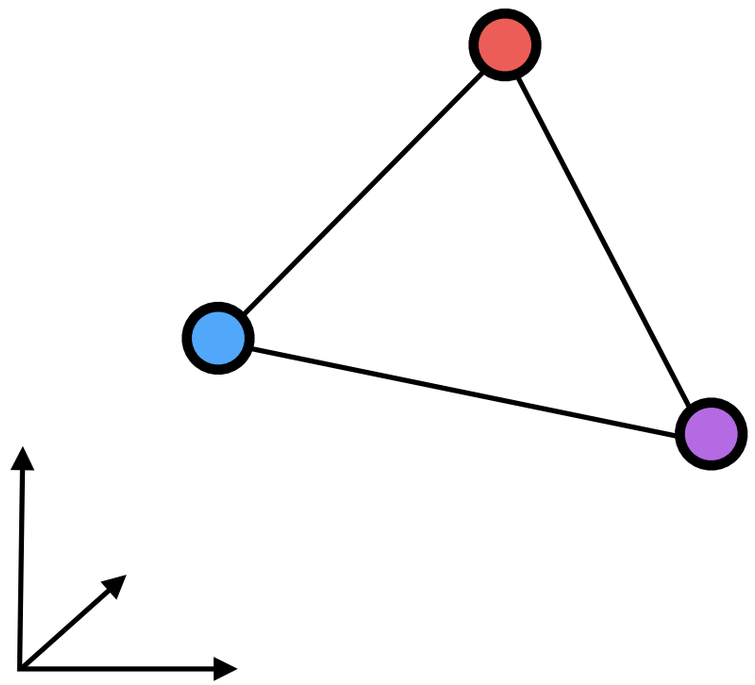
world coordinates

- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]

3-Point Pose Problem (P3P)



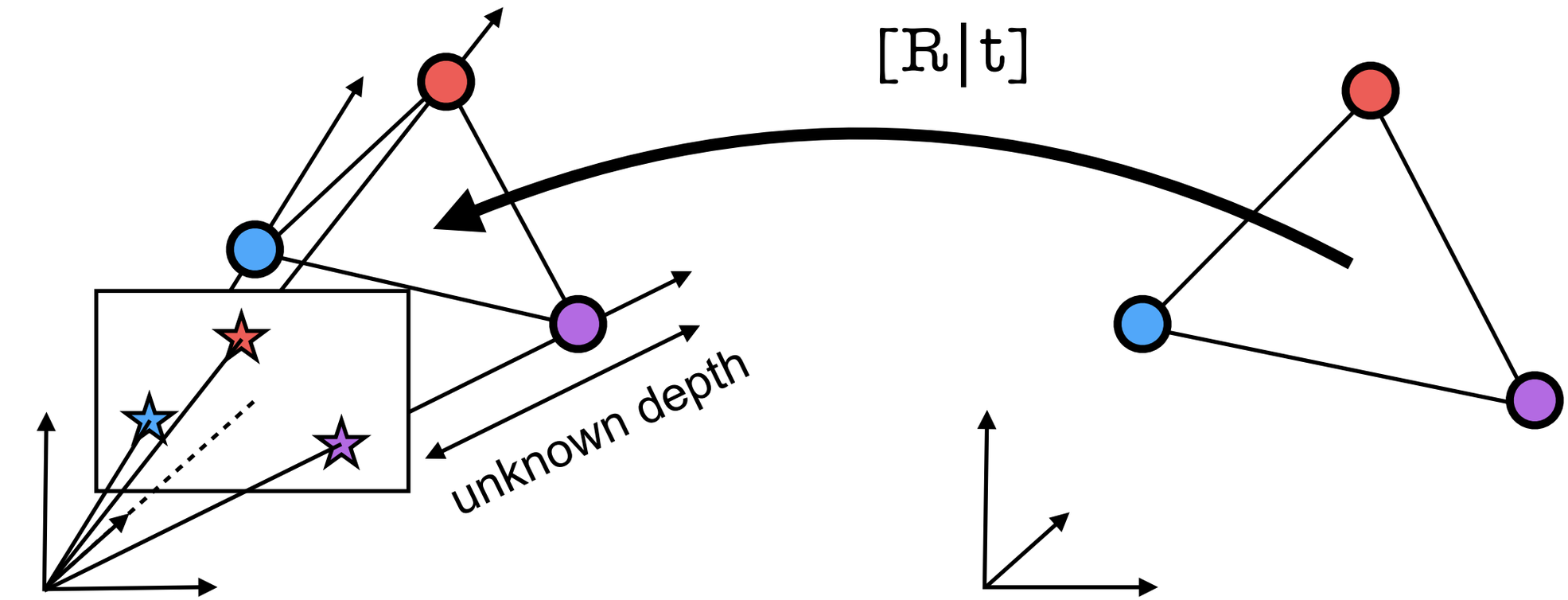
camera coordinates



world coordinates

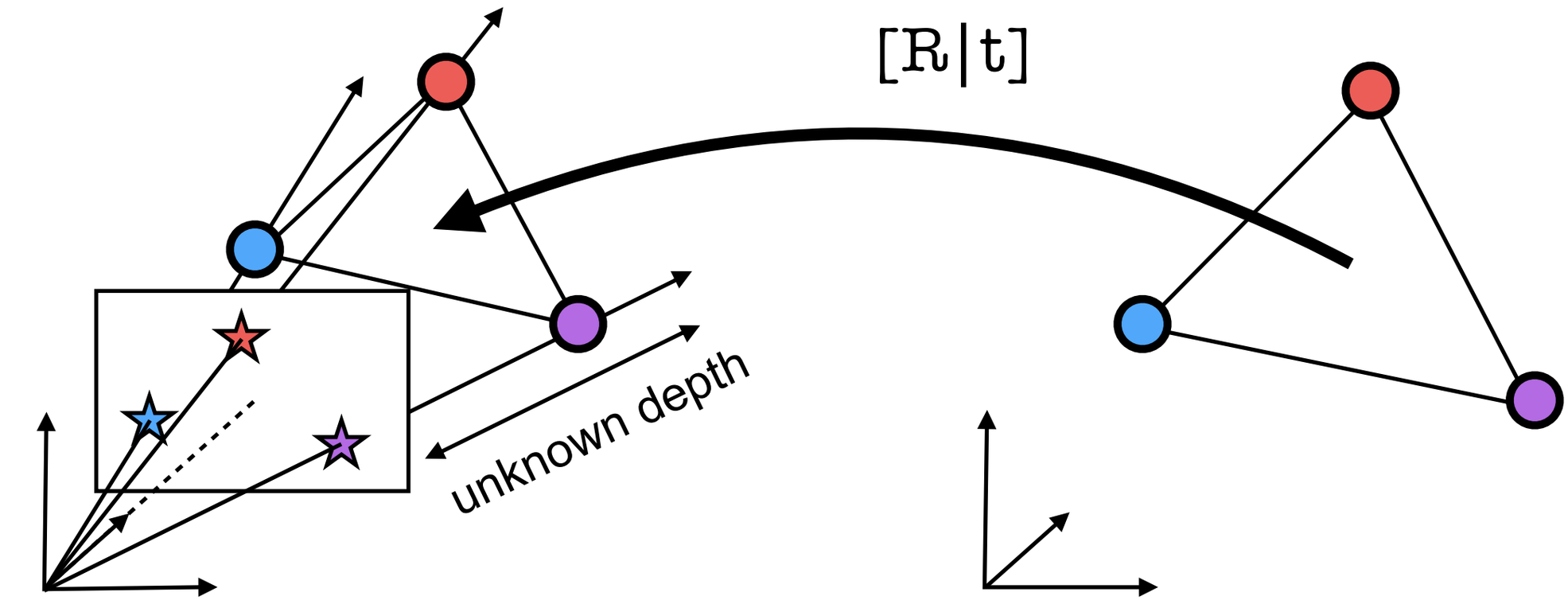
- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]

3-Point Pose Problem (P3P)



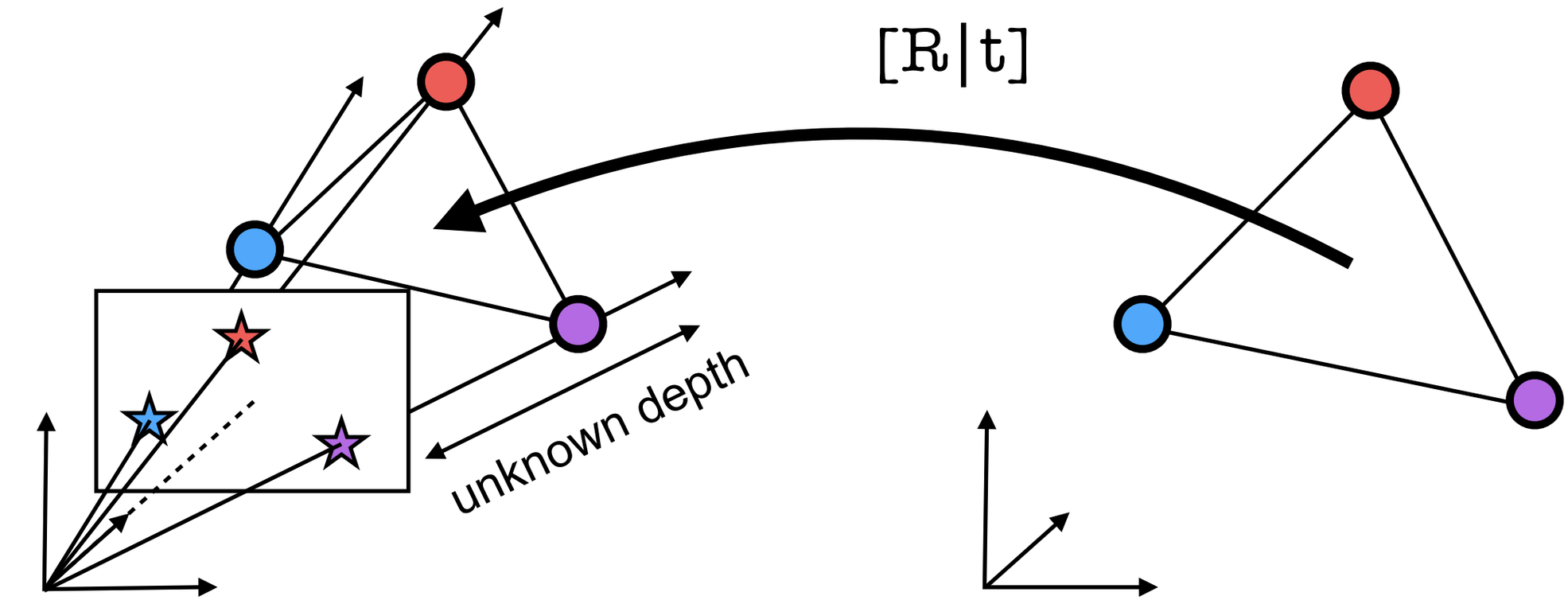
- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]

3-Point Pose Problem (P3P)



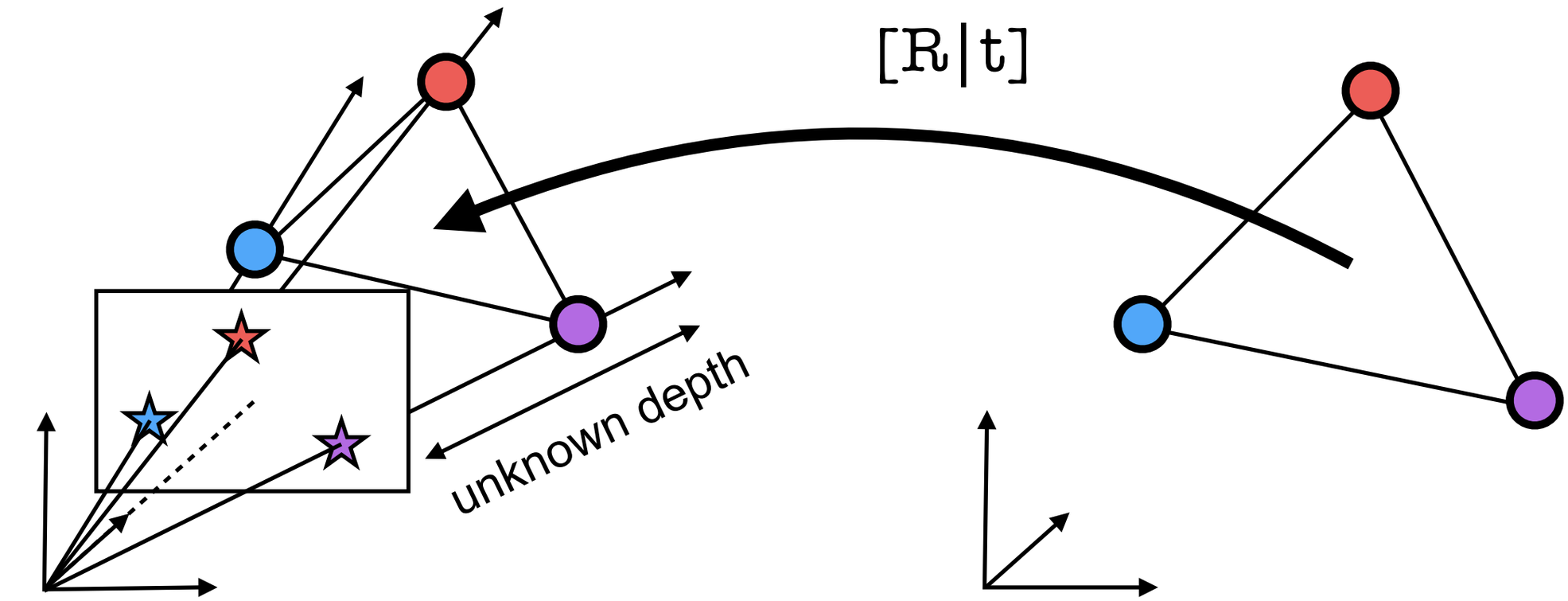
- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]
- Recover pose by aligning local and global point positions

3-Point Pose Problem (P3P)



- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]
- Recover pose by aligning local and global point positions
- Very efficient: ~2μs total [Kneip et al., CVPR'11] [code]

3-Point Pose Problem (P3P)



- **Case**: Intrinsic calibration known [Haralick et al., ICVJ'94]
- Recover depths: Solve 4th degree univariate polynomial [Fischler, Bolles, CACM'91]
- Recover pose by aligning local and global point positions
- Very efficient: ~2µs total [Kneip et al., CVPR'11] [code]
- Up to four solutions: Disambiguate using 4th point

Unknown Internal Calibration

- **Case:** Focal length unknown **(P4Pf)** [Bujnak et al., CVPR'08] [code]
  - Requires 4 matches, leads to system of **multivariate** polynomials
  - Recover variables as Eigenvectors of 10×10 matrix
  - Slow to compute (~45μs [Bujnak et al., CVPR'12])
    - Too slow to apply inside RANSAC loop
  - Multiple solutions: Disambiguate using 5th point

Unknown Internal Calibration

- **Case:** Focal length unknown **(P4Pf)** [Bujnak et al., CVPR'08] [code]
  - Requires 4 matches, leads to system of **multivariate** polynomials
  - Recover variables as Eigenvectors of 10×10 matrix
  - Slow to compute (~45µs [Bujnak et al., CVPR'12])
    - Too slow to apply inside RANSAC loop
  - Multiple solutions: Disambiguate using 5th point

- Recover full projection matrix from 6 matches (**P6P**) [Hartley, Zisserman, 2004]
  - Very efficient to compute: Solve 12×12 linear system
  - Single solution!
  - Can obtain least-squares solutions using >6 matches
  - In practice: RANSAC+P6P faster than RANSAC+P4Pf

Overview

- Basics
- Localization using Prioritized Matching
- Scalable Image-Based Localization
- Open Challenges / Conclusion

Main Challenges for Localization

- **Efficiency**: Quickly localize query images

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

- **Effectiveness**: Localize all query images

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

- **Effectiveness**: Localize all query images
  - Find enough good matches to facilitate pose estimation

Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

- **Effectiveness**: Localize all query images
  - Find enough good matches to facilitate pose estimation

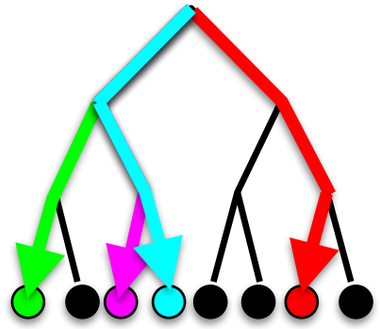
Main Challenges for Localization

- **Efficiency**: Quickly localize query images
  - **Efficient matching**
  - Efficient pose estimation = Find mostly correct matches

- **Effectiveness**: Localize all query images
  - Find enough good matches to facilitate pose estimation

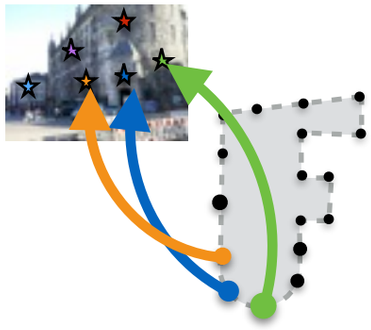
- **Accuracy**: Accurately recover camera pose

Localization - Overview



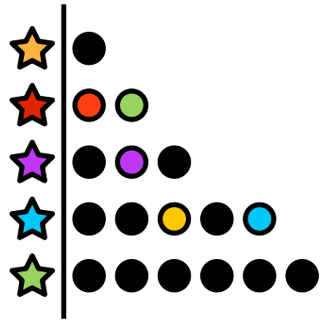
Baseline: kd-tree search

[Sattler et al., ICCV'11]



P2F

[Li et al., ECCV'10]



VPS

[Sattler et al., ICCV'11]

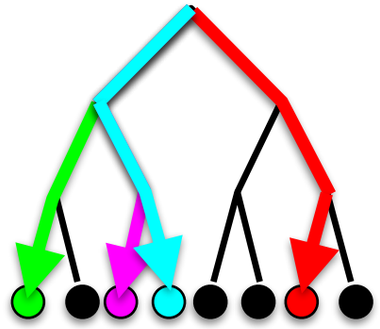


Active Search

[Sattler et al., ECCV'12]

effectivness	✓✓	✗✗	✗	✓✓✓
efficiency	✗✗	✓	✓	✓✓

Localization - Overview

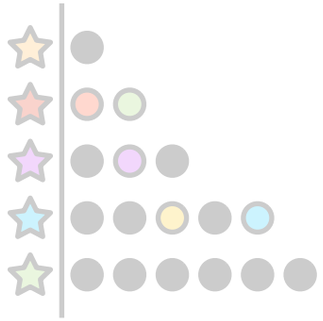


Baseline:
kd-tree search

[Sattler et al., ICCV'11]

P2F

[Li et al., ECCV'10]



VPS

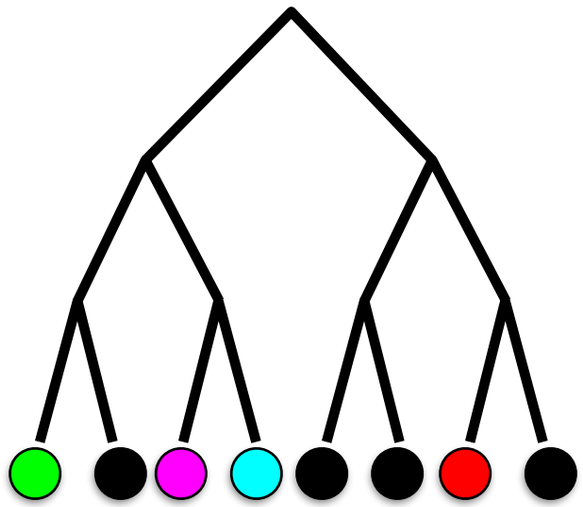
[Sattler et al., ICCV'11]

Active Search

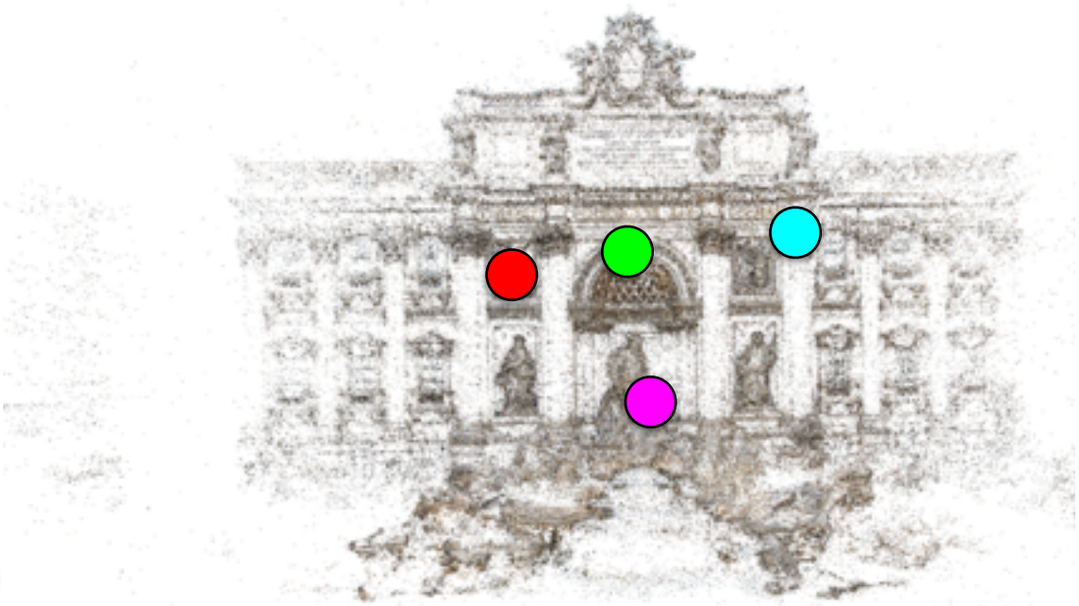
[Sattler et al., ECCV'12]

effectivness ✓✓	✗✗	✗	✓✓✓
efficiency ✗✗	✓	✓	✓✓

Baseline: Tree-Based Search



kd-tree



3D Model

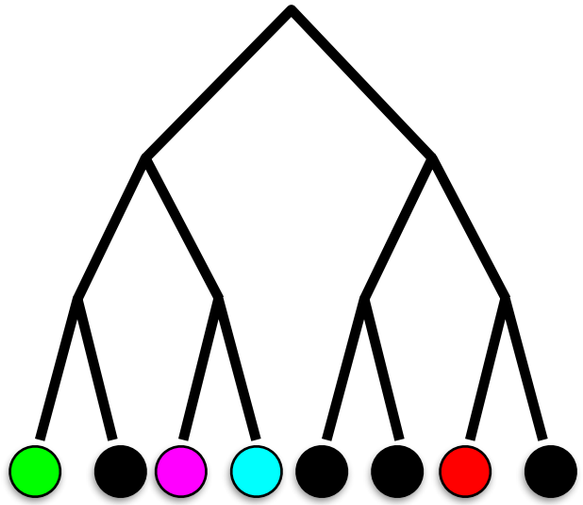
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

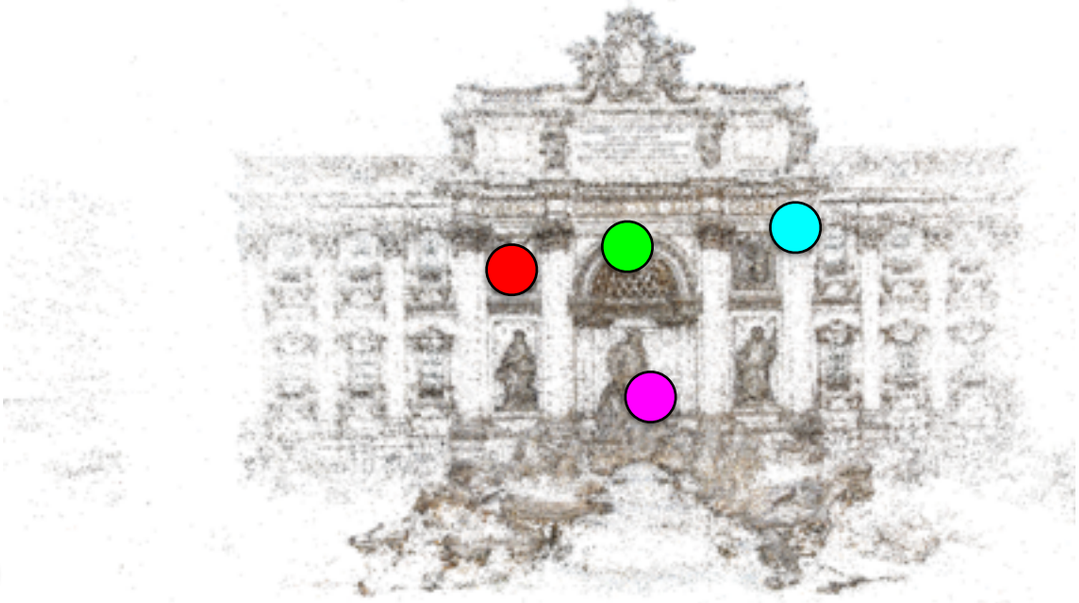
Baseline: Tree-Based Search



Query Image



kd-tree



3D Model

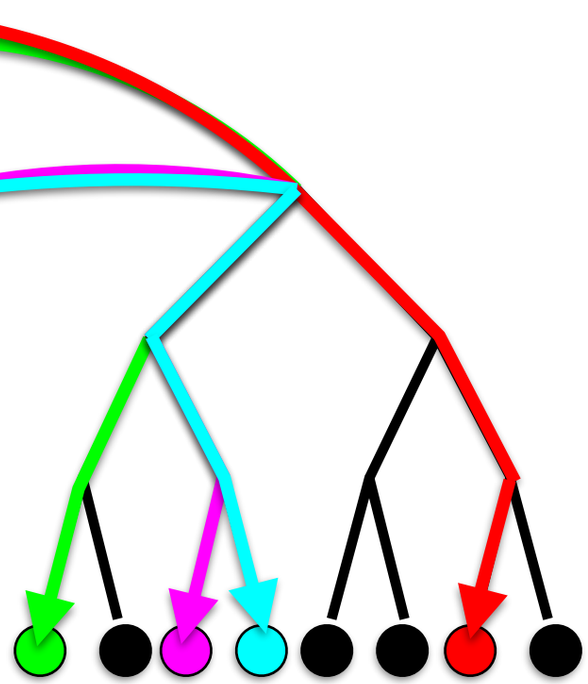
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

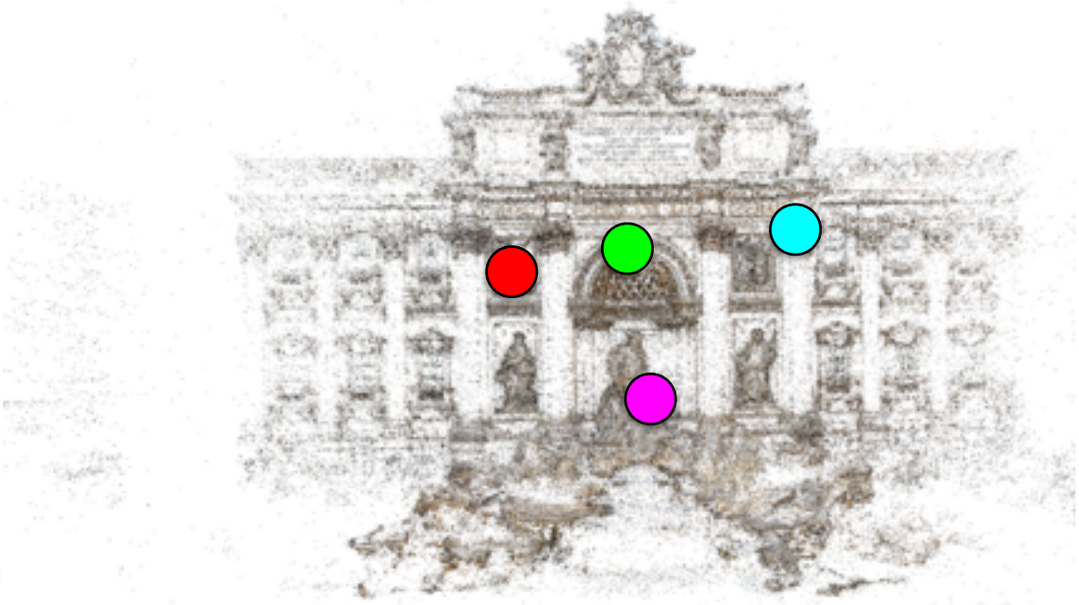
Baseline: Tree-Based Search



Query Image



kd-tree



3D Model

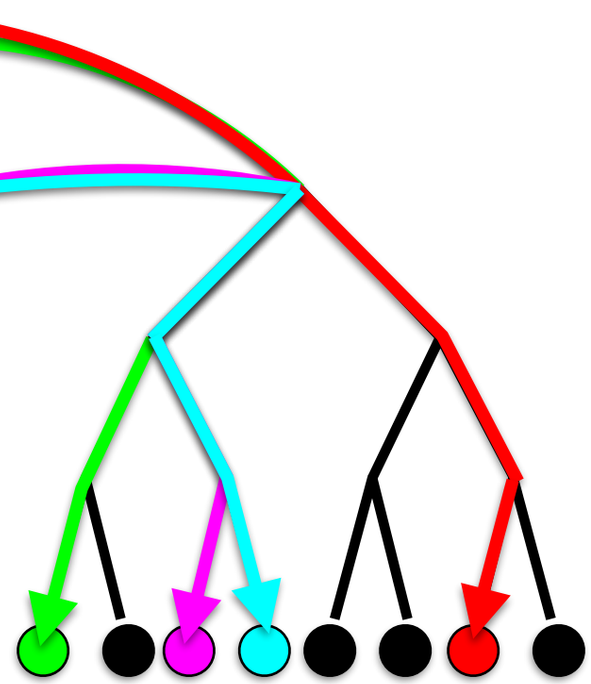
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

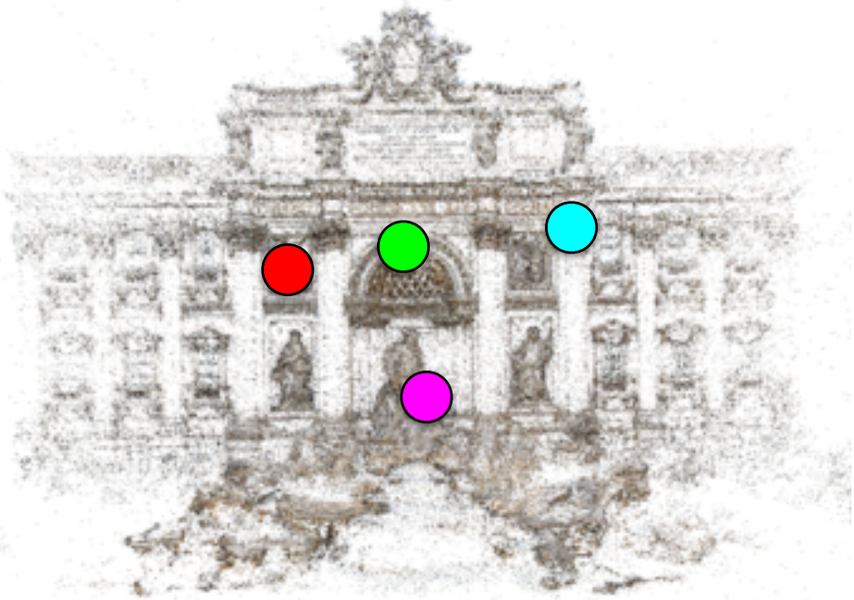
Baseline: Tree-Based Search



Query Image



kd-tree



3D Model

2-nn search & ratio test

$$\frac{||d - d_1||_2}{||d - d_2||_2} < 0.7$$

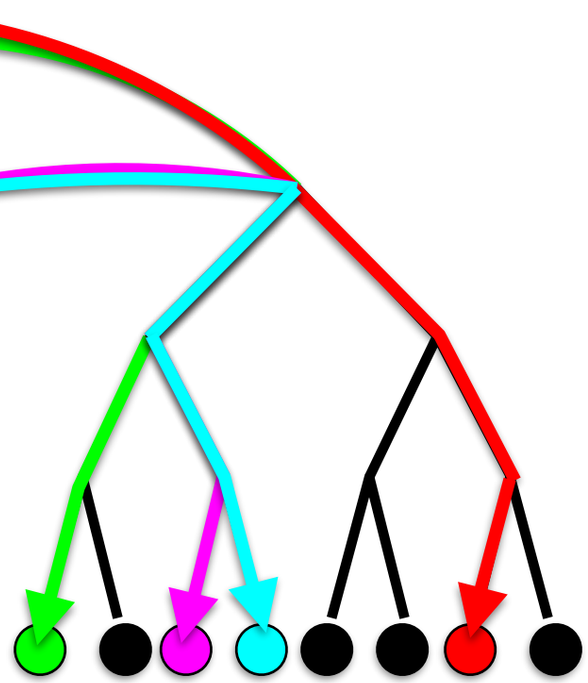
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

Baseline: Tree-Based Search



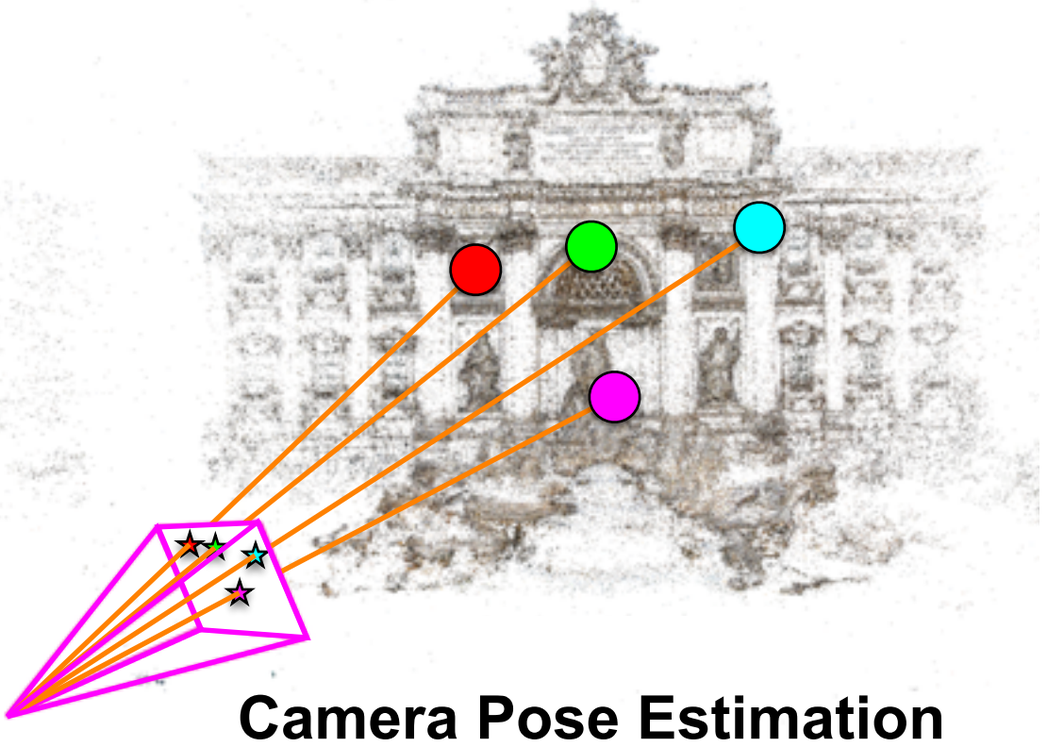
Query Image



kd-tree

2-nn search & ratio test

$$\frac{||d - d_1||_2}{||d - d_2||_2} < 0.7$$



Camera Pose Estimation

LO-RANSAC +$T_{1,1}$ + P6P

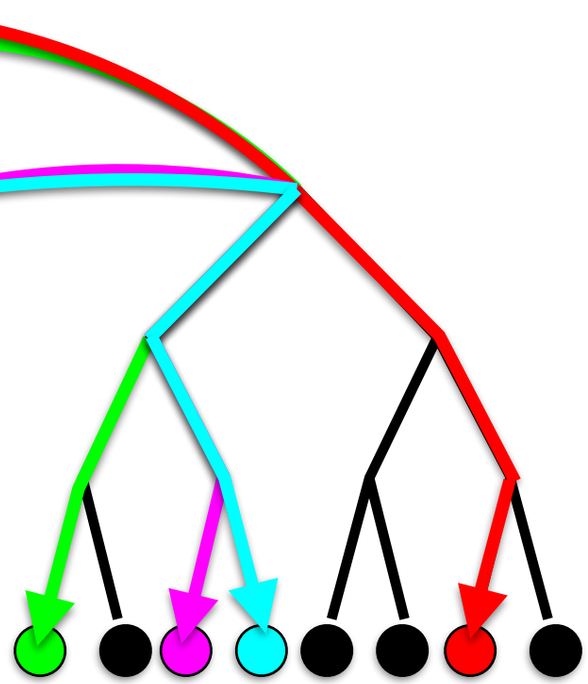
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

Baseline: Tree-Based Search



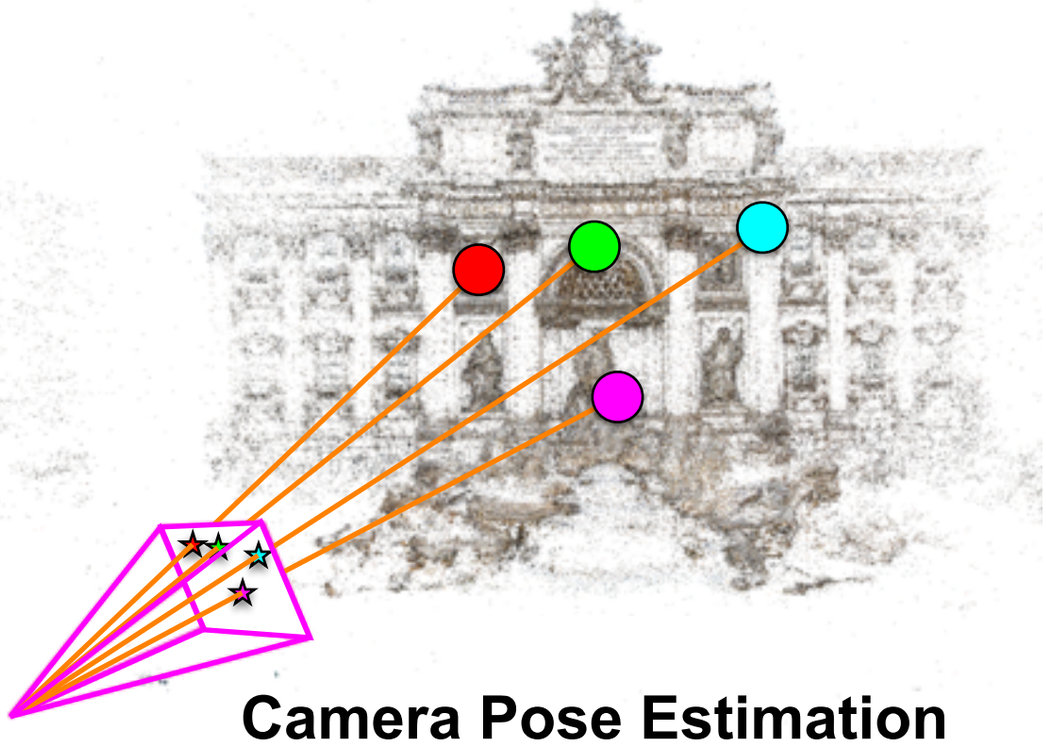
Query Image



kd-tree

2-nn search & ratio test

$$\frac{||d - d_1||_2}{||d - d_2||_2} < 0.7$$



Camera Pose Estimation

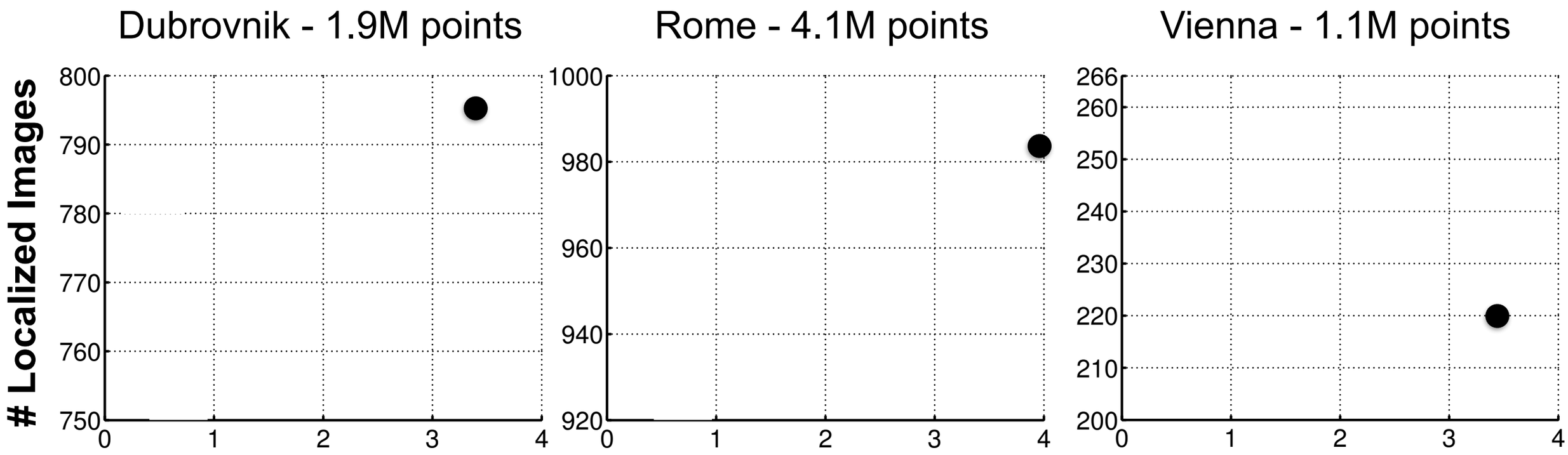
LO-RANSAC +$T_{1,1}$ + P6P

pose valid if ≥ 12 inliers

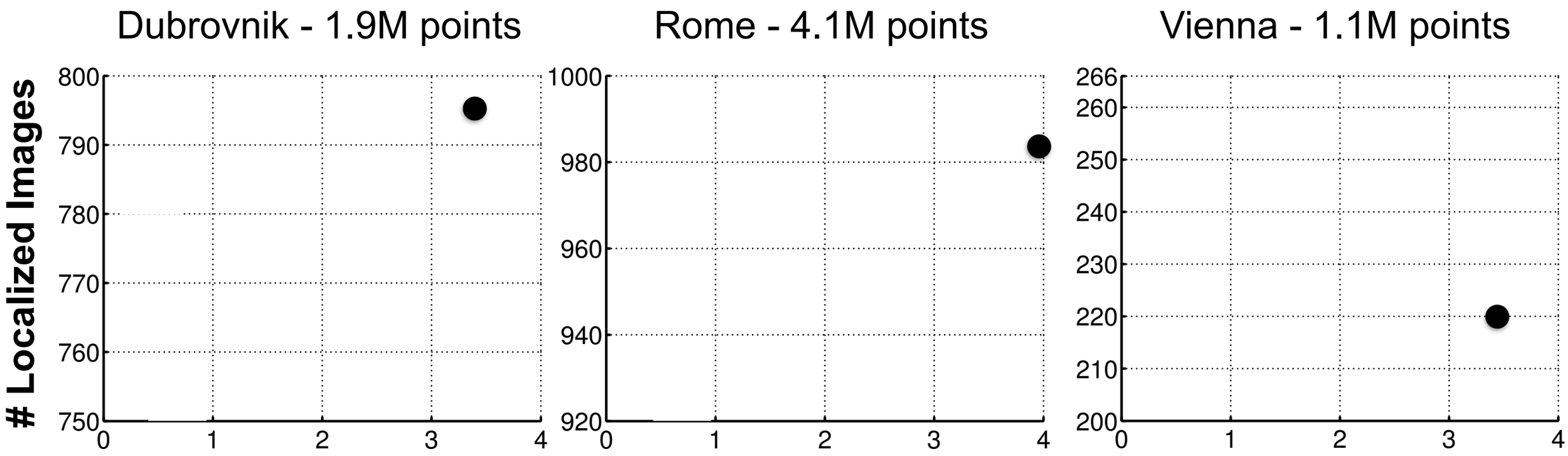
[Sattler et al., ICCV'11] [code]

photo credit: Ming Li

Results


Dubrovnik - 1.9M points
Rome - 4.1M points
Vienna - 1.1M points
Localized Images
800
790
780
770
760
750
1000
980
960
940
920
266
260
250
240
230
220
210
200
0
1
2
3
4
Mean localization time per image [s] (excluding feature extraction)
kd-tree

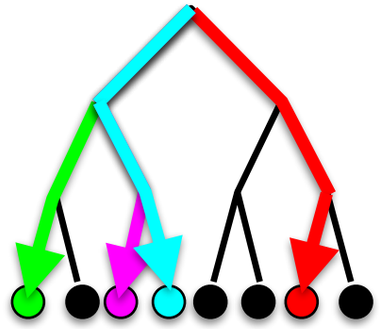
Results



✓Excellent localization effectiveness…

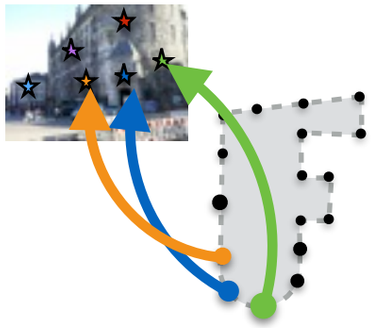
✗ … but very slow!

Localization - Overview



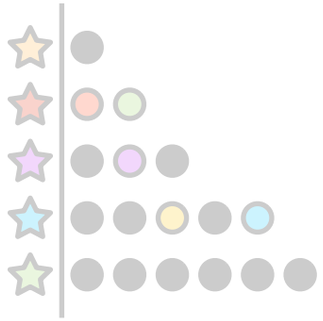
Baseline: kd-tree search

[Sattler et al., ICCV'11]



P2F

[Li et al., ECCV'10]



VPS

[Sattler et al., ICCV'11]

Active Search

[Sattler et al., ECCV'12]

effectivness ✓✓	**✗✗**	✗	✓✓✓
efficiency ✗✗	**✓**	✓	✓✓

Tree-Based Search Drawbacks

- Tree-based approach so slow because it

Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches

Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**

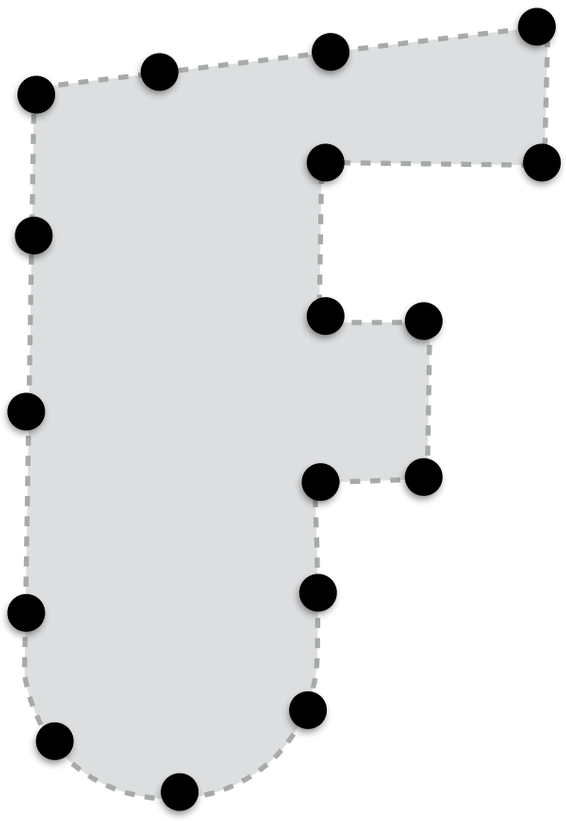
Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches

Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches

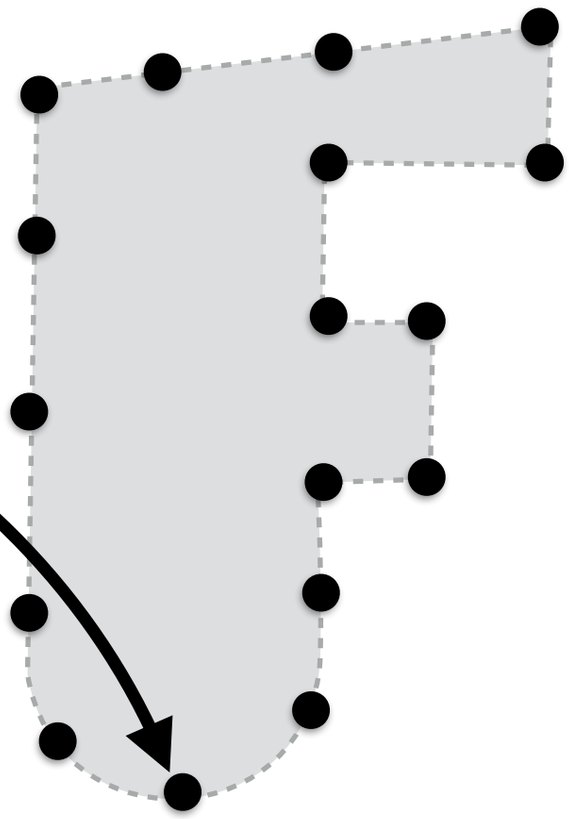




Tree-Based Search Drawbacks

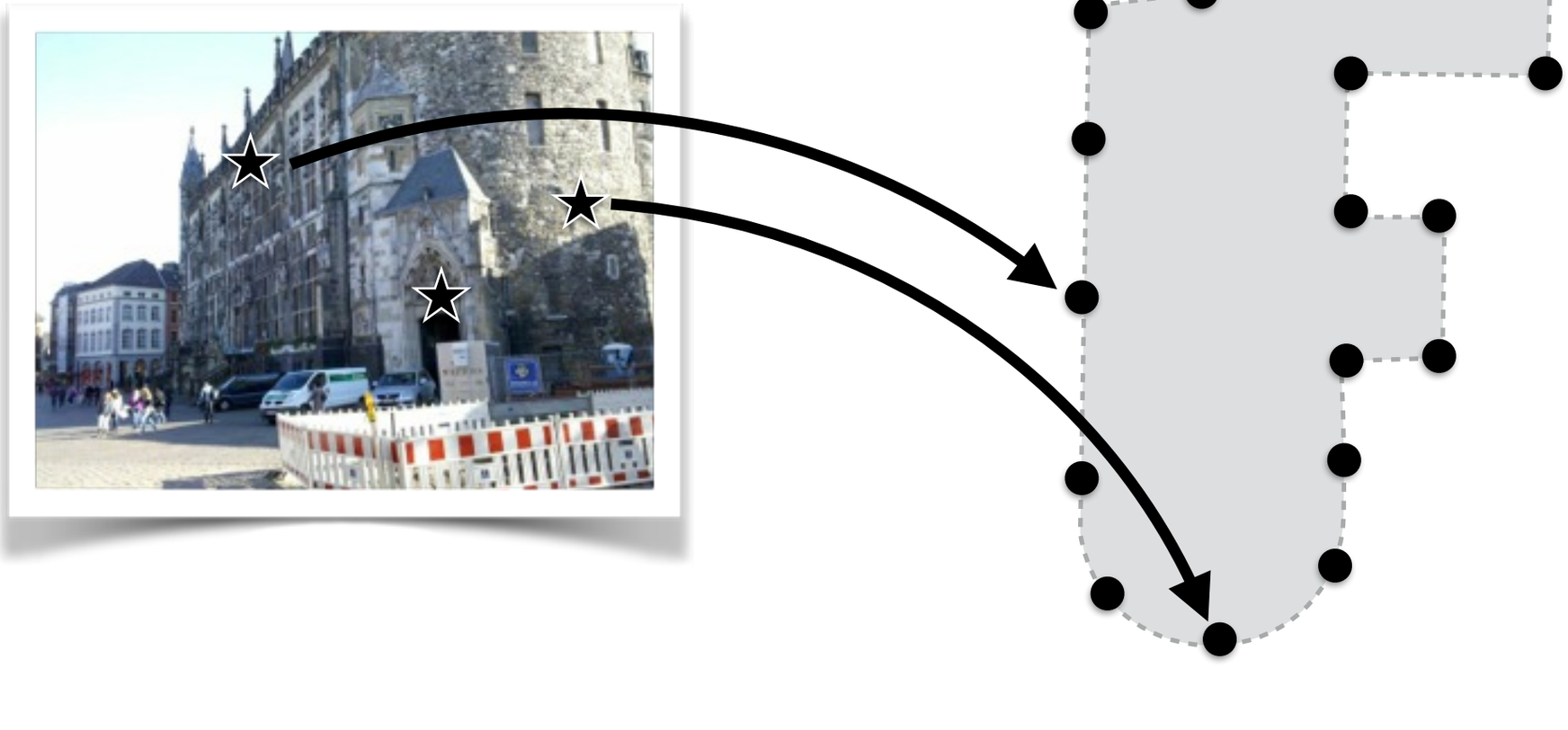
- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches





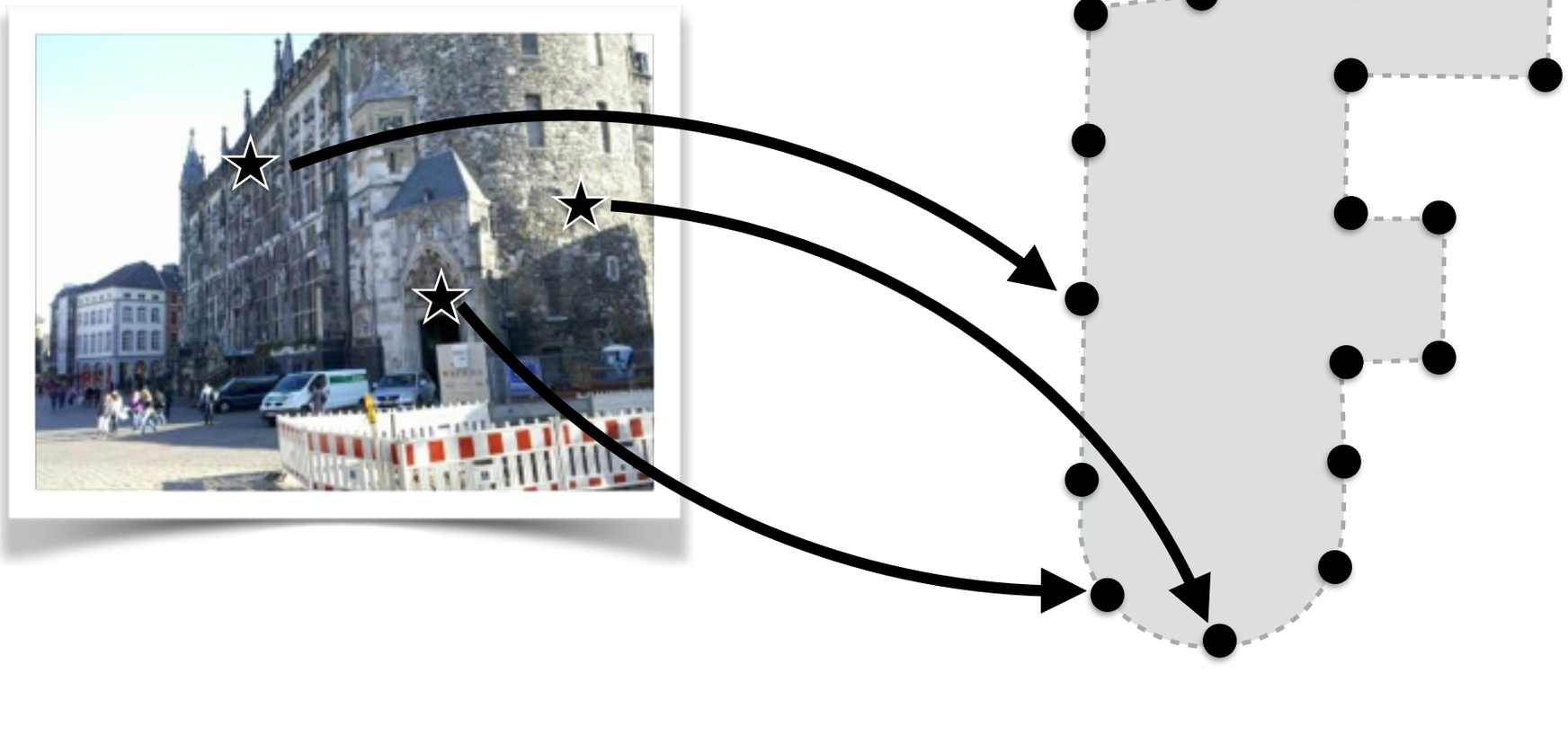
Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches



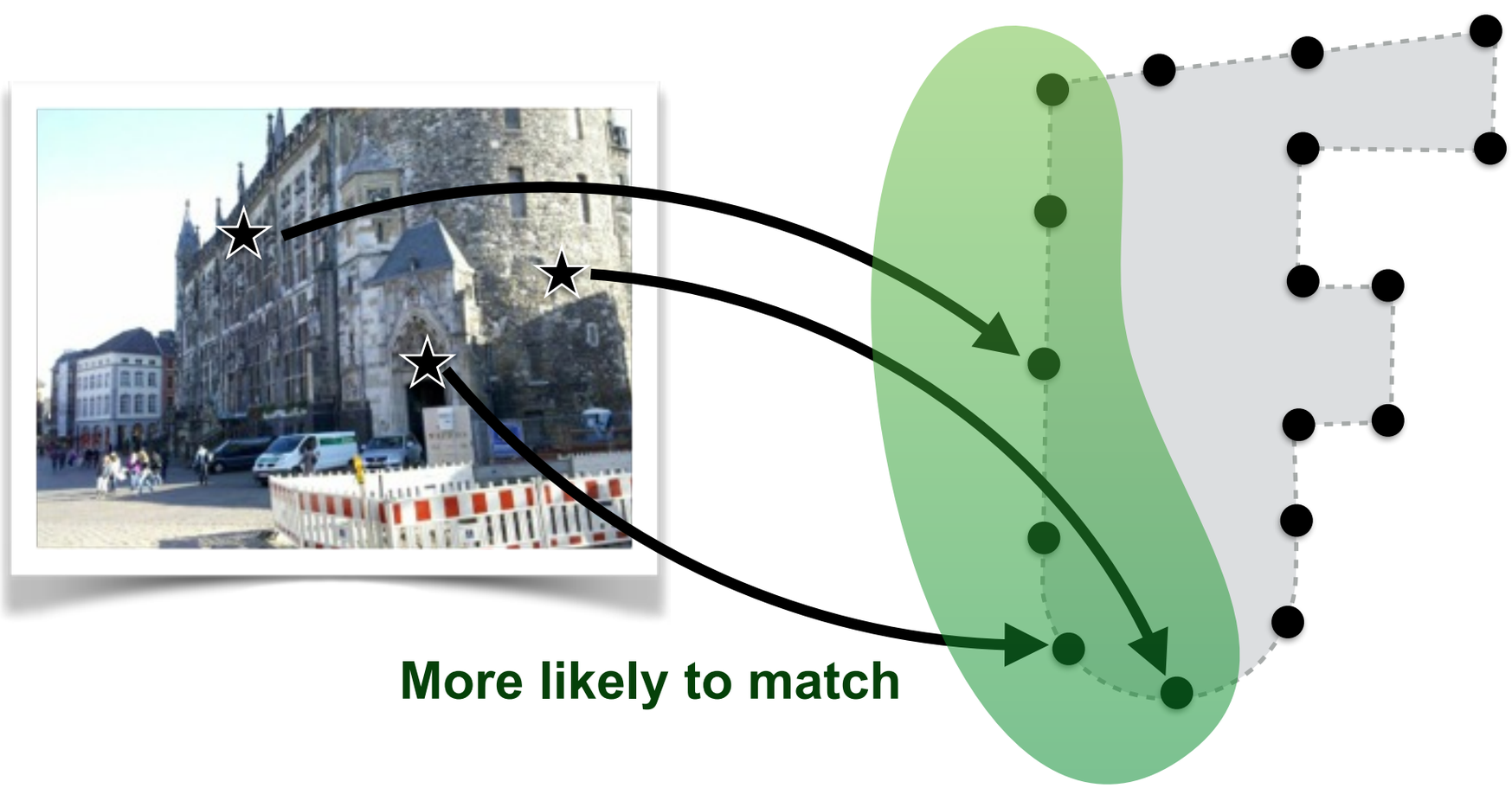
Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches



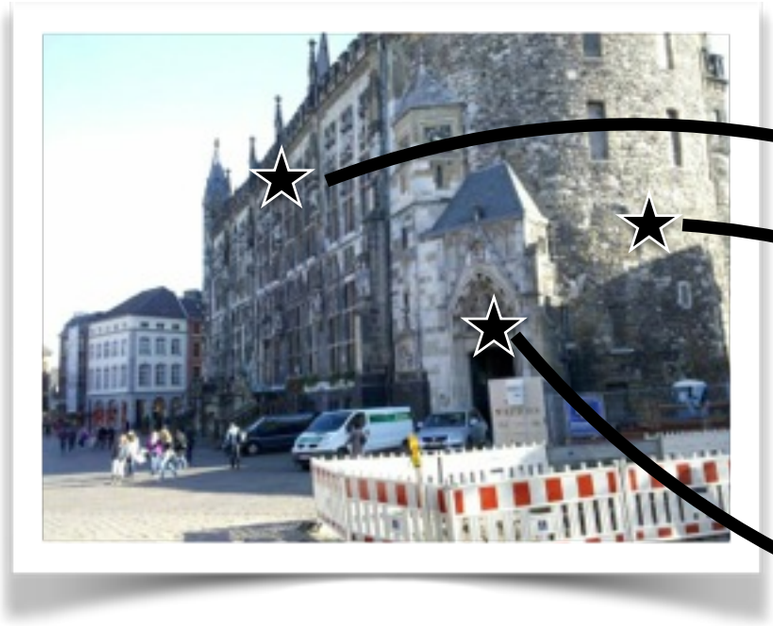
Tree-Based Search Drawbacks

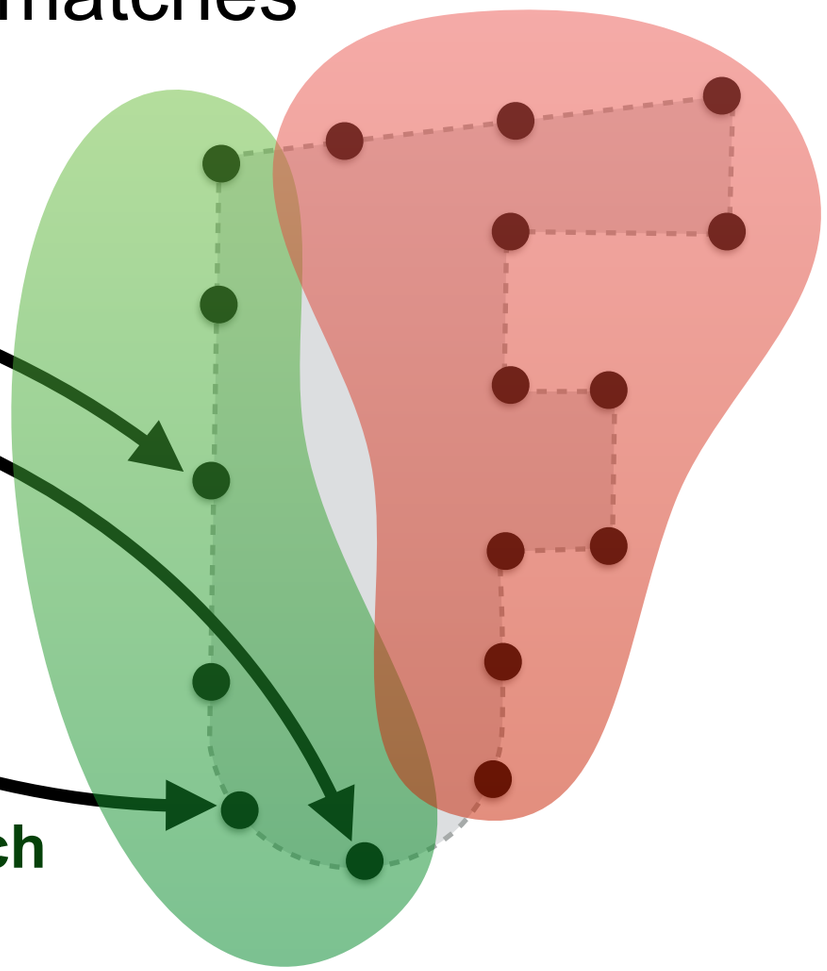
- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches



Tree-Based Search Drawbacks

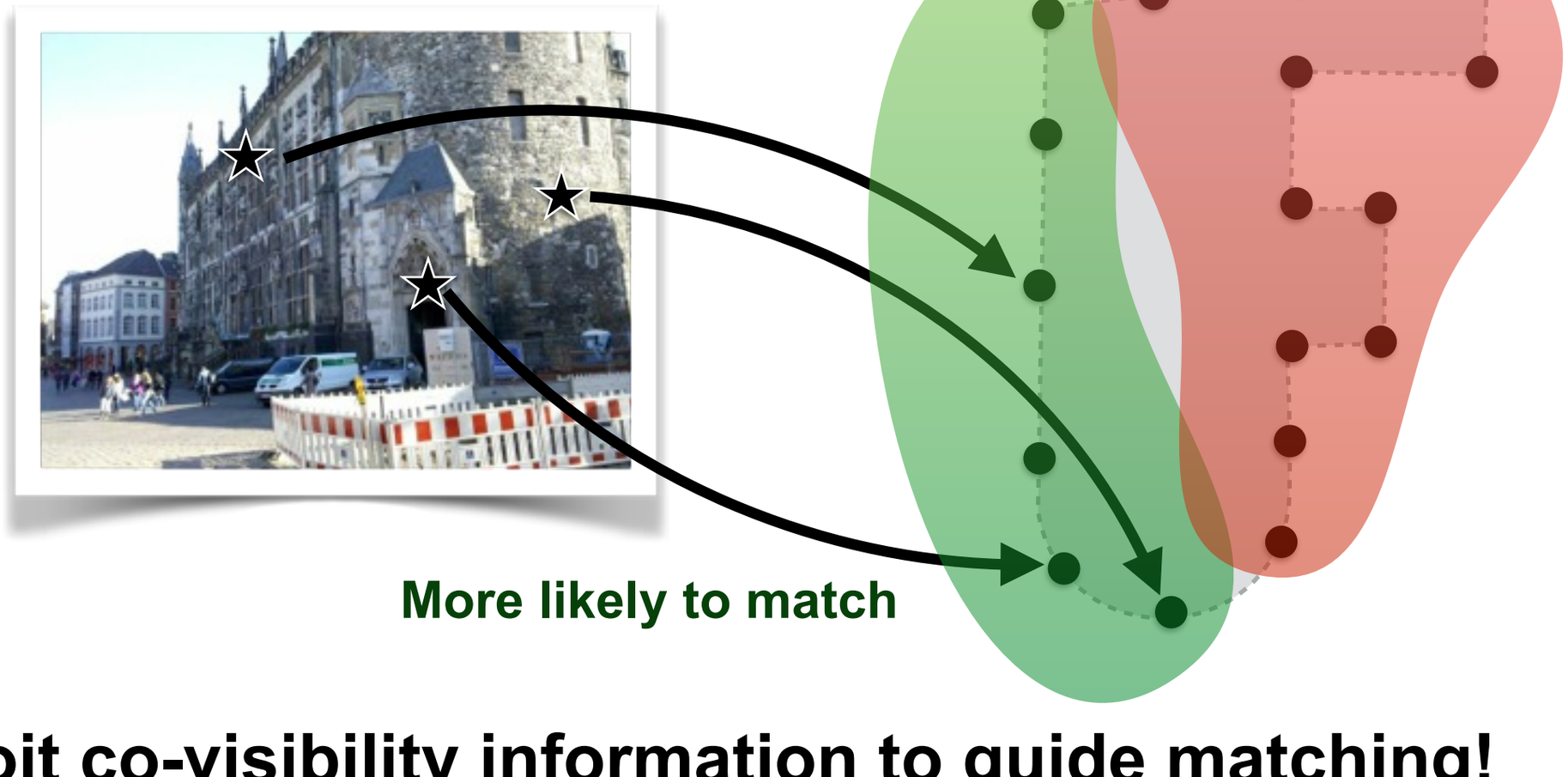
- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don't need all of them!**
  - … ignores dependencies between matches





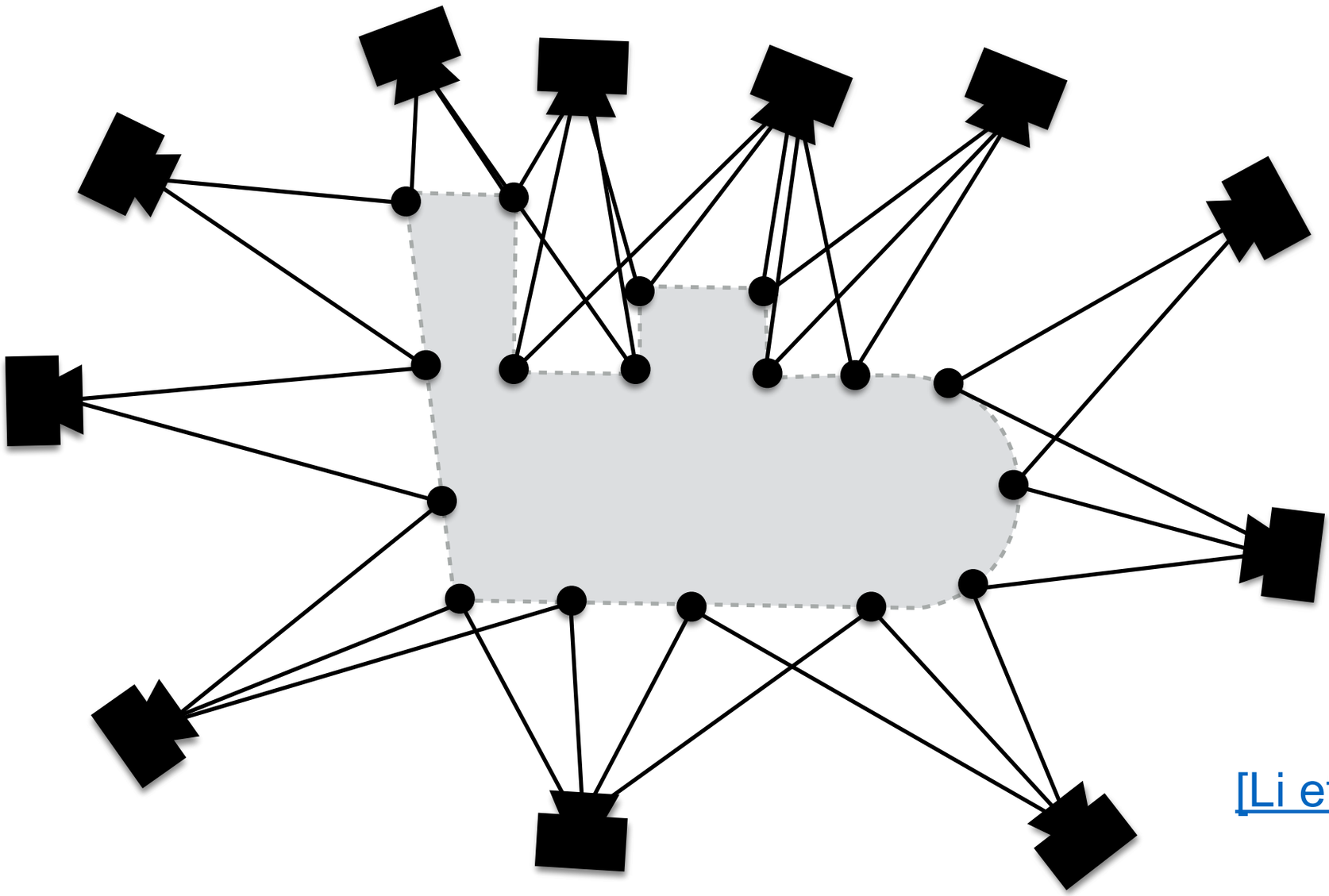
Tree-Based Search Drawbacks

- Tree-based approach so slow because it
  - … tries to find all possible matches
    - **Don’t need all of them!**
  - … ignores dependencies between matches



➡ **Exploit co-visibility information to guide matching!**

The Visibility Graph



[Li et al., ECCV'10]

- Bipartite visibility graph G_V defined by SfM reconstruction
- Two points co-visible if share a common camera

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching

Build kd-tree for query image



[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching

Build kd-tree for query image

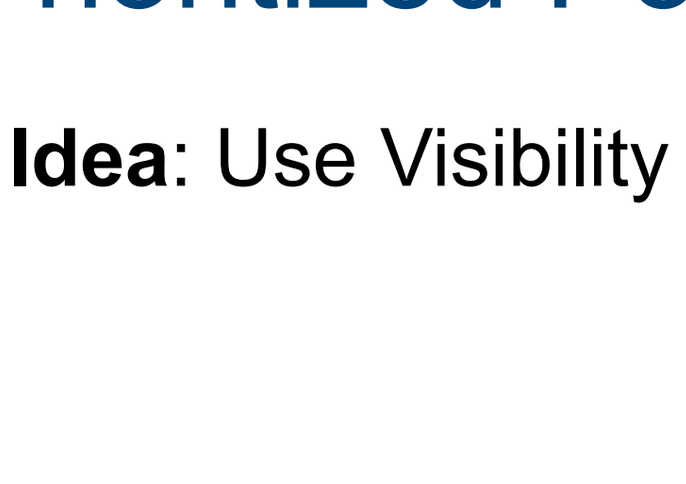
Prioritized 3D-to-2D Matching

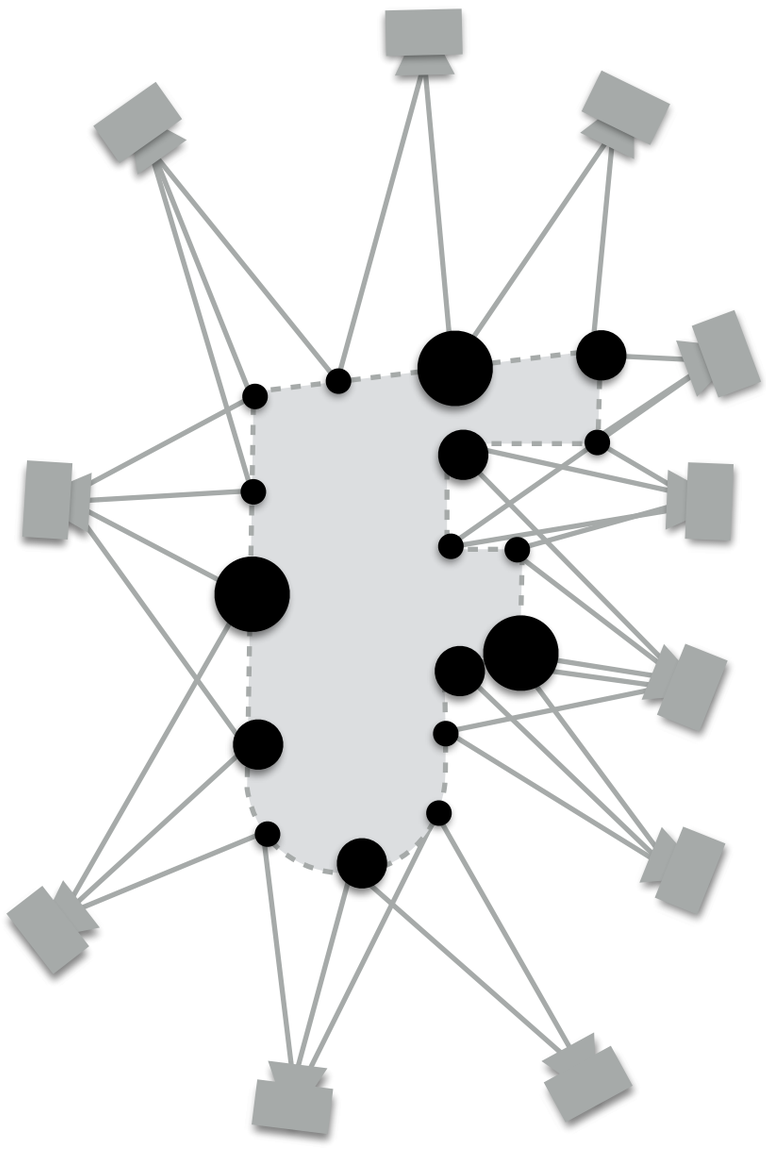


[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





Build kd-tree for query image

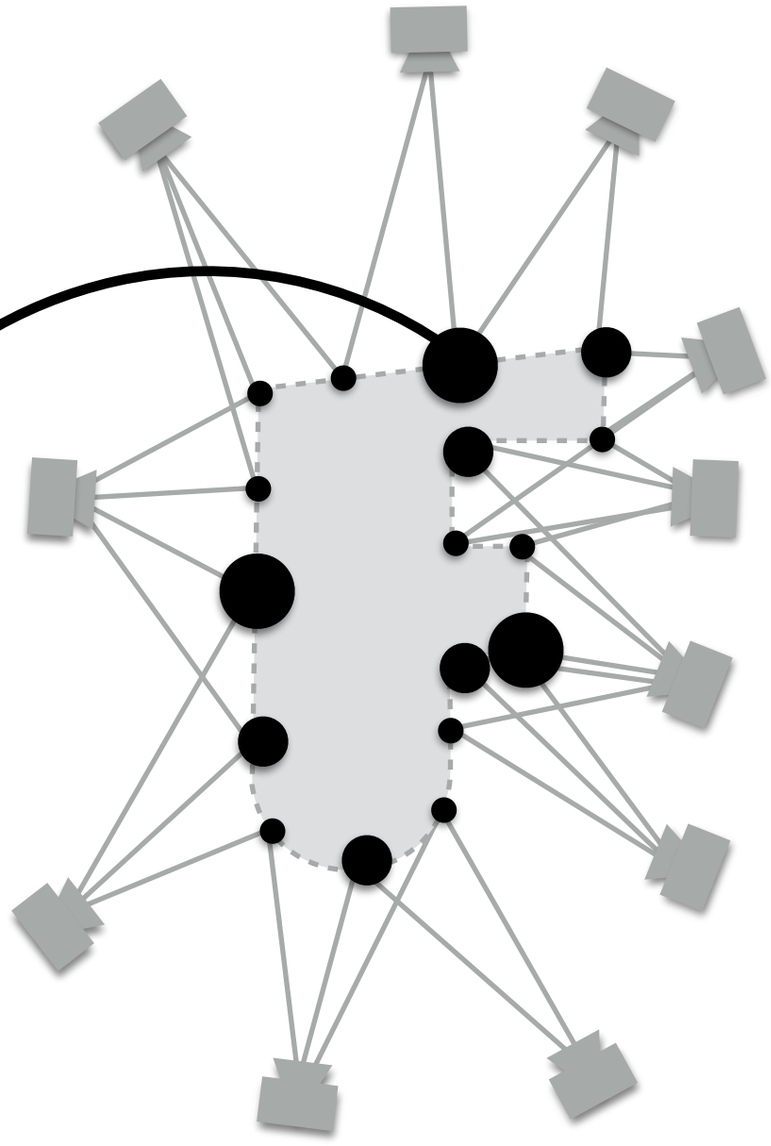
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





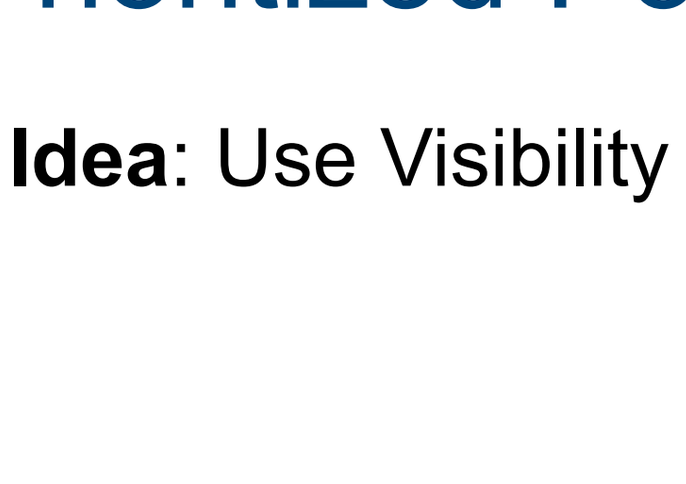
Build kd-tree for query image

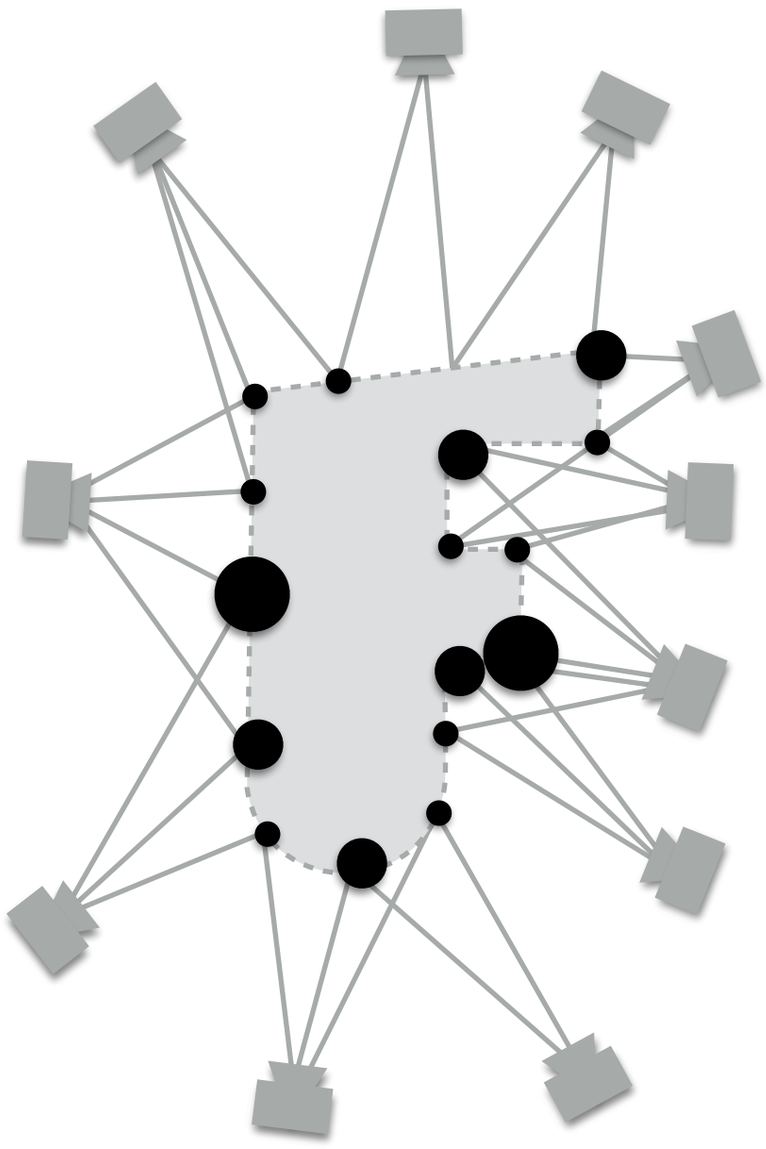
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





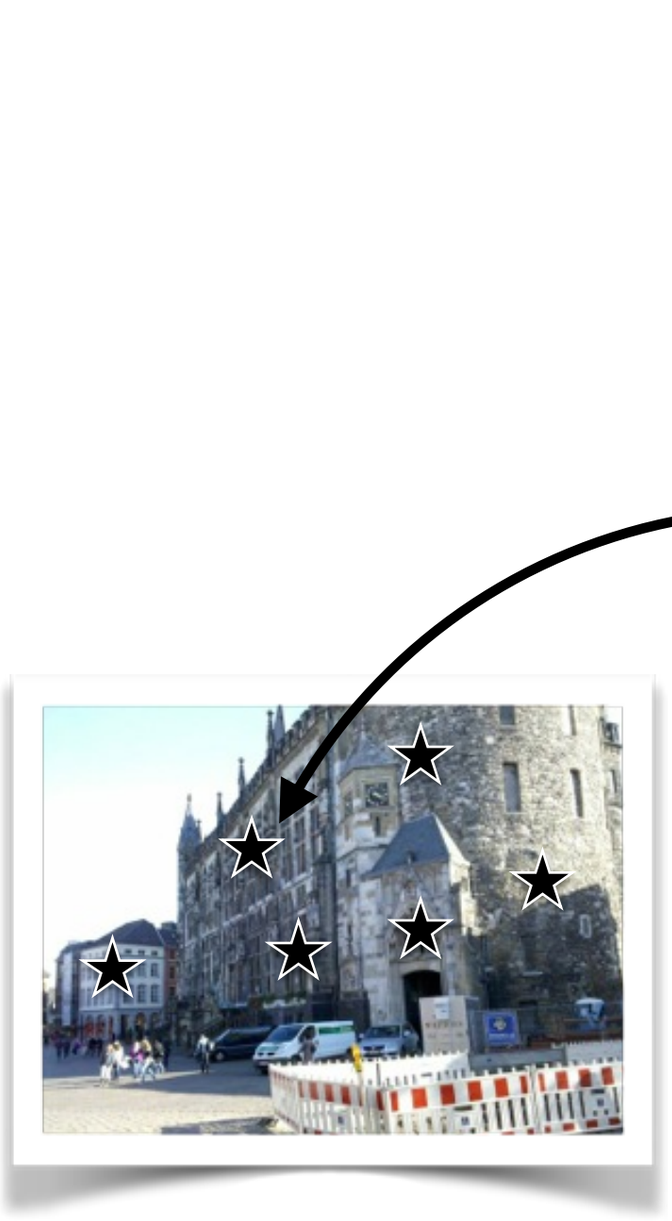
Build kd-tree for query image

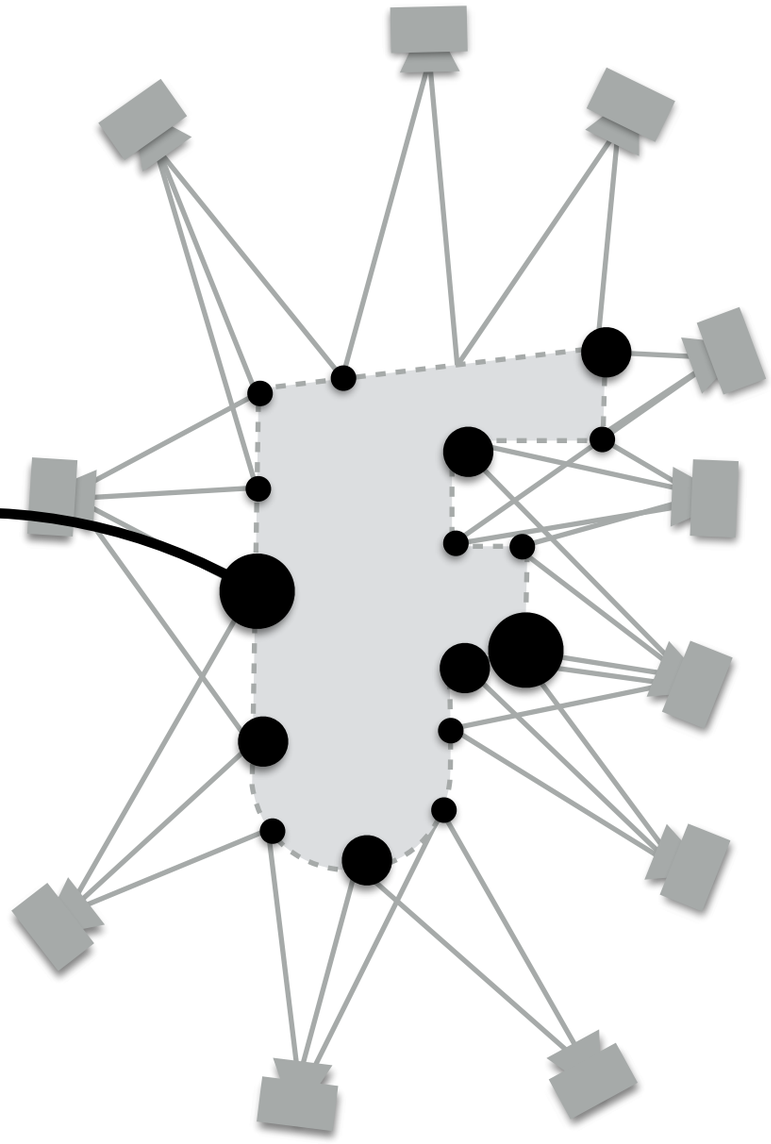
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





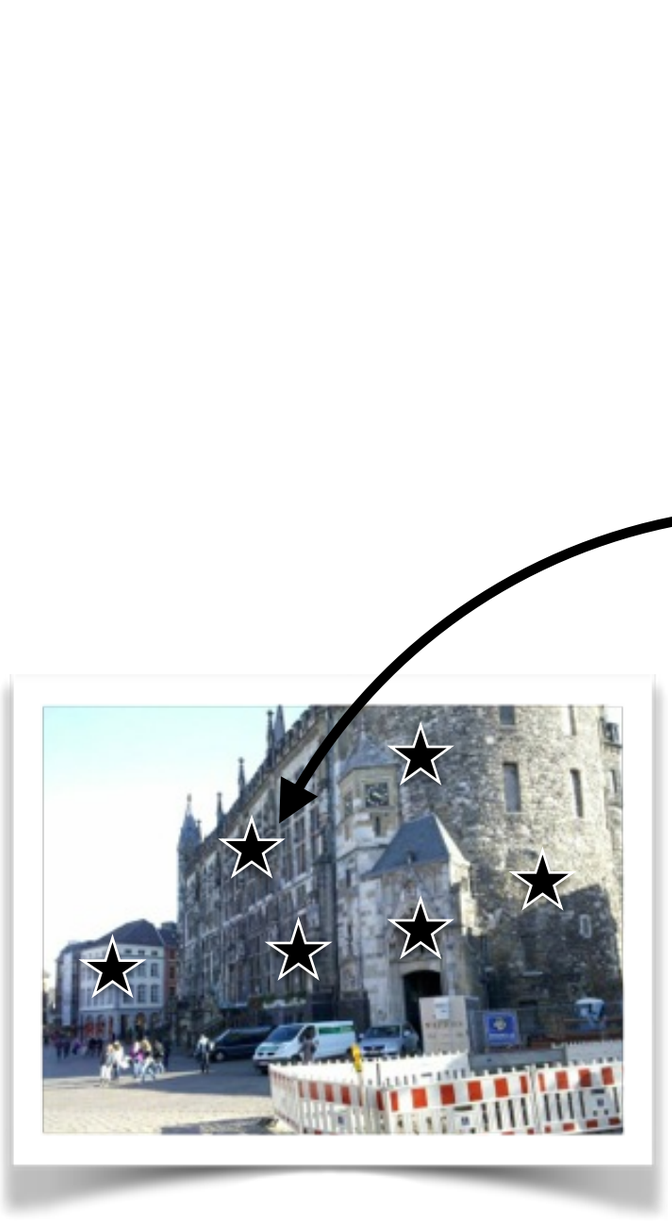
Build kd-tree for query image

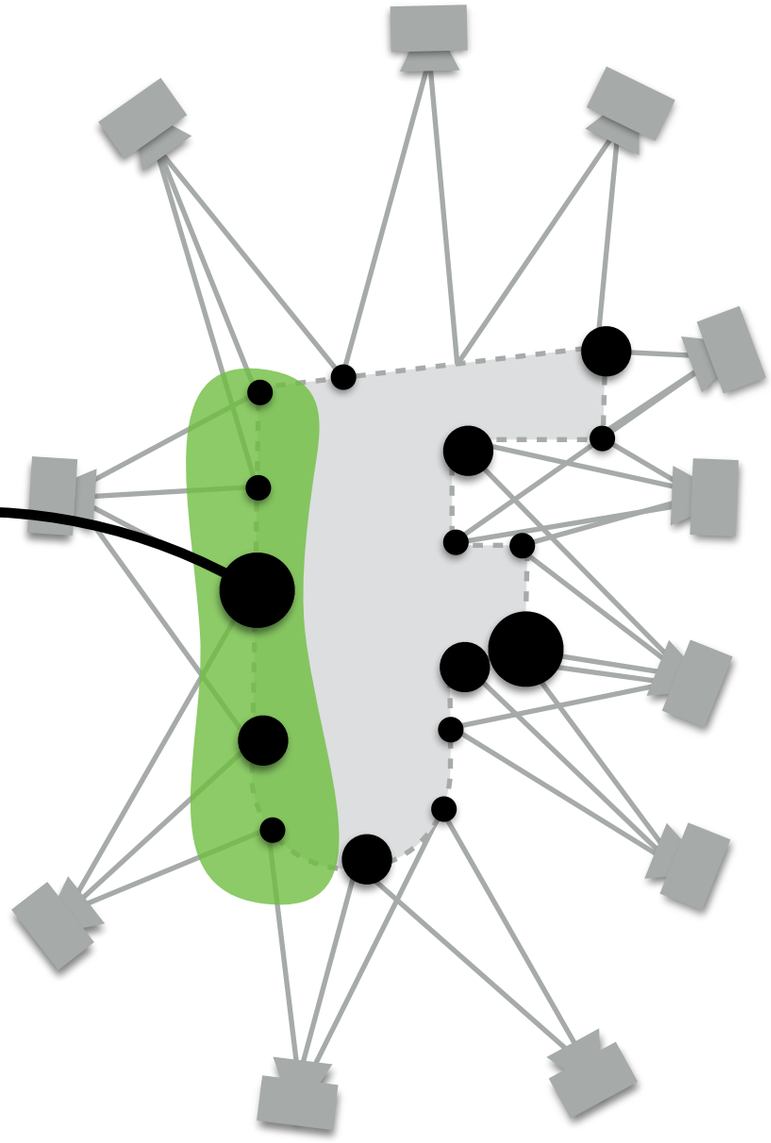
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





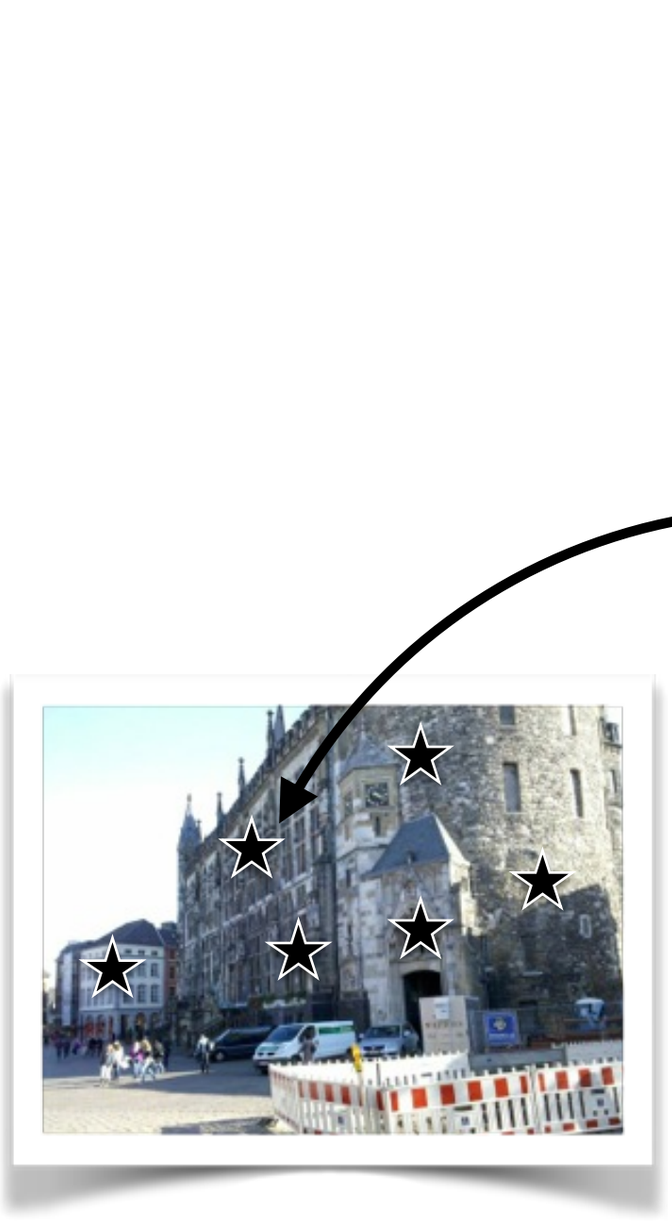
Build kd-tree for query image

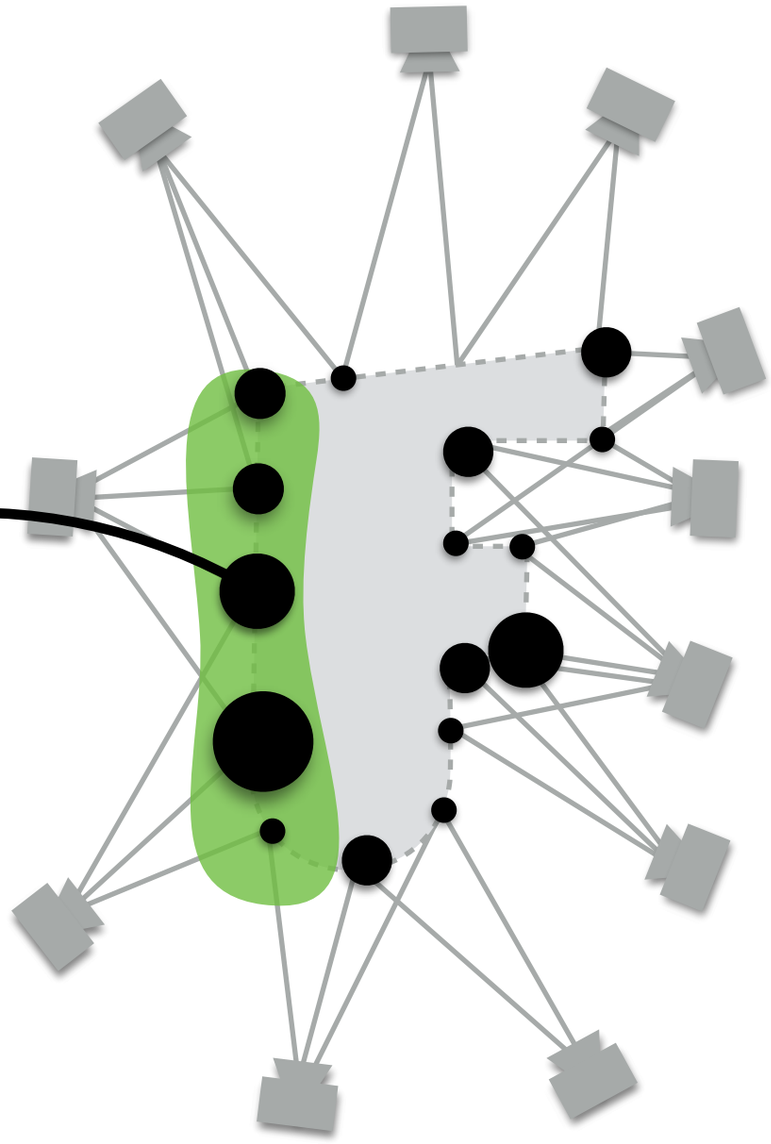
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





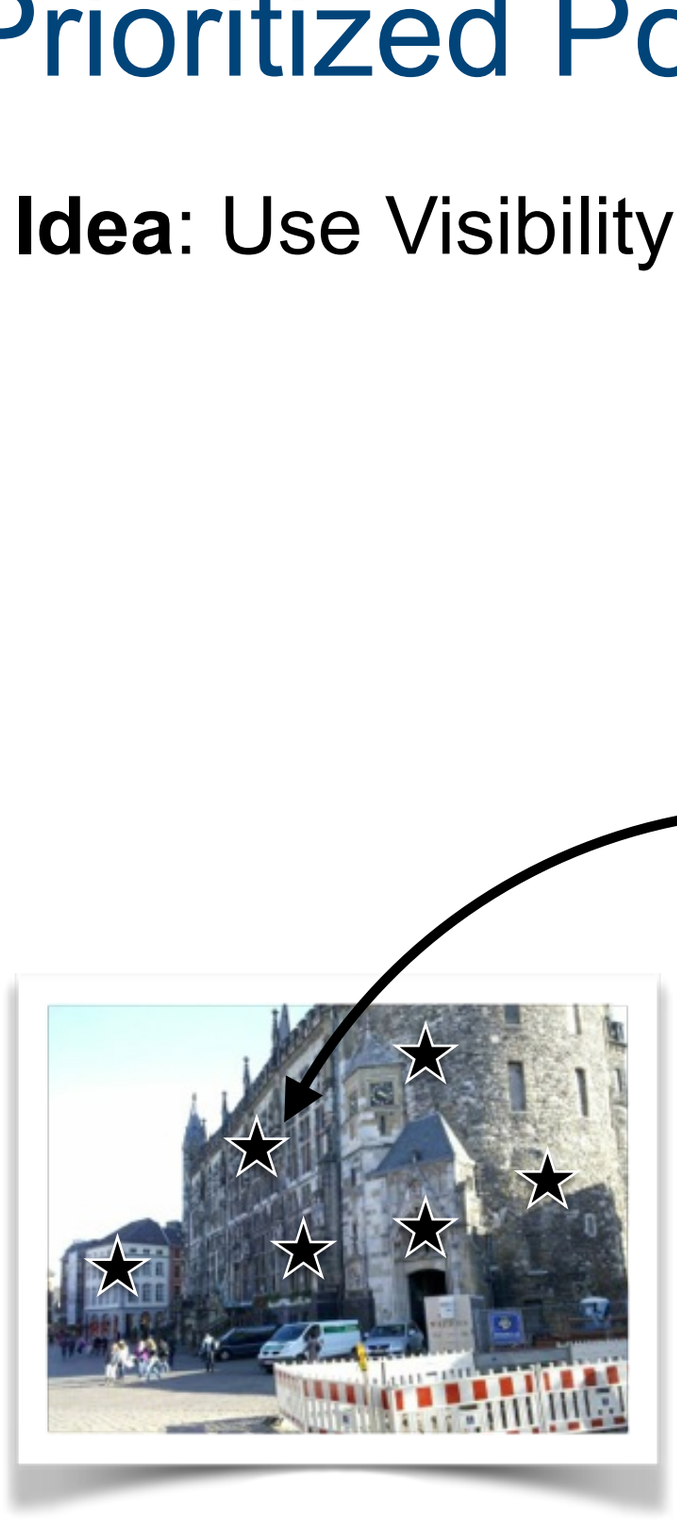
Build kd-tree for query image

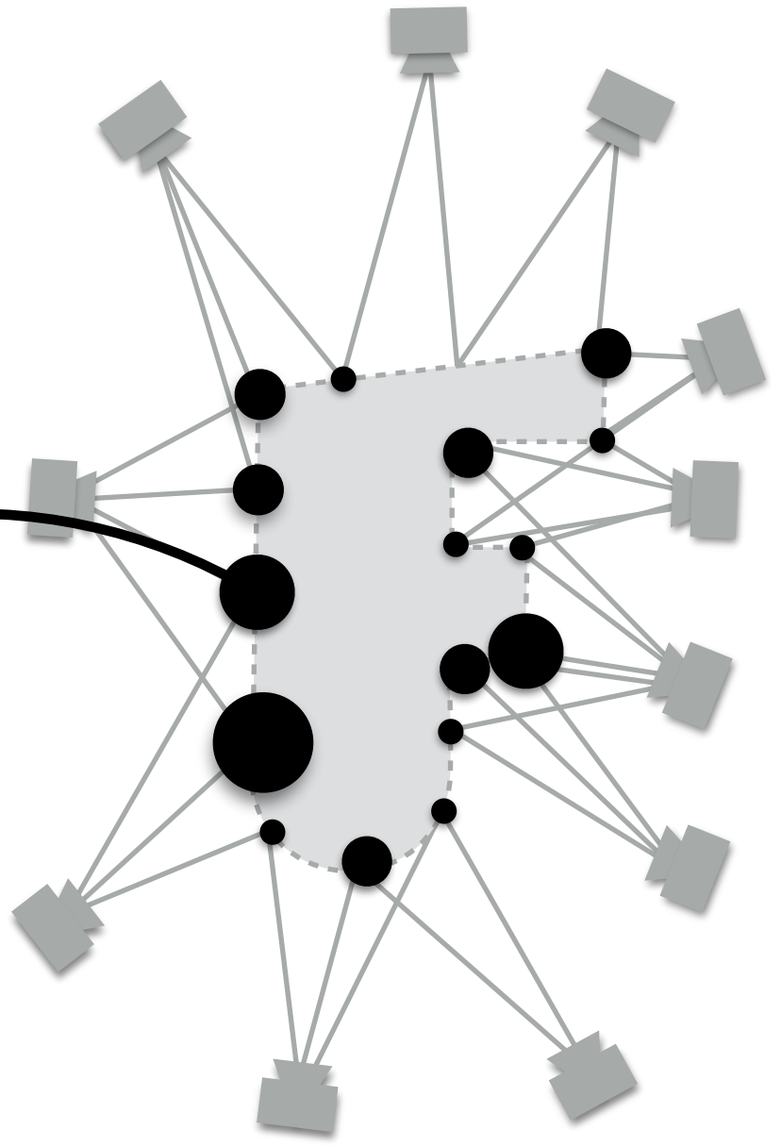
Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching





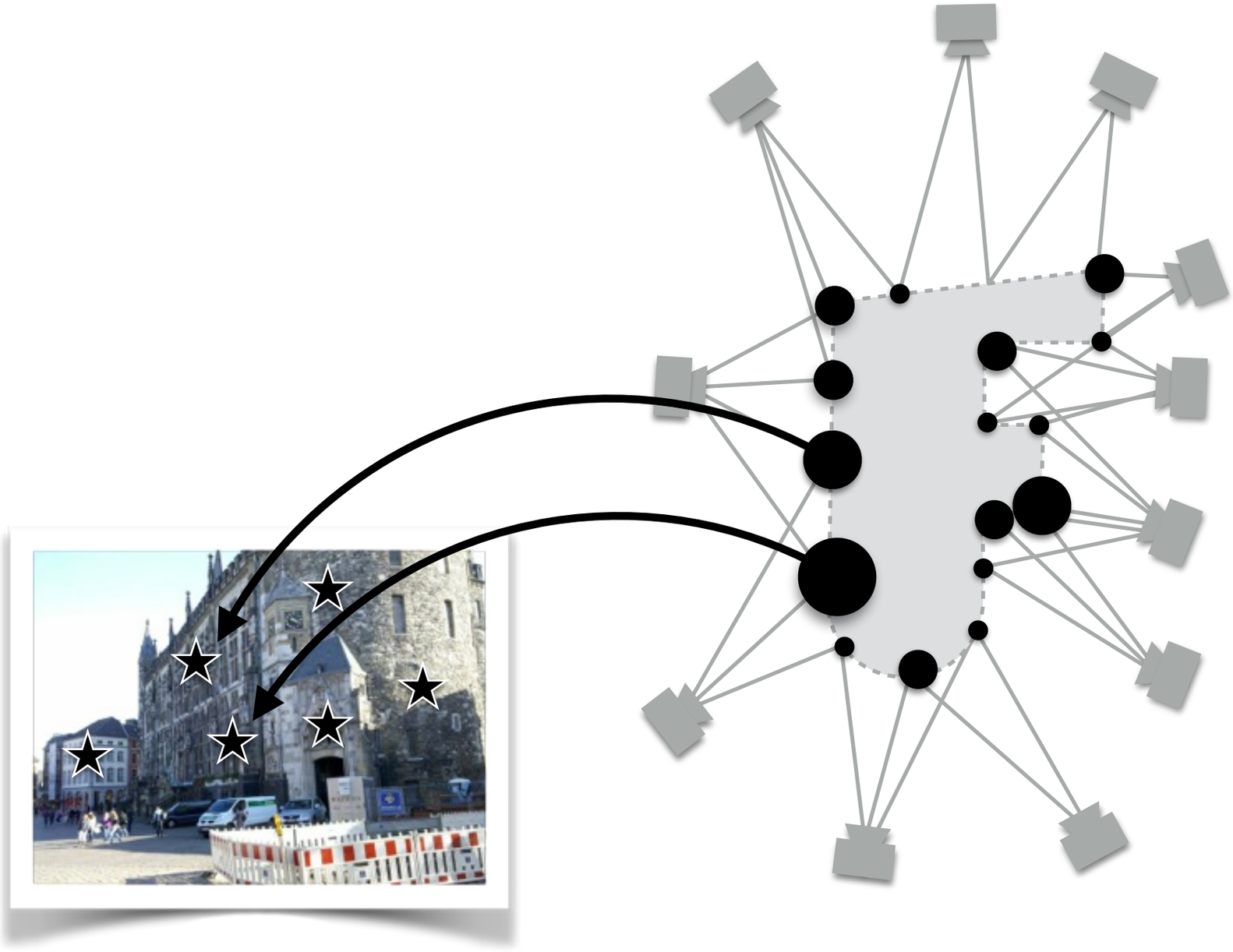
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



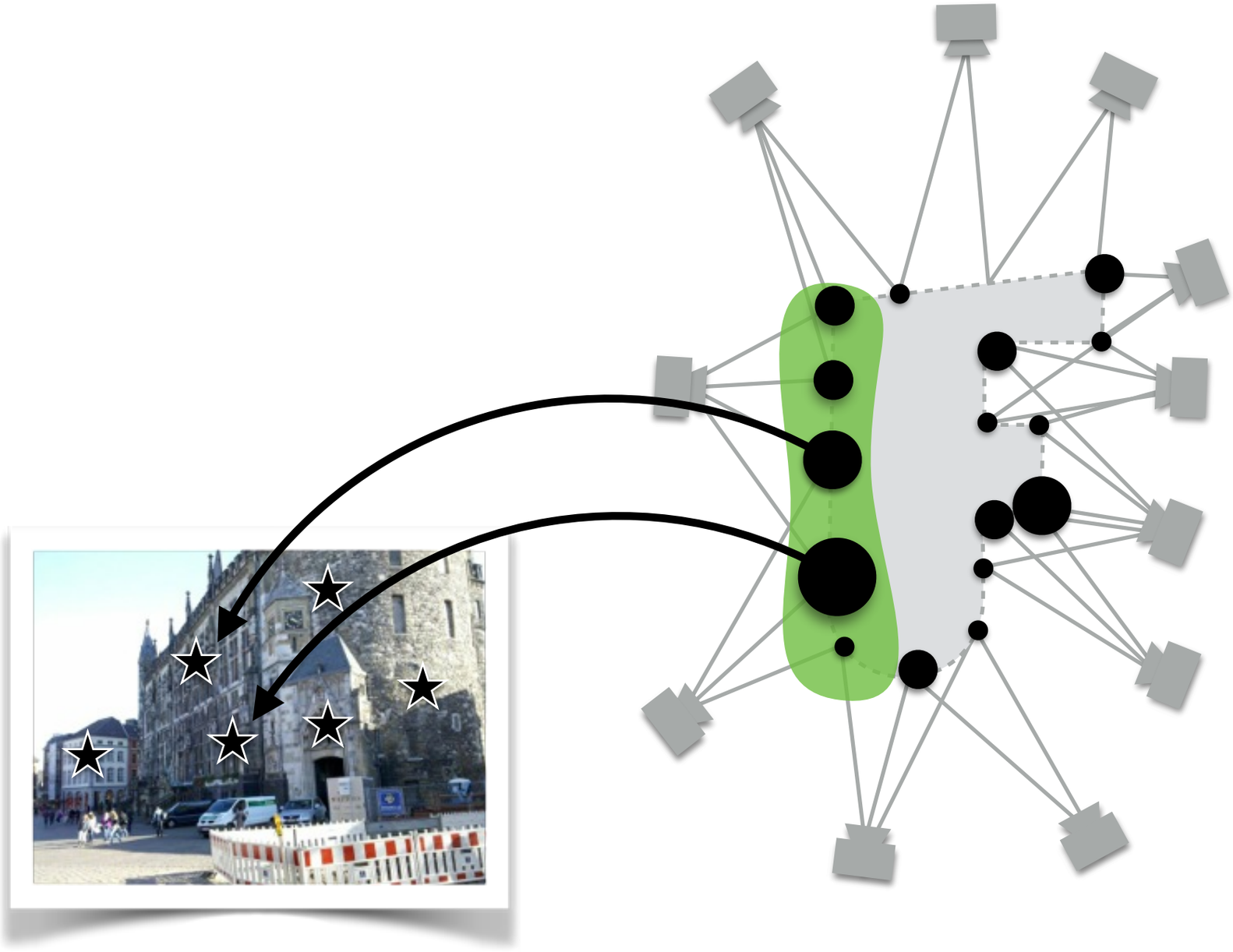
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



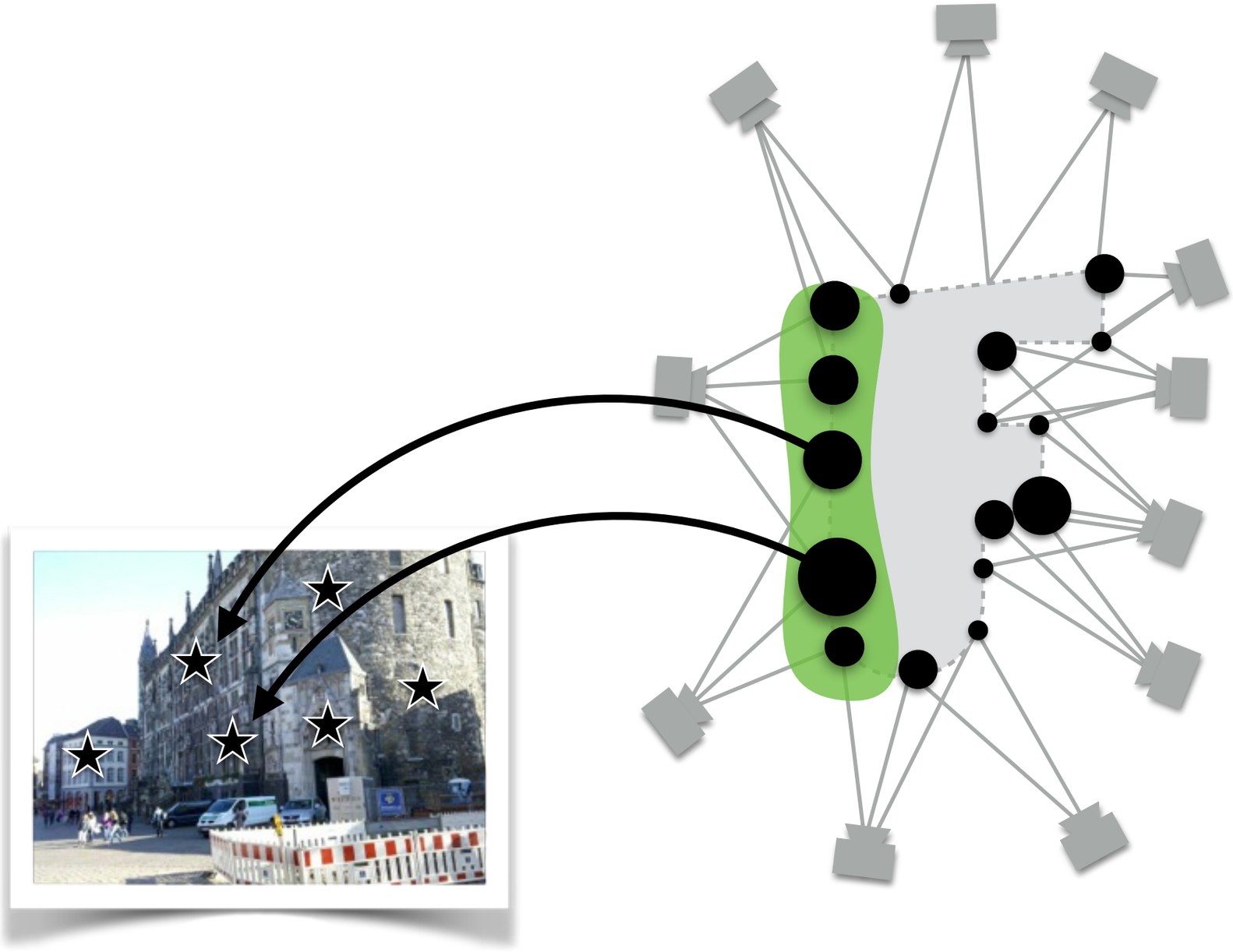
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



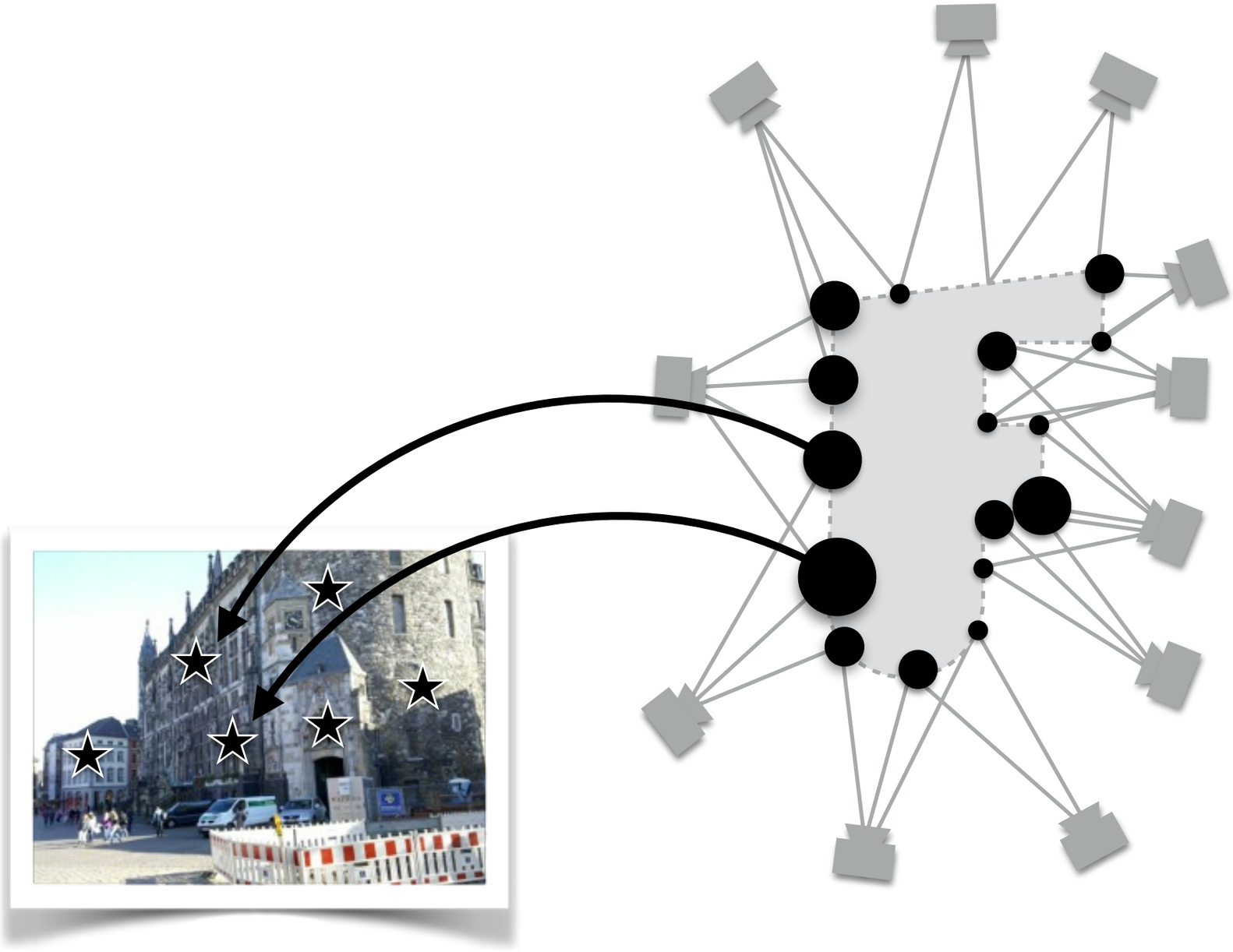
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



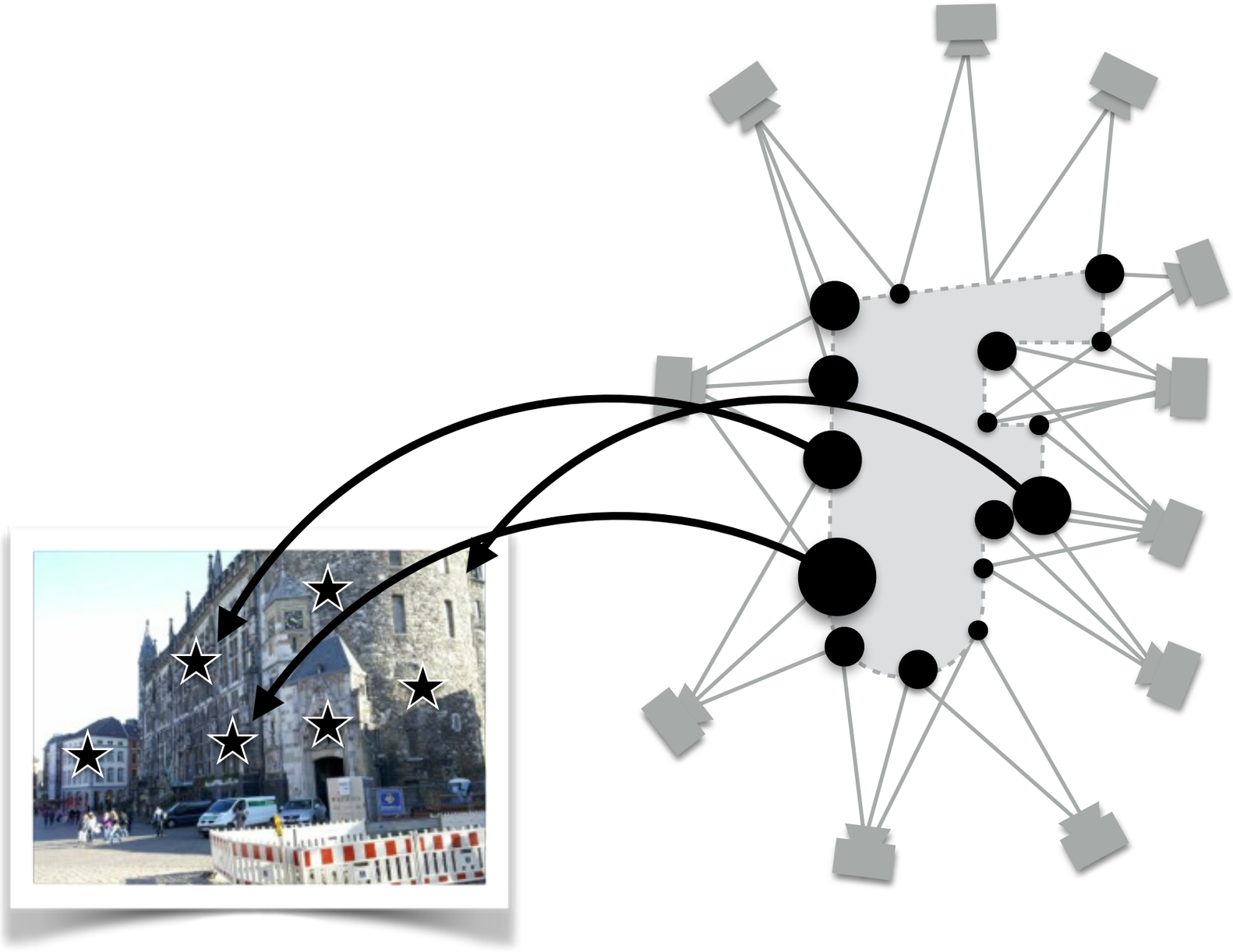
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



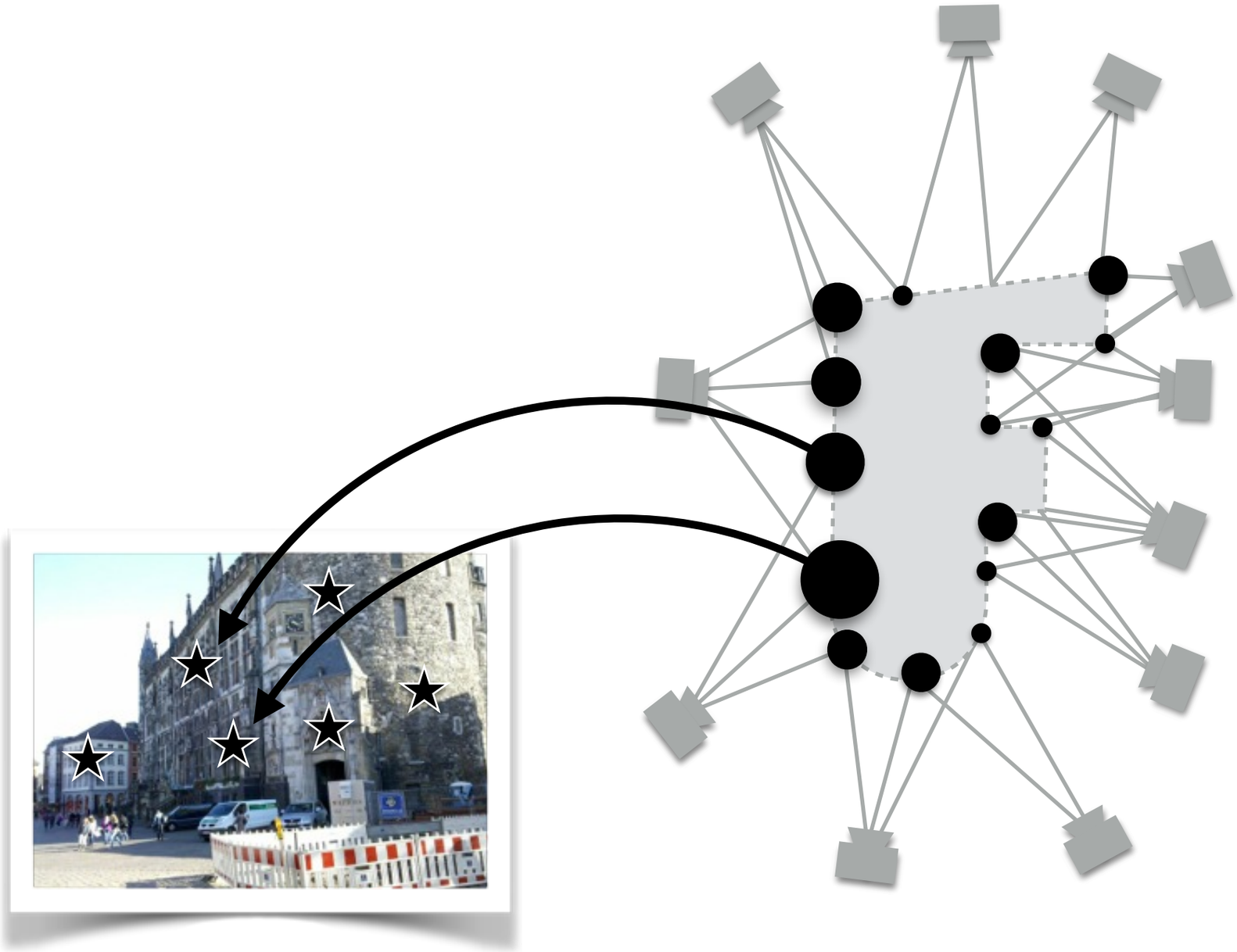
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



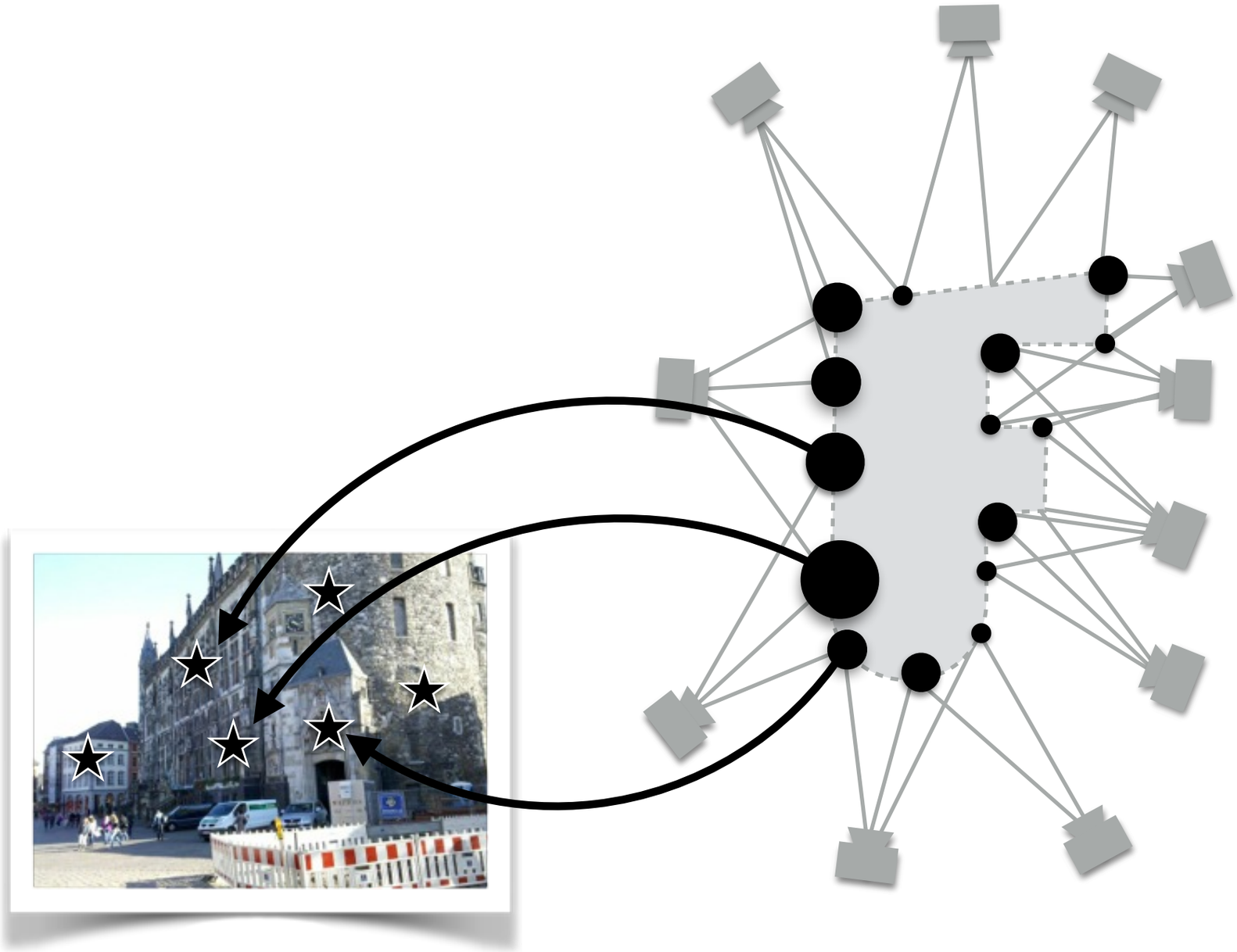
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



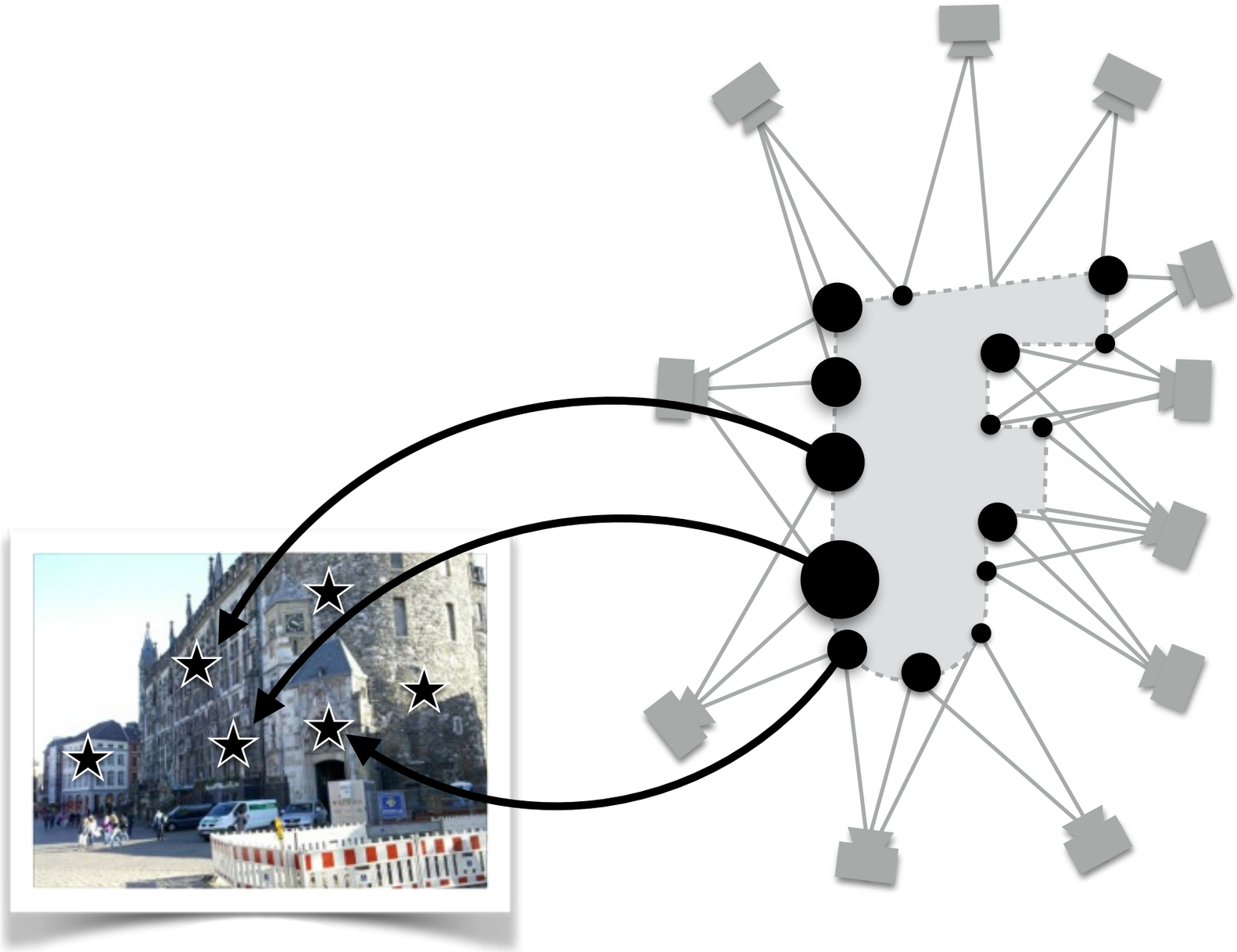
Build kd-tree for query image

Prioritized 3D-to-2D Matching

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



Build kd-tree for query image

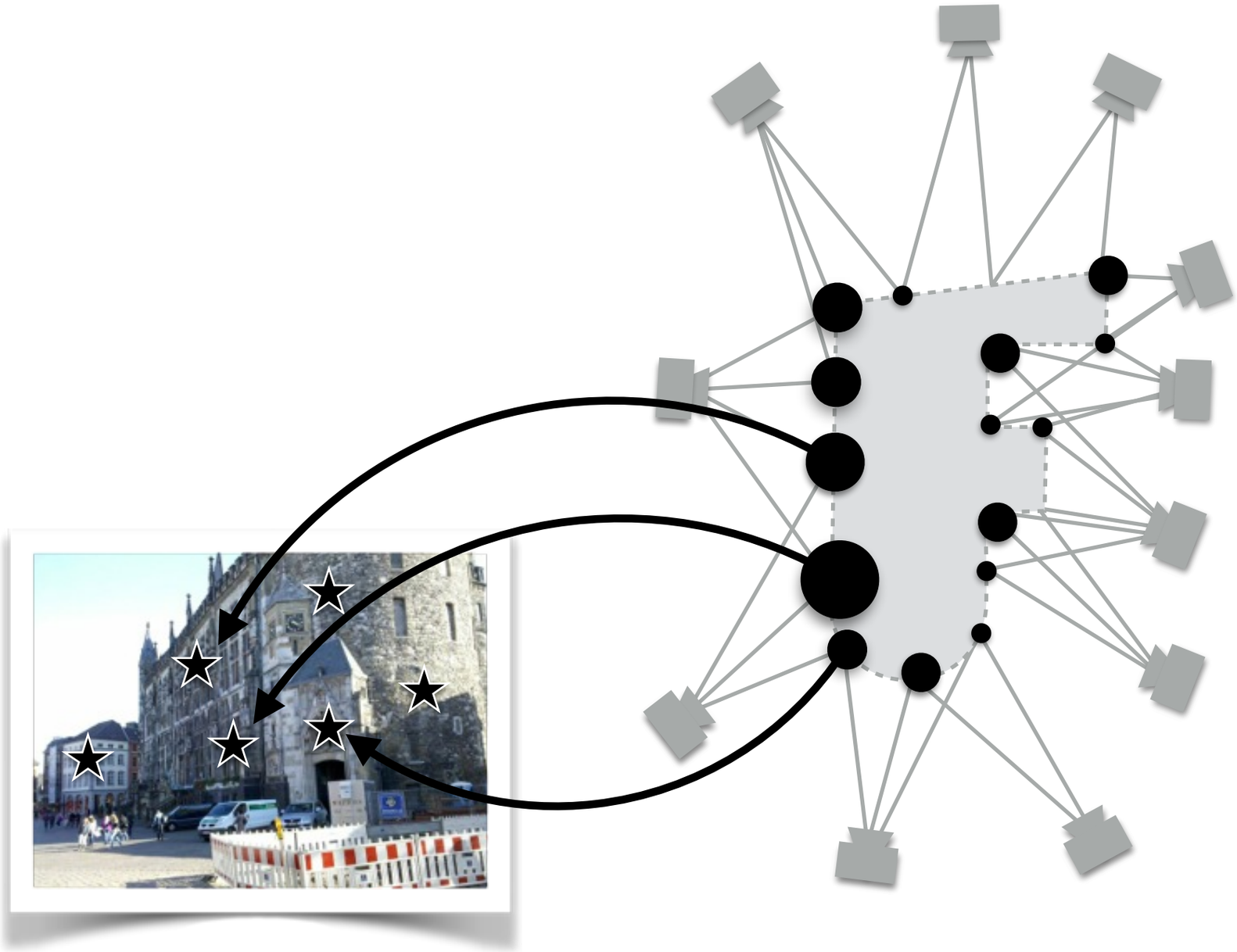
Prioritized 3D-to-2D Matching

Stop after 100 **matches**

[Li et al., ECCV'10]

Prioritized Point-to-Feature Matching (P2F)

Idea: Use Visibility Graph to guide 3D-to-2D matching



Build kd-tree for query image

Prioritized 3D-to-2D Matching

Stop after 100 **matches**

Camera Pose Estimation: RANSAC + P6P

[Li et al., ECCV'10]

Point Priorities

- Start with points that are …

[Li et al., ECCV'10]

Point Priorities

- Start with points that are …
  - stable under viewpoint changes

[Li et al., ECCV'10]

Point Priorities

- Start with points that are …
  - stable under viewpoint changes
  - at more popular parts of the model

[Li et al., ECCV'10]

Point Priorities

- Start with points that are …
  - stable under viewpoint changes
  - at more popular parts of the model
  - ➡ Points with high degree in G_V

[Li et al., ECCV'10]

Point Priorities

- Start with points that are …
  - stable under viewpoint changes
  - at more popular parts of the model
  - ➡ Points with high degree in G_V

- **Initial priority** of point p_i: $S_i = d_i$ = degree in G_V

[Li et al., ECCV'10]

Point Priorities

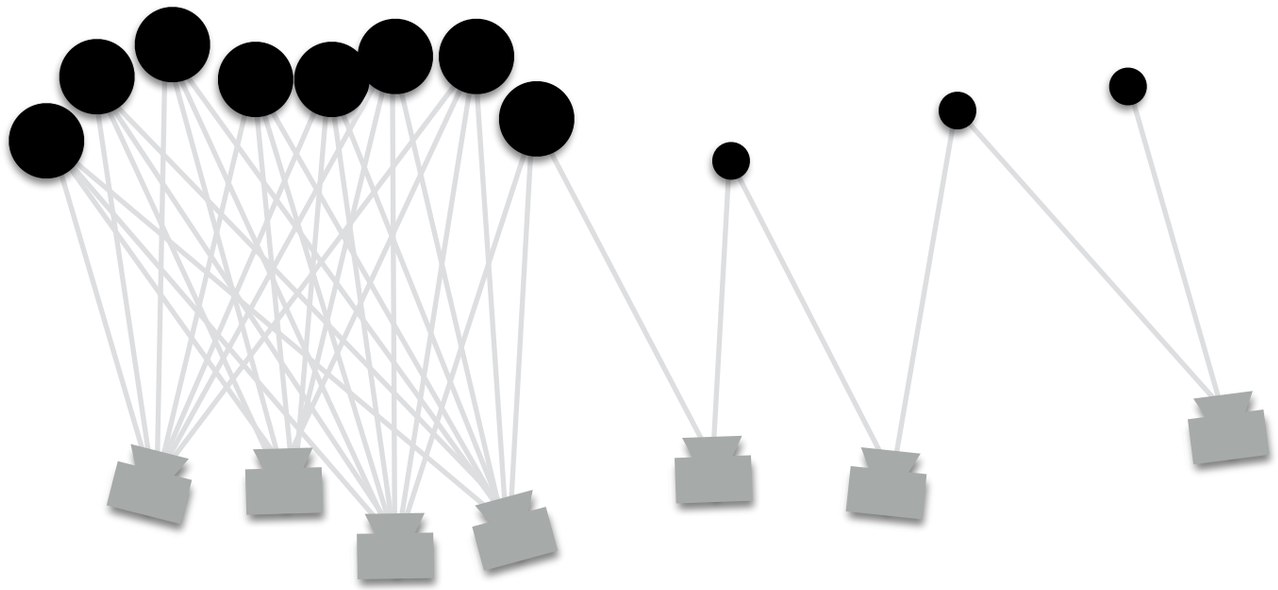
- Start with points that are …
  - stable under viewpoint changes
  - at more popular parts of the model
  - ➡ Points with high degree in G_V

- **Initial priority** of point p_i: $S_i = d_i$ = degree in G_V

- **Update priorities**: $S_j = S_j + 10/d_i$ for co-visible points p_j

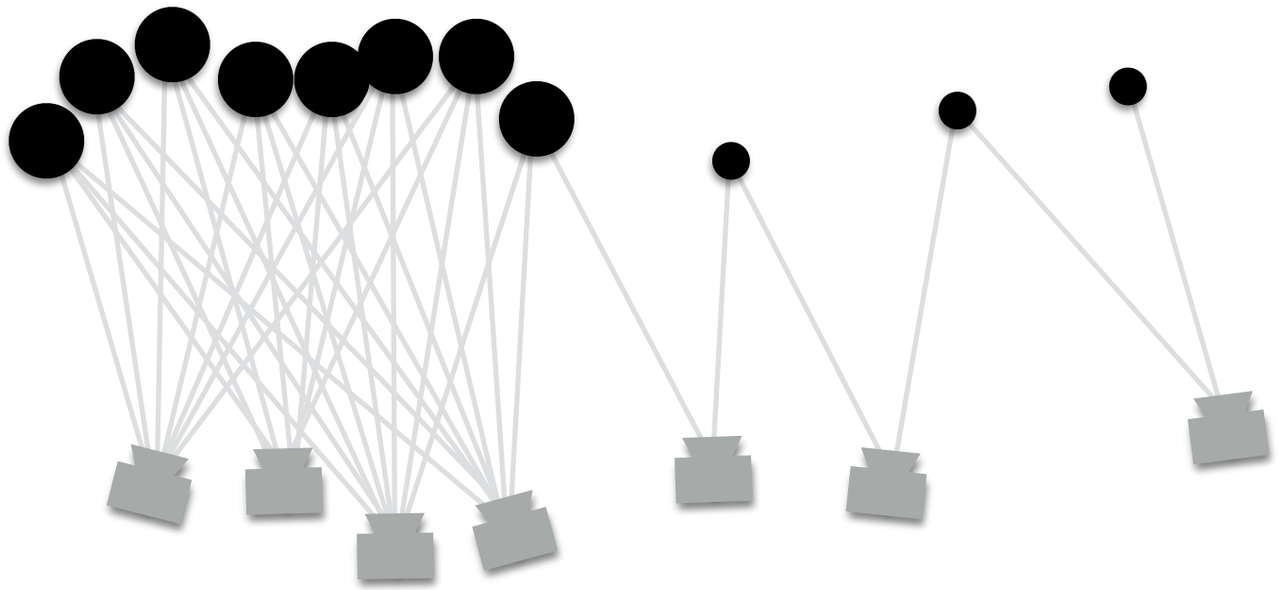
[Li et al., ECCV'10]

Building Compact Models



[Li et al., ECCV'10]

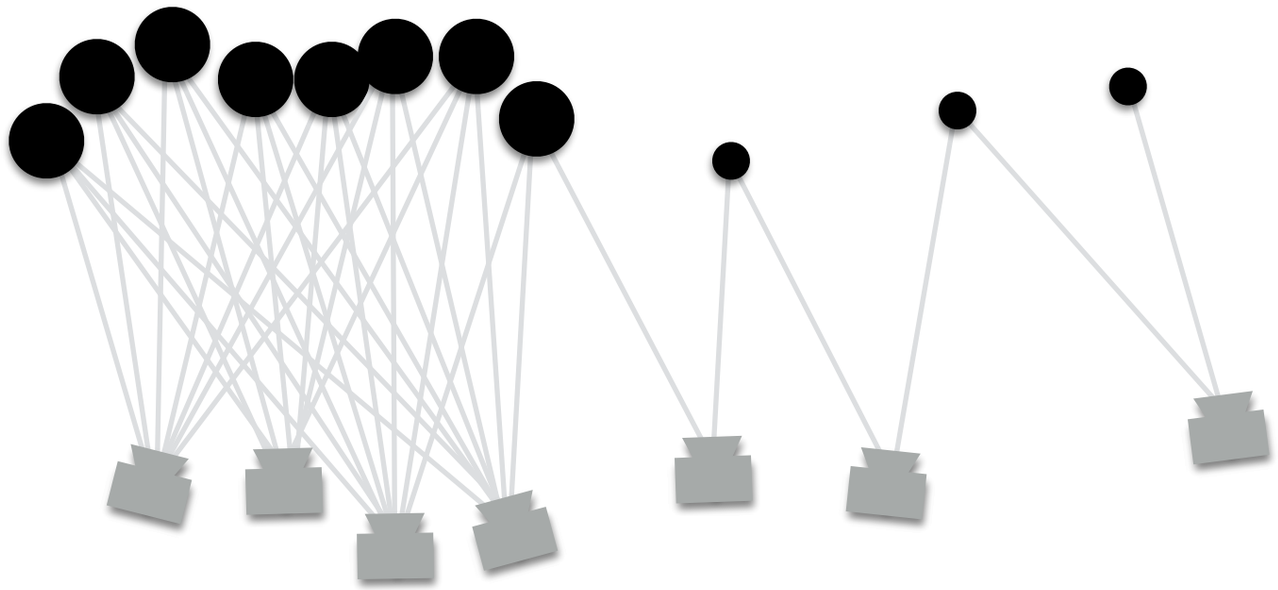
Building Compact Models



Use set of **seed points** to initially “explore” the model:

[Li et al., ECCV’10]

Building Compact Models

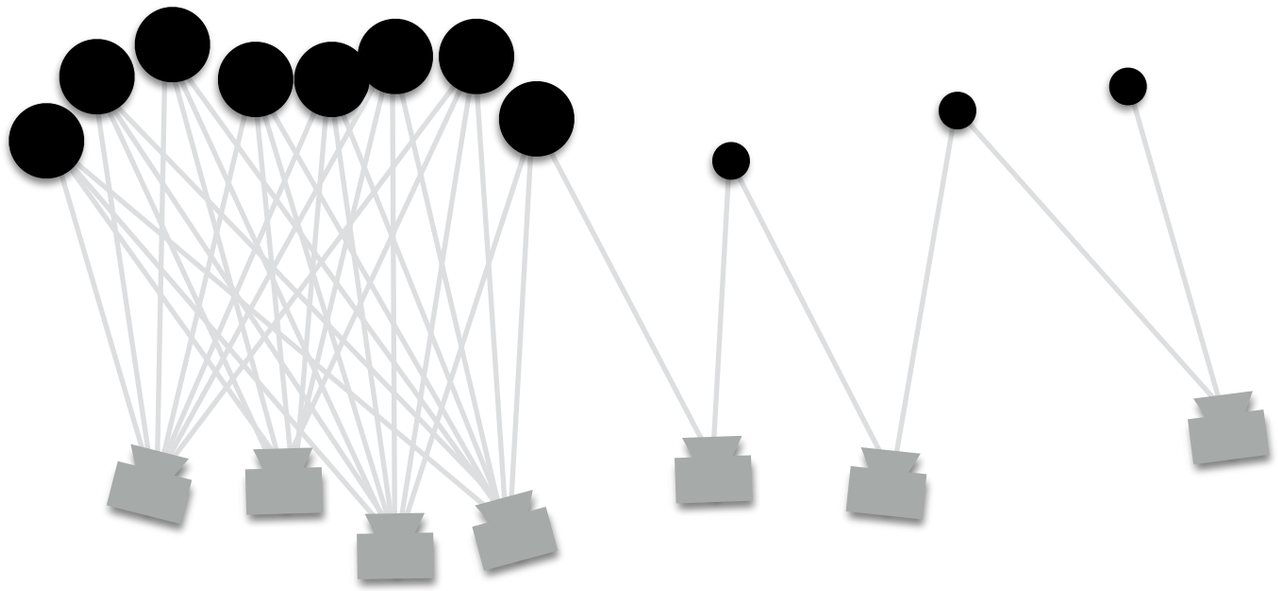


Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points

[Li et al., ECCV’10]

Building Compact Models

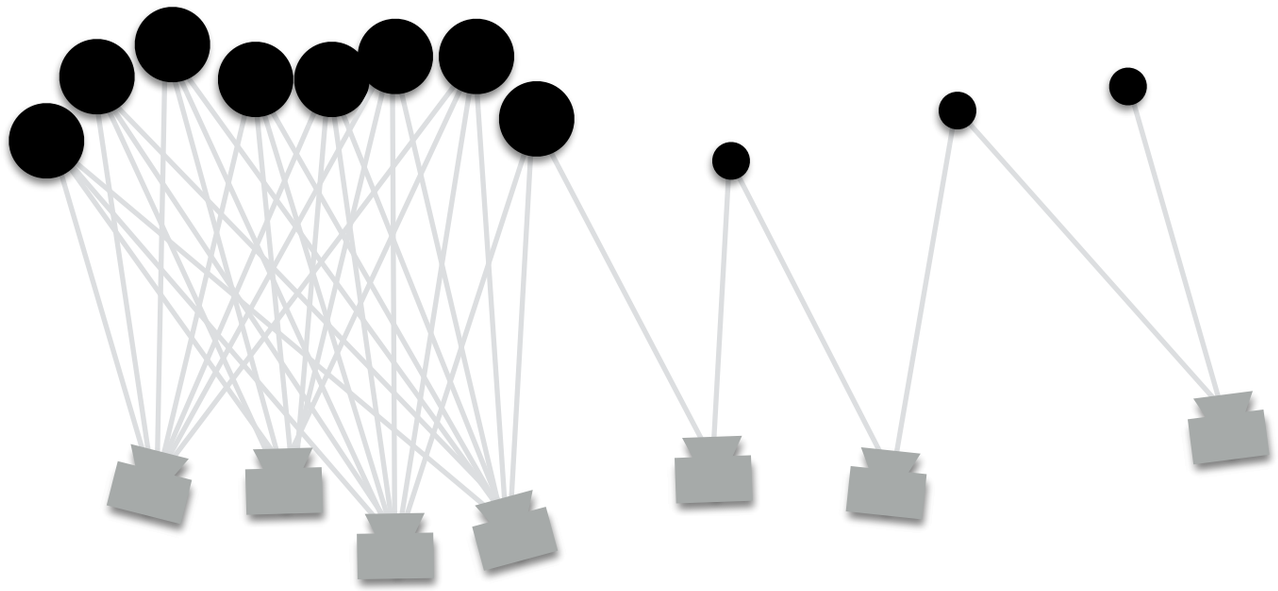


Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

[Li et al., ECCV’10]

Building Compact Models



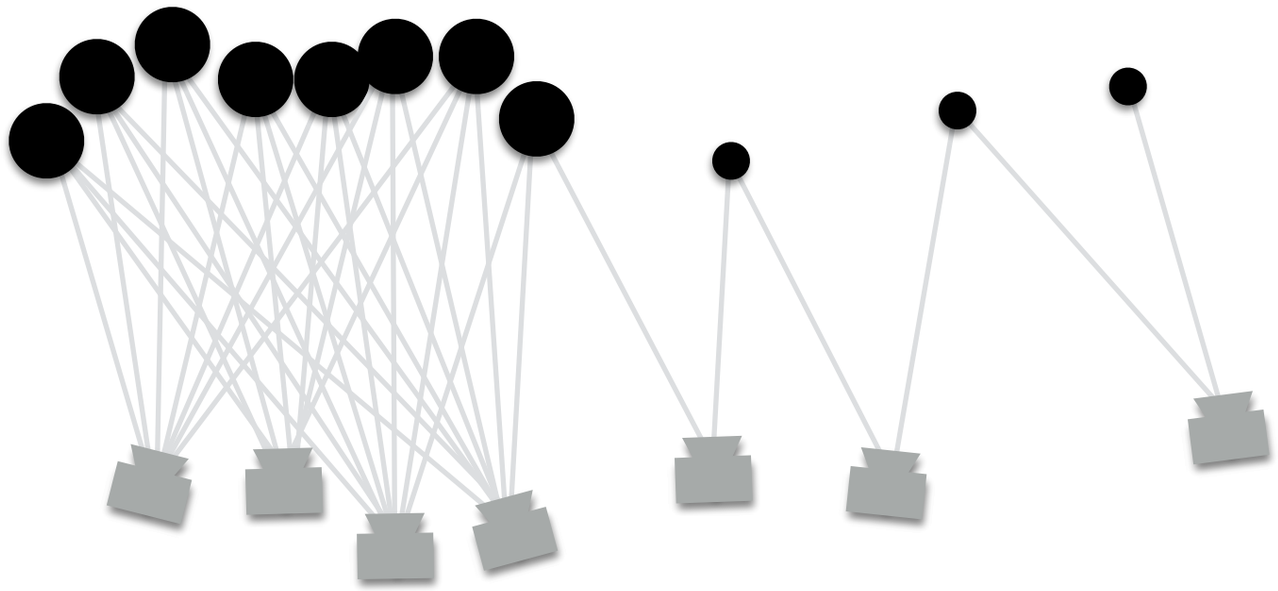
Use set of **seed points** to initially "explore" the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points

[Li et al., ECCV'10]

Building Compact Models



Use set of **seed points** to initially "explore" the model:

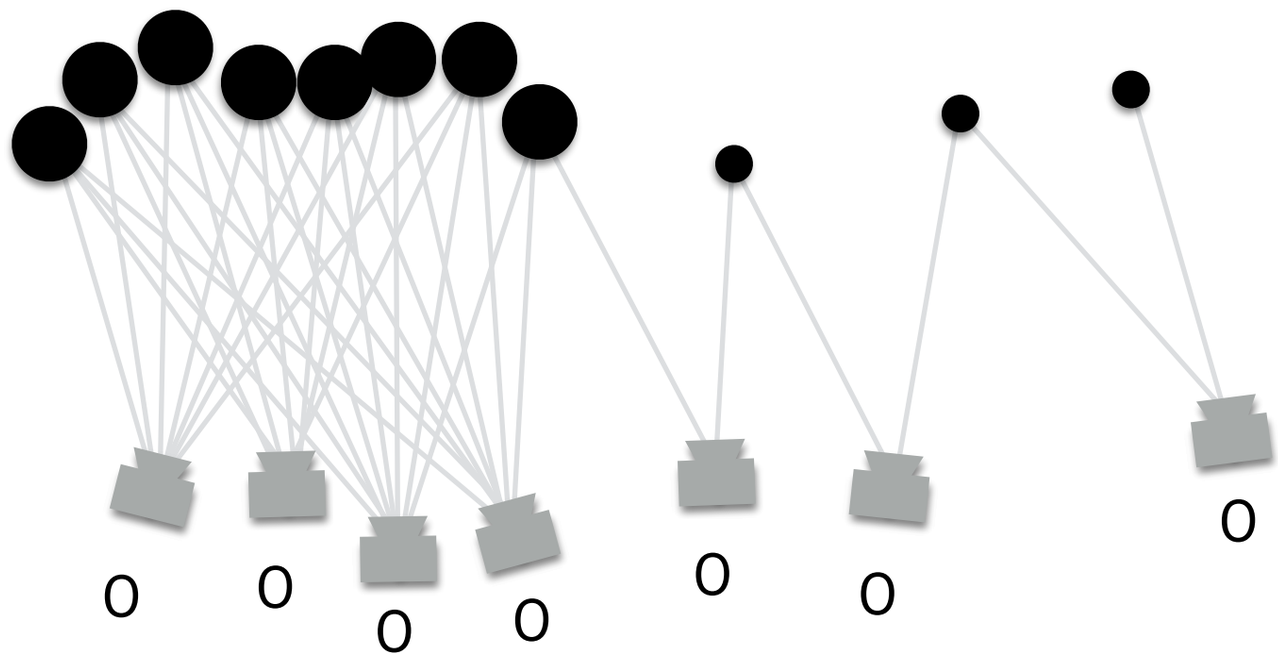
- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points

- Seed points are inserted before normal points in prioritization

[Li et al., ECCV'10]

Building Compact Models



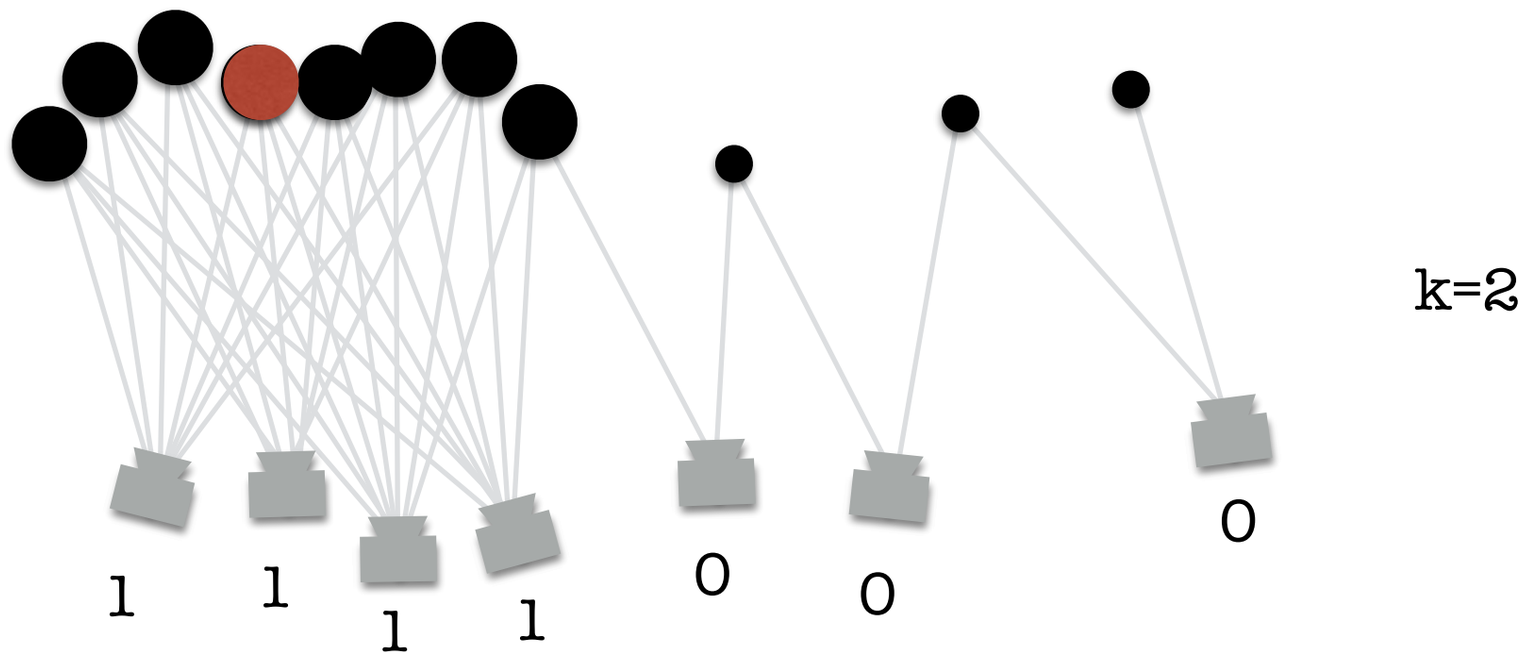
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



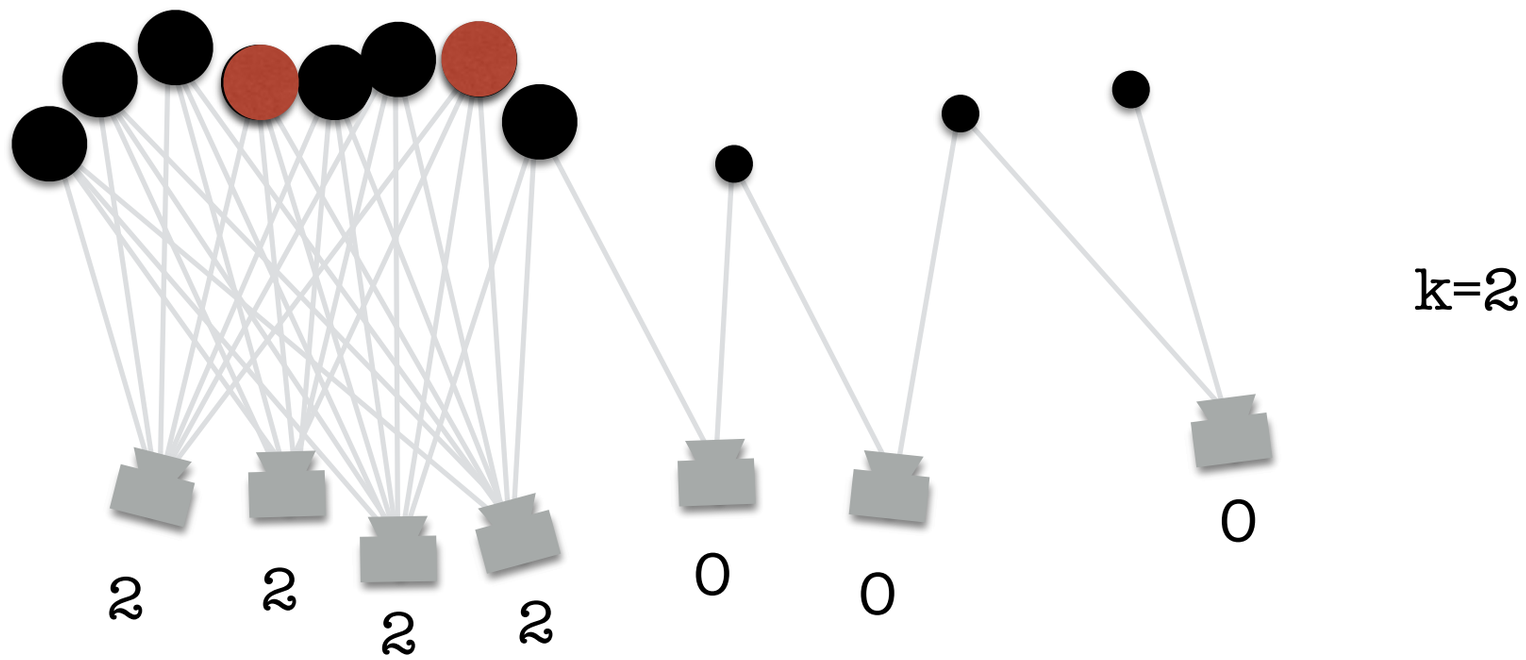
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



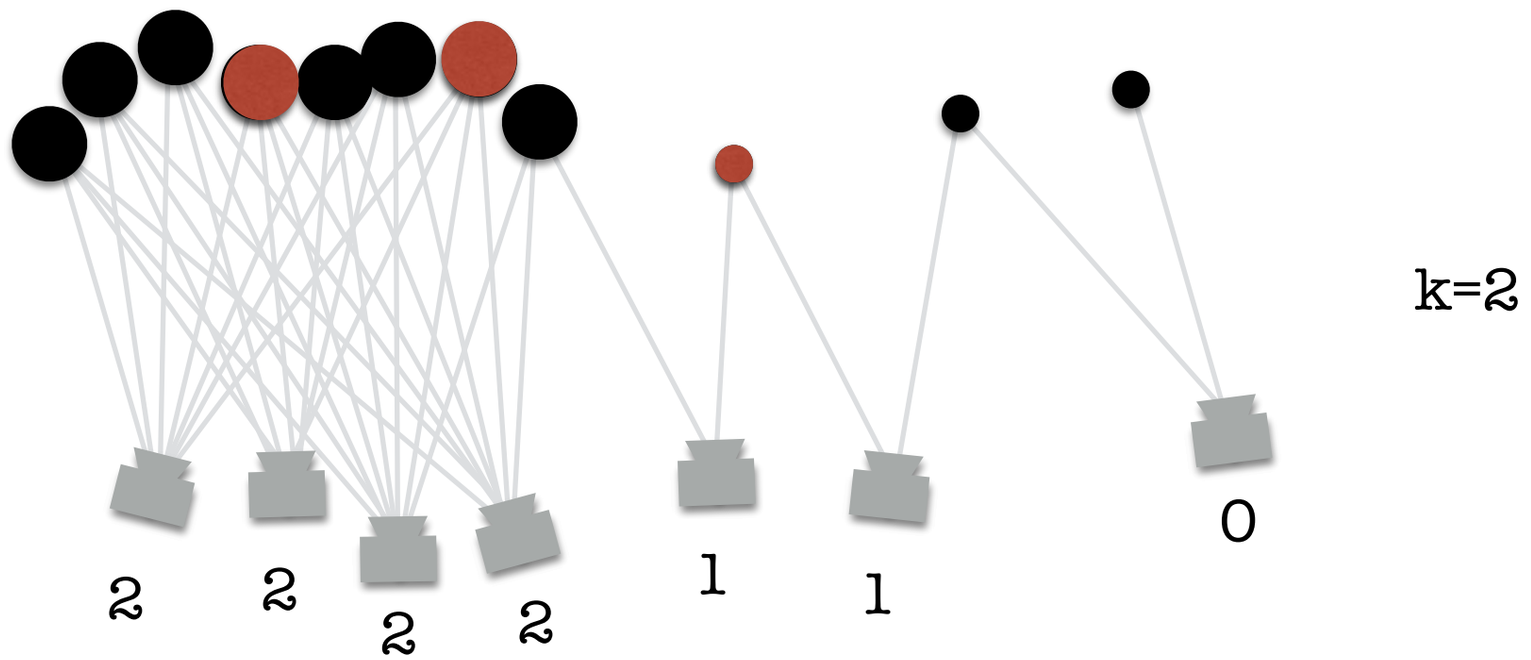
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



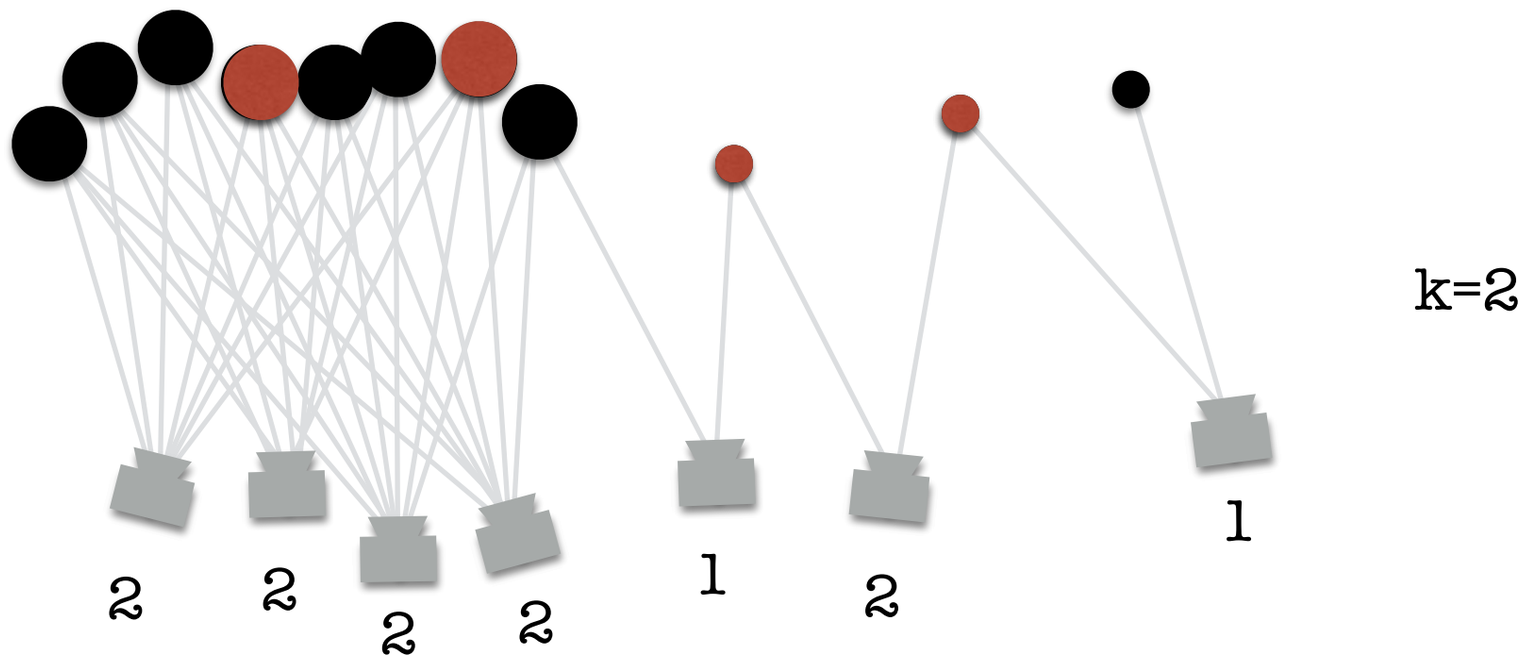
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



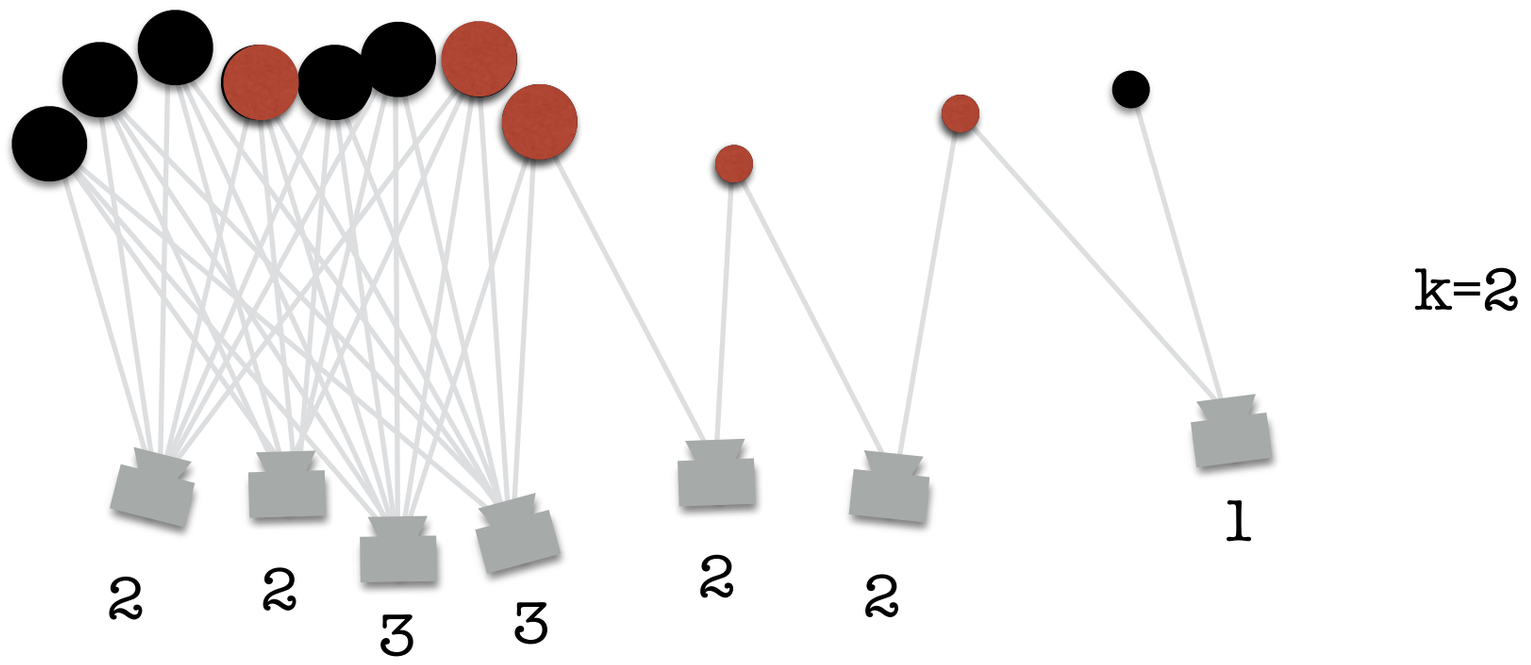
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



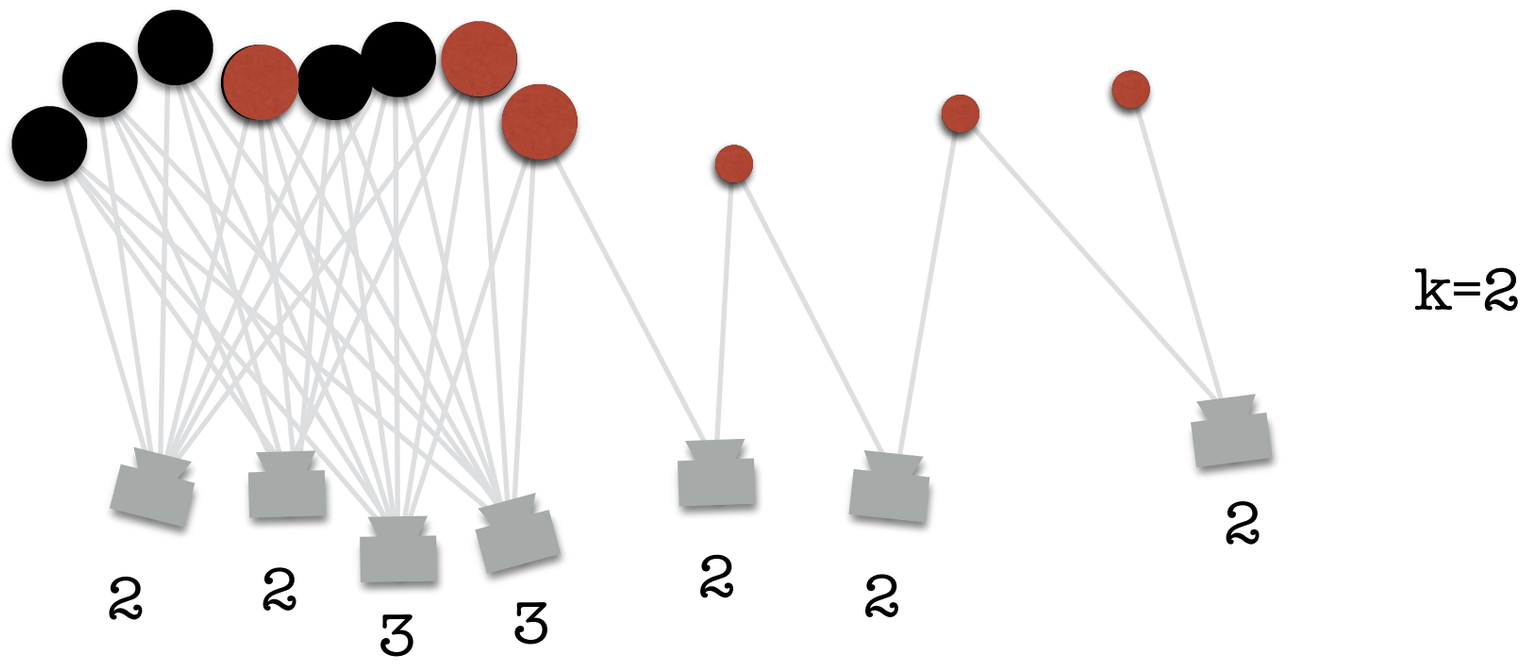
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

Building Compact Models



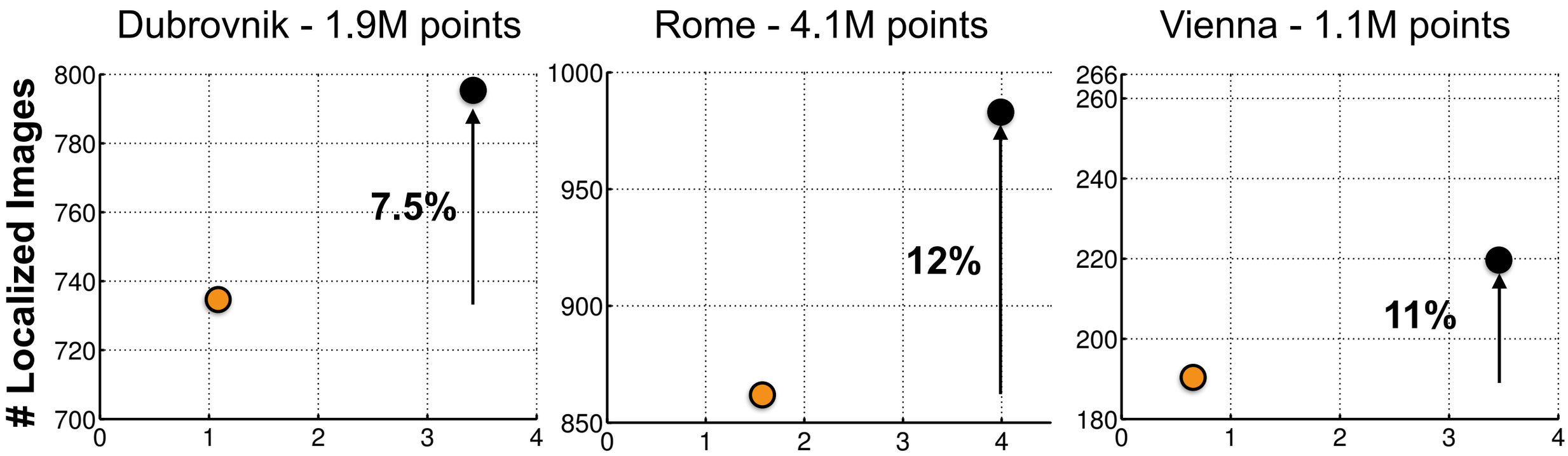
Use set of **seed points** to initially “explore” the model:

- Select subset of points s.t. every camera sees ≥k points
- Greedy Algorithm for Set Cover Problem (NP-complete):

  Iteratively choose point visible in most cameras with ≤k points
- Seed points are inserted before normal points in prioritization

[Li et al., ECCV’10]

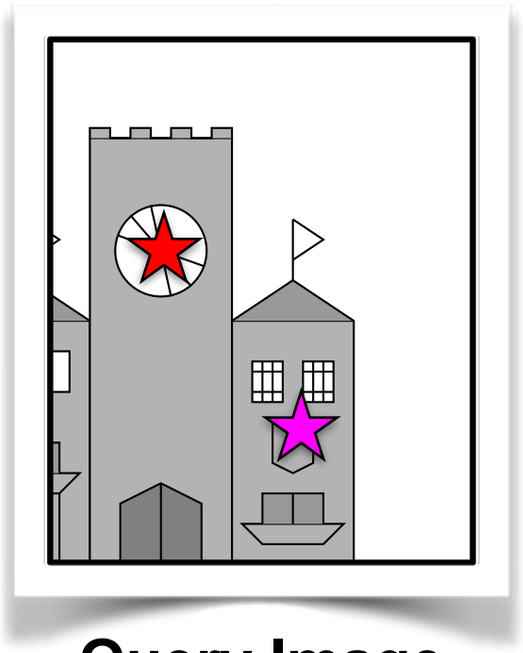
Results



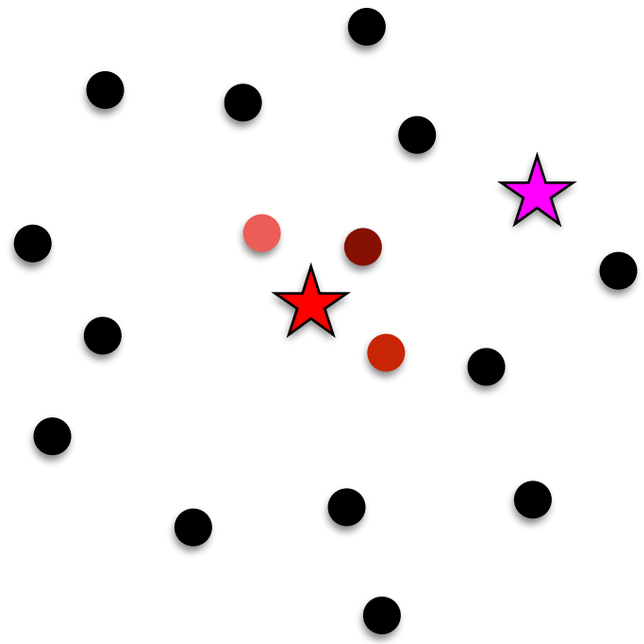
✓Faster than tree-based method due to prioritization…

✗… but significantly less effective

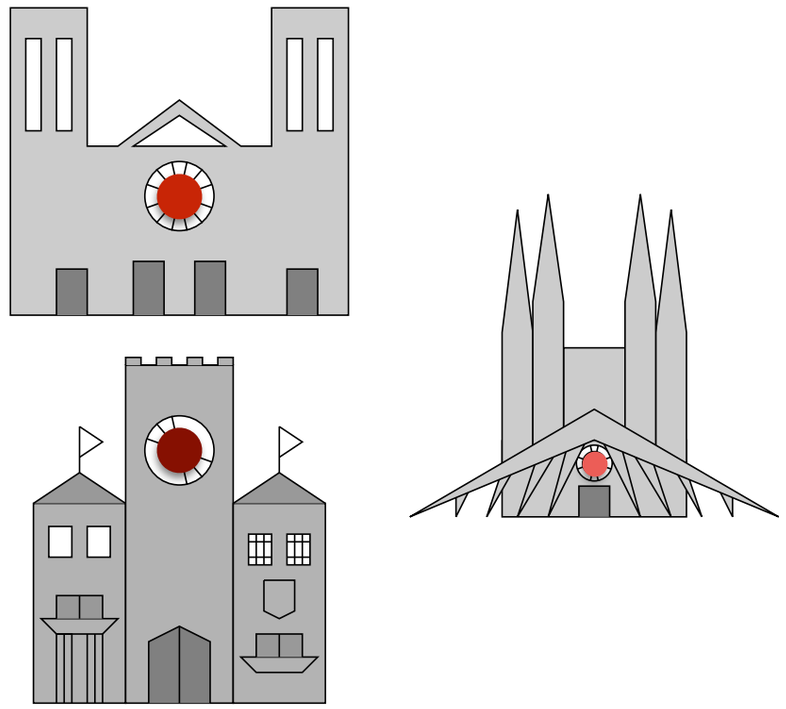
2D-to-3D vs. 3D-to-2D Matching



Query Image

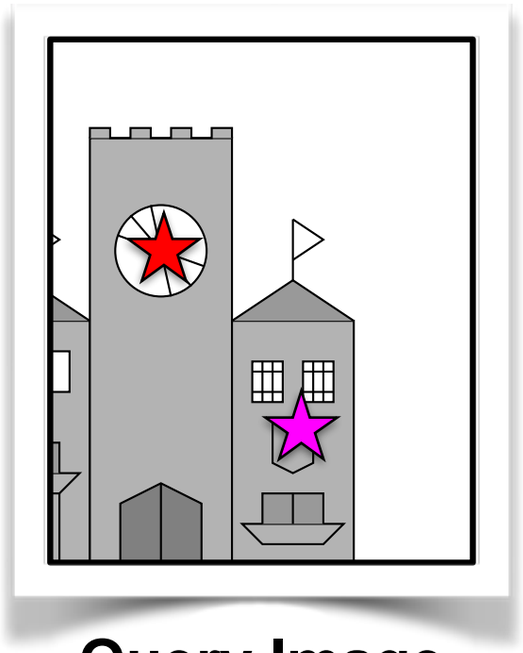


Descriptor Space

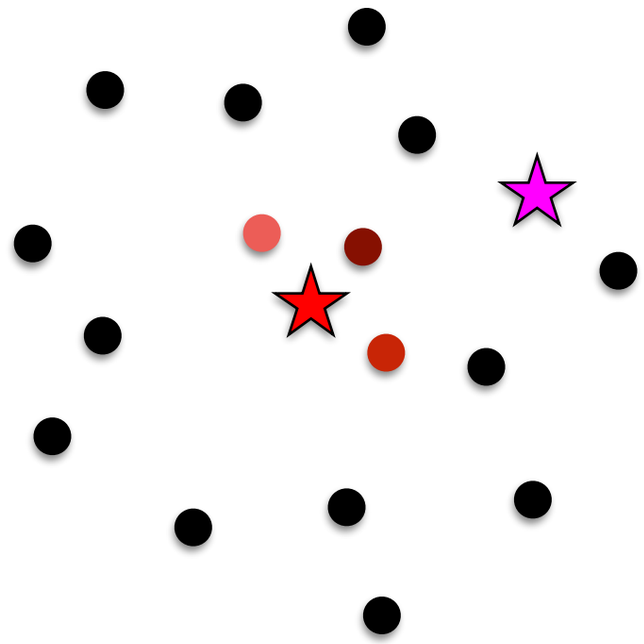


Locally Similar Structures

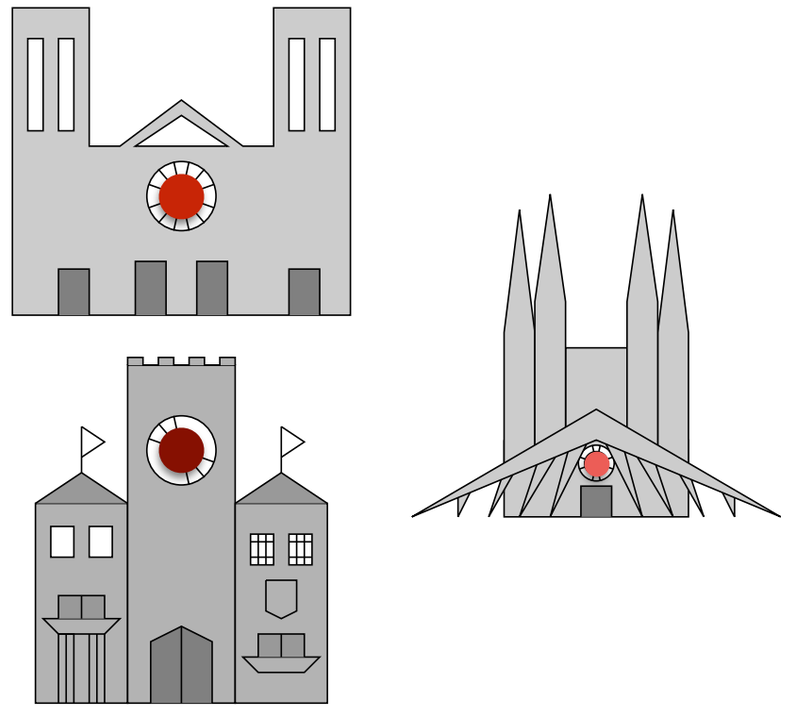
2D-to-3D vs. 3D-to-2D Matching



Query Image



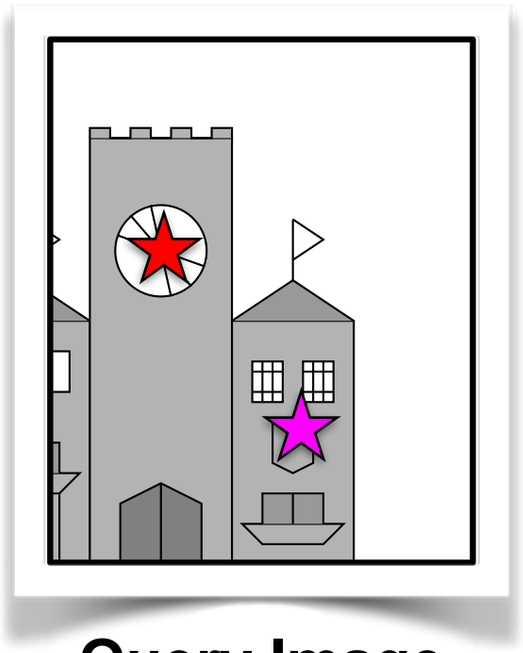
Descriptor Space



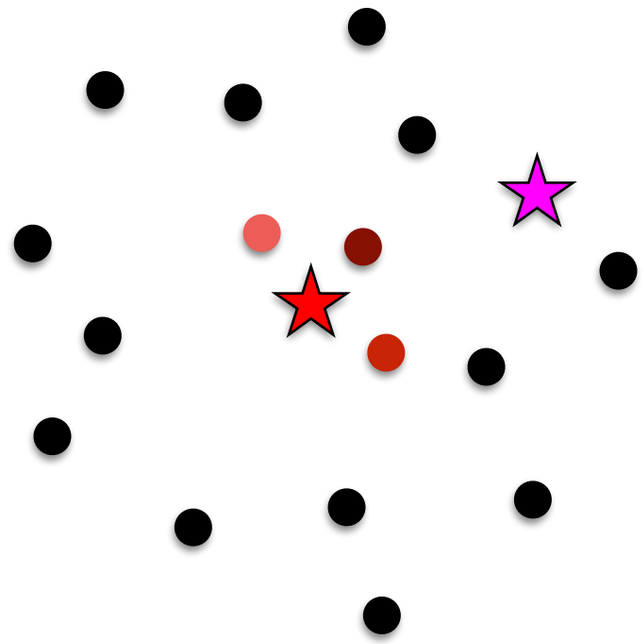
Locally Similar Structures

Globally ambiguous structures

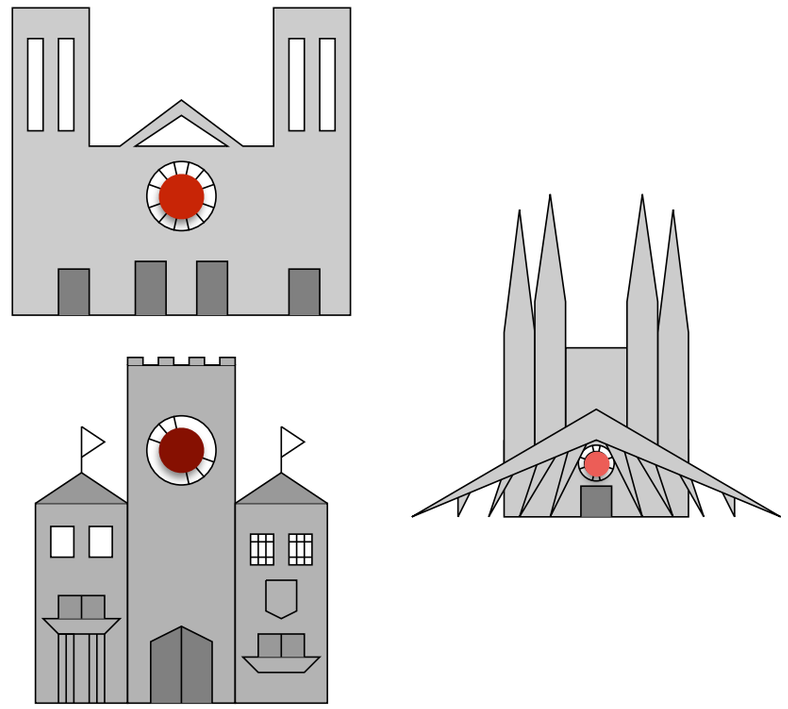
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

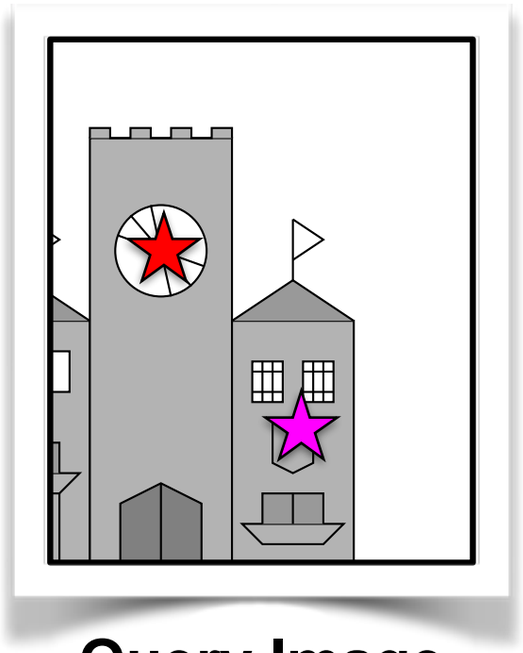


Locally Similar Structures

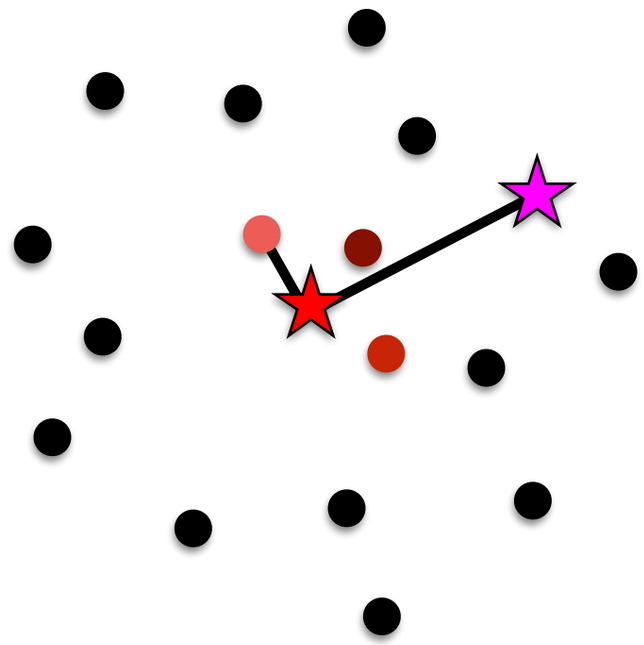
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching

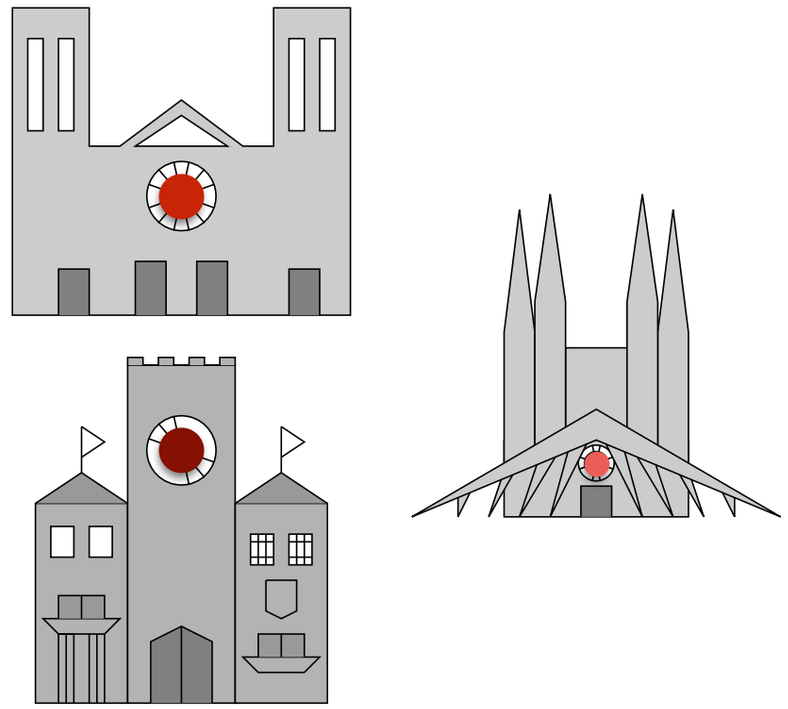
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

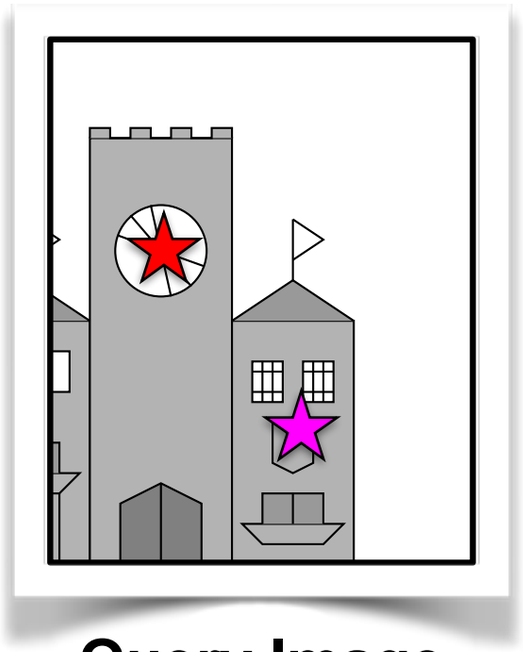


Locally Similar Structures

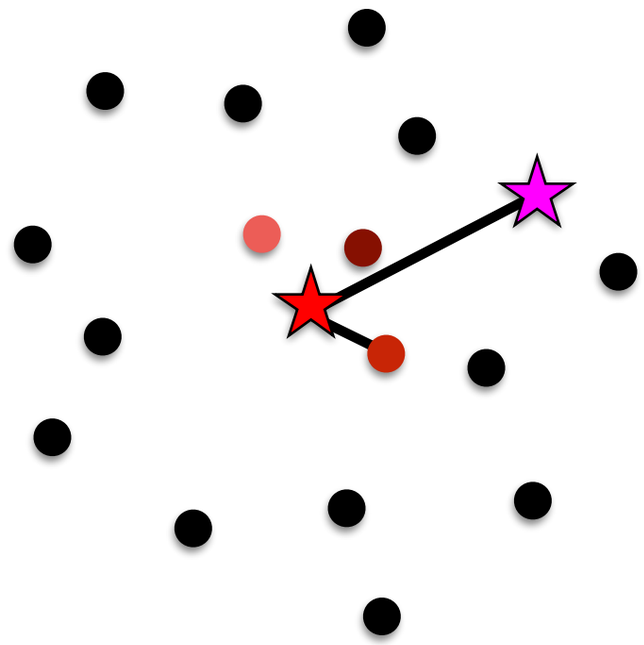
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching

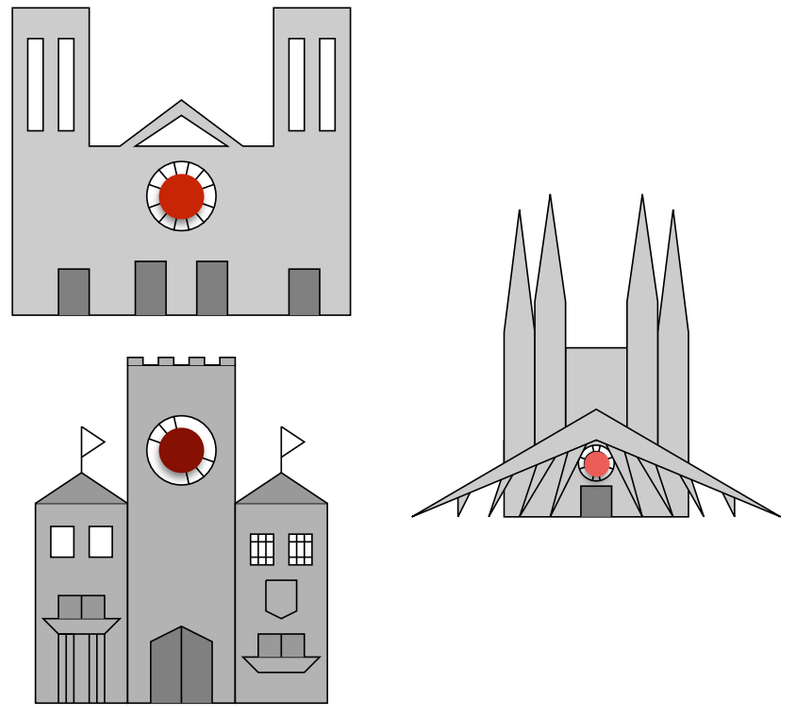
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

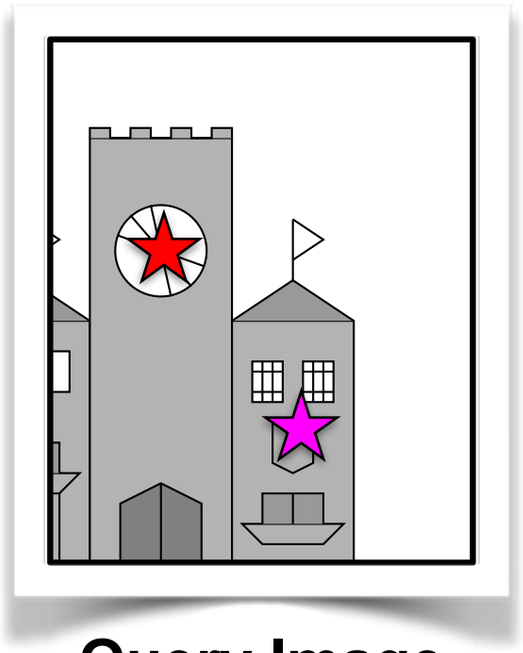


Locally Similar Structures

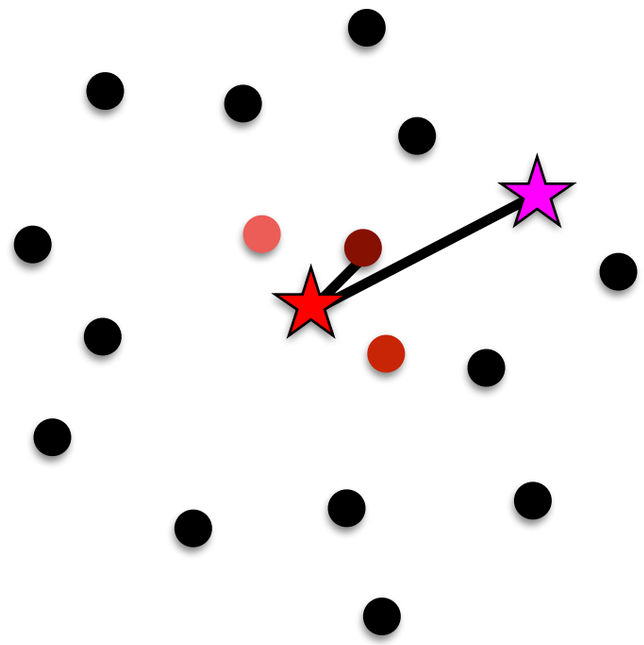
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching

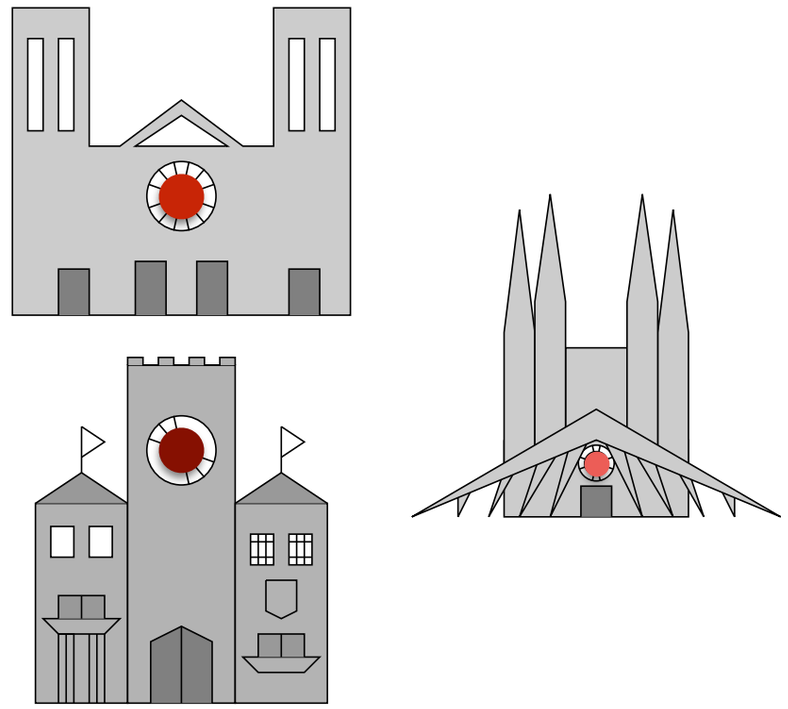
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

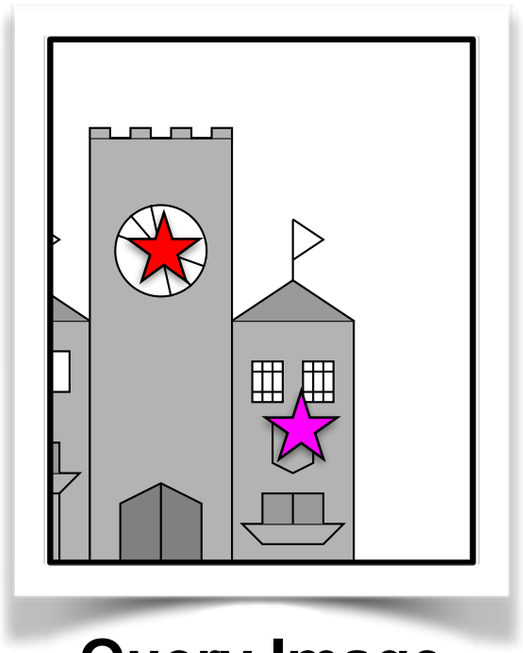


Locally Similar Structures

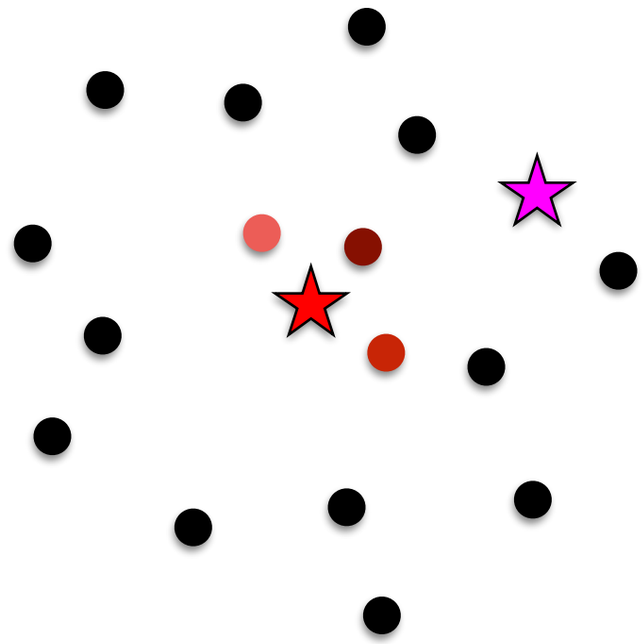
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching

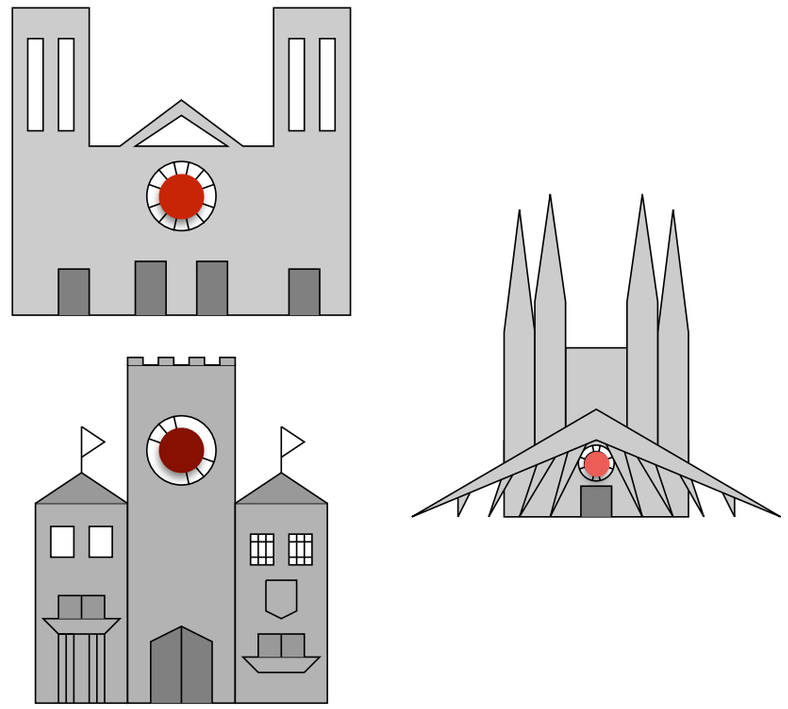
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

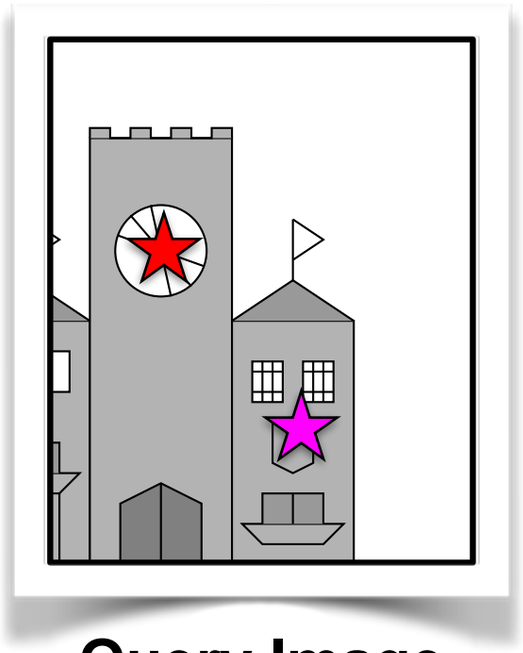


Locally Similar Structures

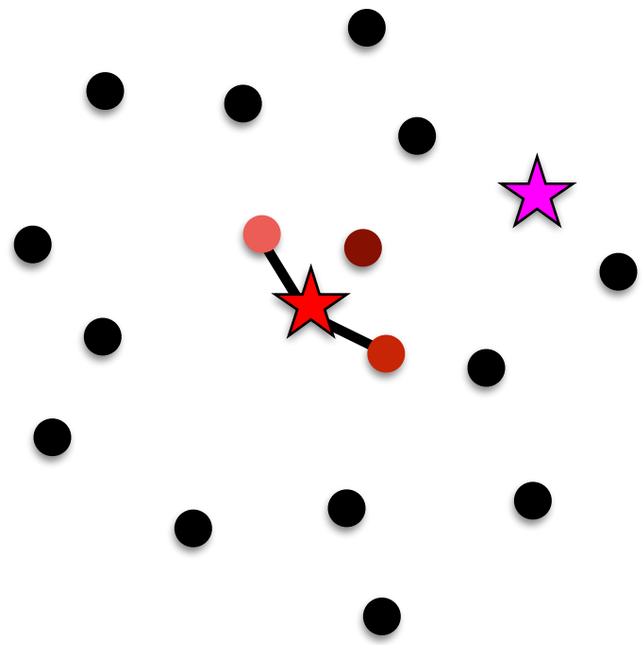
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching
- … rejected by ratio test for 2D-to-2D matching

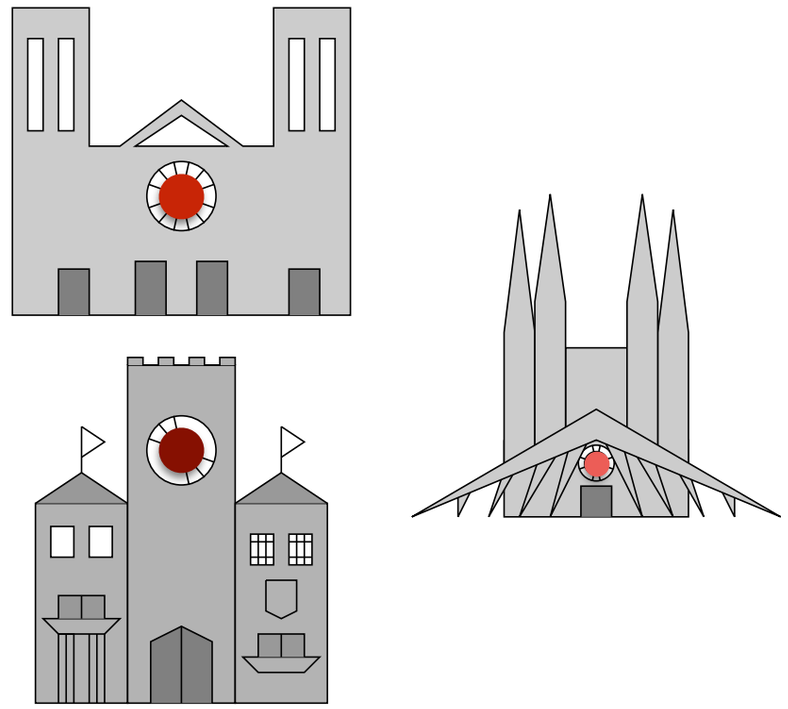
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space

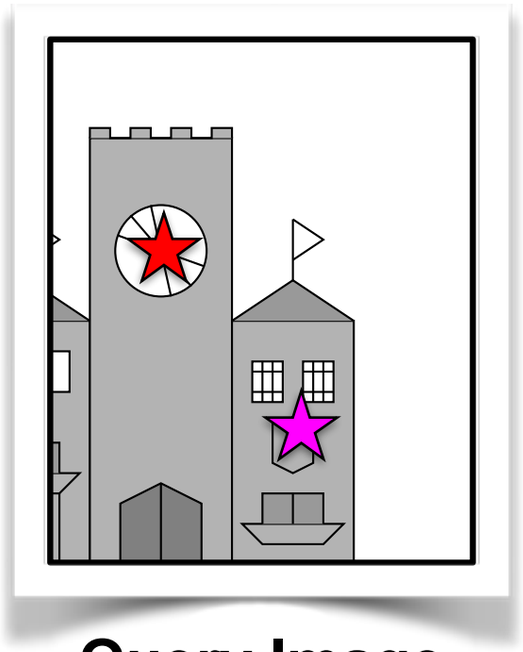


Locally Similar Structures

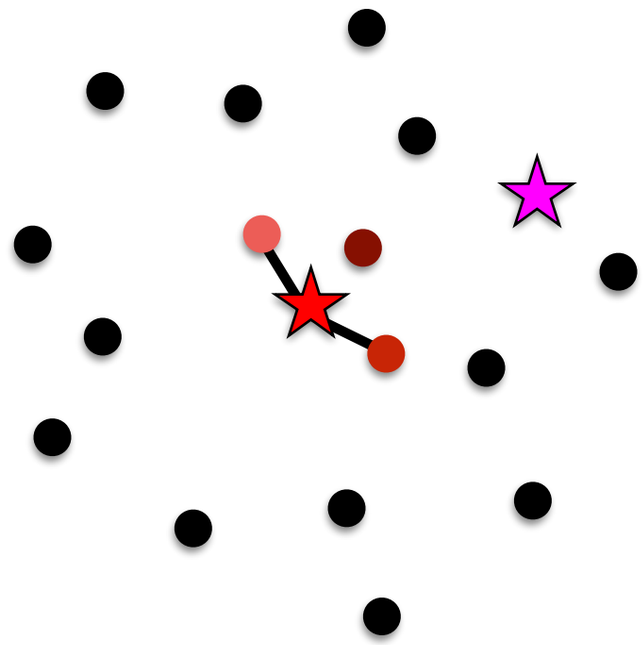
Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching
- … rejected by ratio test for 2D-to-2D matching

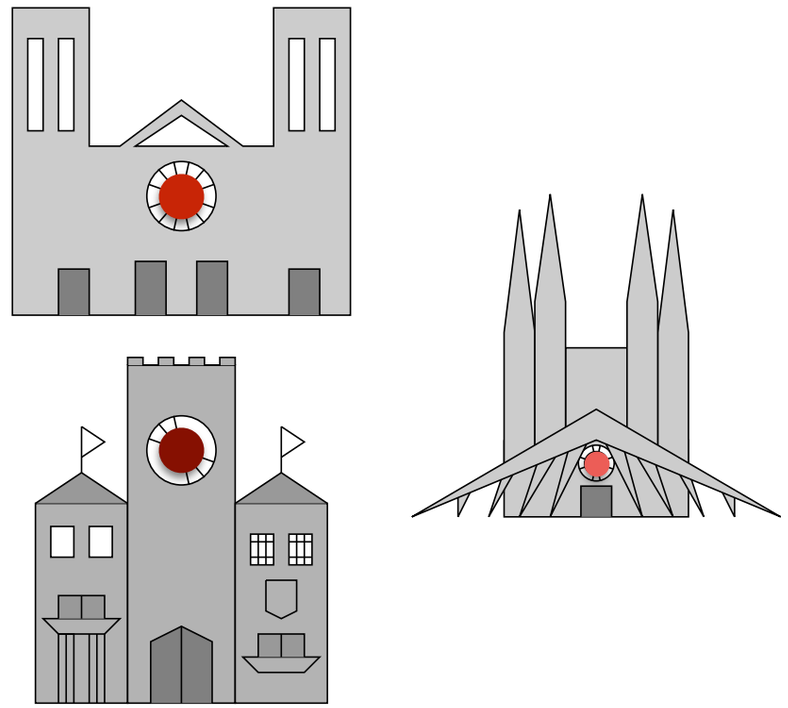
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space



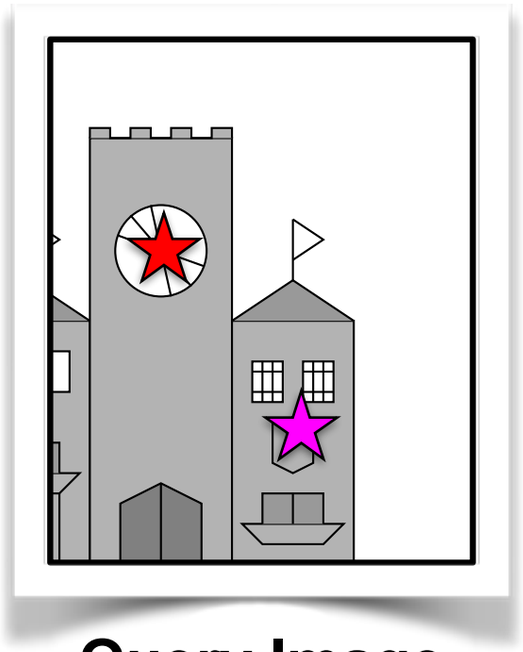
Locally Similar Structures

Globally ambiguous structures

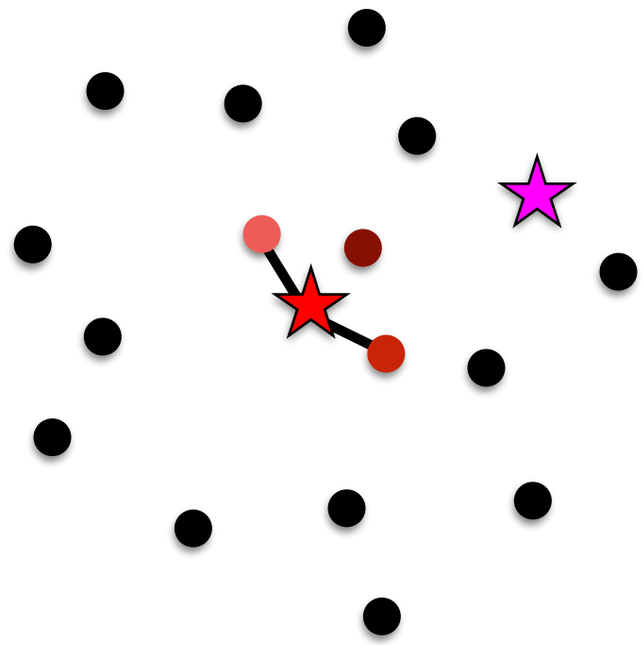
- … pass ratio test for 3D-to-2D matching
- … rejected by ratio test for 2D-to-2D matching

- P2F might terminate too early!

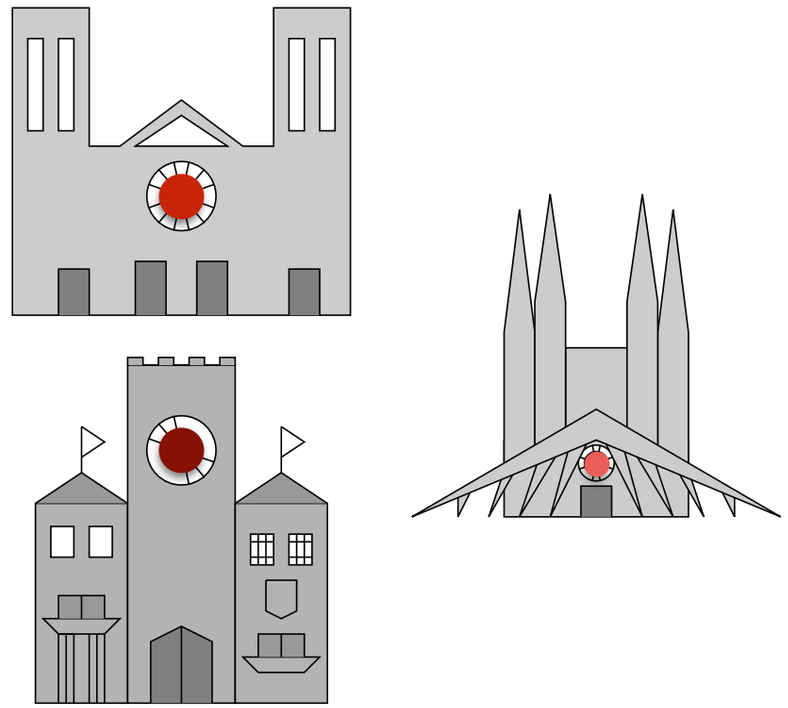
2D-to-3D vs. 3D-to-2D Matching



Query Image



Descriptor Space



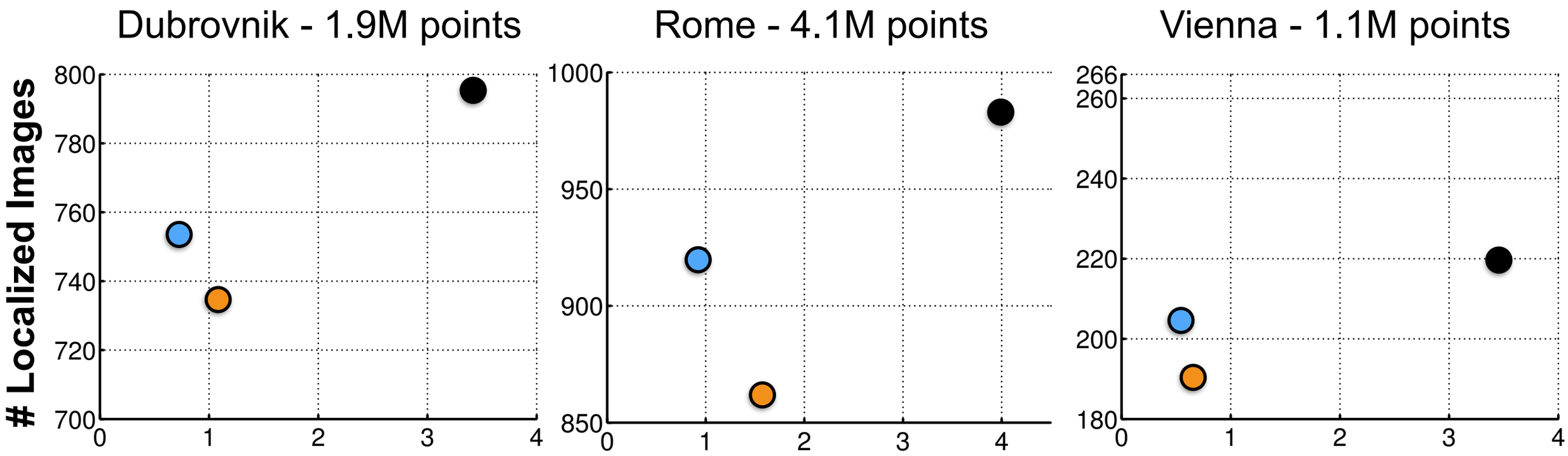
Locally Similar Structures

Globally ambiguous structures

- … pass ratio test for 3D-to-2D matching
- … rejected by ratio test for 2D-to-2D matching

- P2F might terminate too early!
- Use compact version of model obtained using Greedy (2-4% of points)

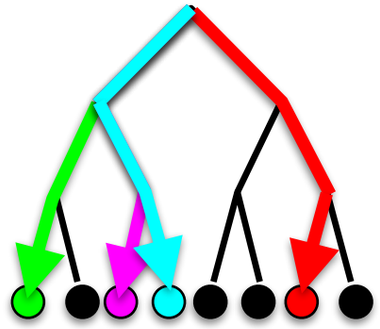
Results



✓ Better than when using the full model

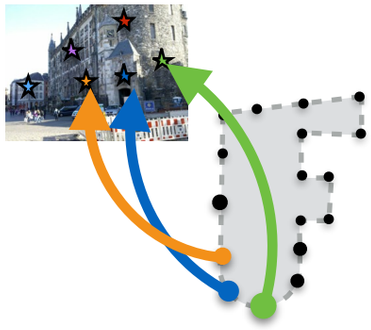
✗ … but still less effective than 2D-to-3D matching

Localization - Overview



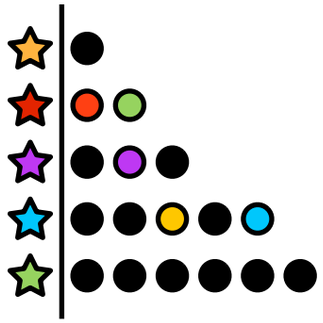
Baseline: kd-tree search

[Sattler et al., ICCV'11]



P2F

[Li et al., ECCV'10]



VPS

[Sattler et al., ICCV'11]

Active Search

[Sattler et al., ECCV'12]

effectivness	✓✓	✗✗	✗	✓✓✓
efficiency	✗✗	✓	✓	✓✓

Prioritizing 2D-to-3D Search

- 2D-to-3D search more effective than 3D-to-2D matching

[Sattler et al., ICCV'11]

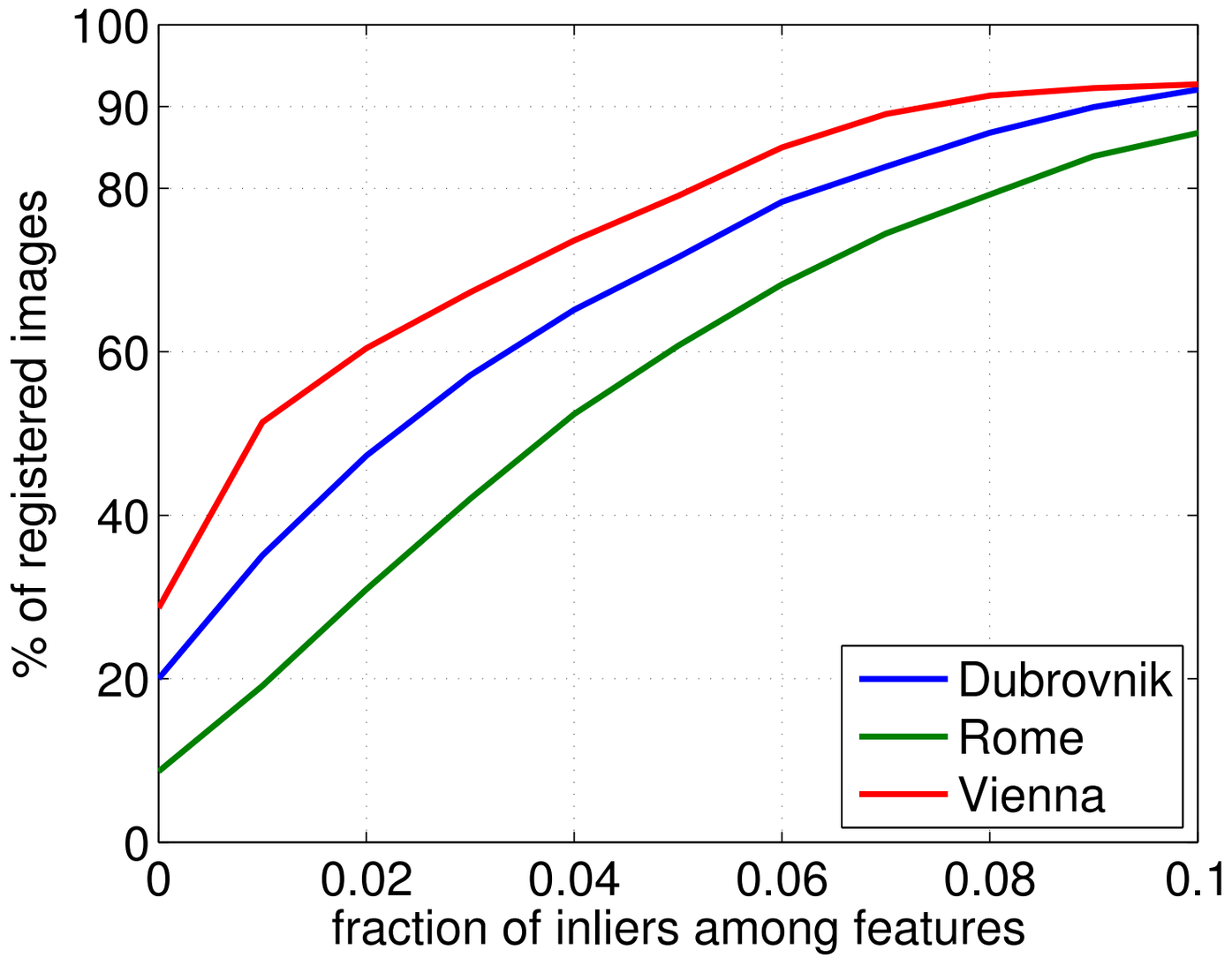
Prioritizing 2D-to-3D Search

- 2D-to-3D search more effective than 3D-to-2D matching
  - … but unclear how to best incorporate co-visibility

[Sattler et al., ICCV'11]

Prioritizing 2D-to-3D Search

- 2D-to-3D search more effective than 3D-to-2D matching
  - … but unclear how to best incorporate co-visibility
- Still, lot of potential to improve upon tree-based search:



[Sattler et al., ICCV'11]

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:
  - p_i: Probability of finding correct match for i^{th} feature

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:
  - p_i: Probability of finding correct match for i^{th} feature
  - c_i: Search cost for i^{th} feature

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:
  - p_i: Probability of finding correct match for i^{th} feature
  - c_i: Search cost for i^{th} feature
  - Select subset of features by solving

$$\min \sum_i X_i c_i \quad \text{s.t.} \quad \sum_i X_i p_i \geq N_t \text{ with } X_i \in \{0,1\}$$

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:
  - p_i: Probability of finding correct match for i^{th} feature
  - c_i: Search cost for i^{th} feature
  - Select subset of features by solving

$$\min \sum_i X_i c_i \quad \text{s.t.} \quad \sum_i X_i p_i \geq N_t \text{ with } X_i \in \{0,1\}$$

search costs

* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search

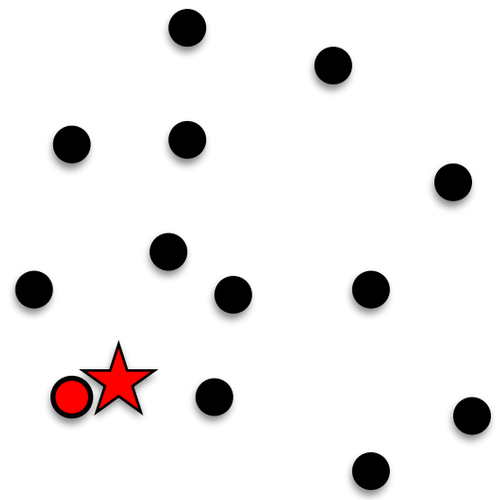
- 10x speed-up ... if we identify matching features **before** matching*
- **Probabilistic approach**:
  - p_i: Probability of finding correct match for i^{th} feature
  - c_i: Search cost for i^{th} feature
  - Select subset of features by solving

$$\min \sum_i X_i c_i \quad \text{s.t.} \quad \sum_i X_i p_i \geq N_t \text{ with } X_i \in \{0,1\}$$

search costs — **expected # matches**

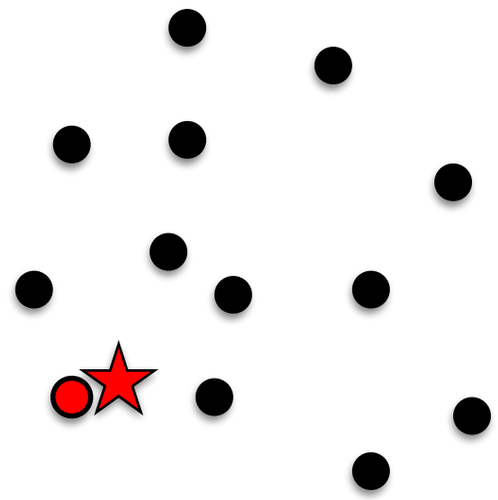
* Predicting matchability [Hartmann et al., CVPR'14] (Orals 1A)

Probabilistic 2D-to-3D Search



Efficient computation of p_i, c_i

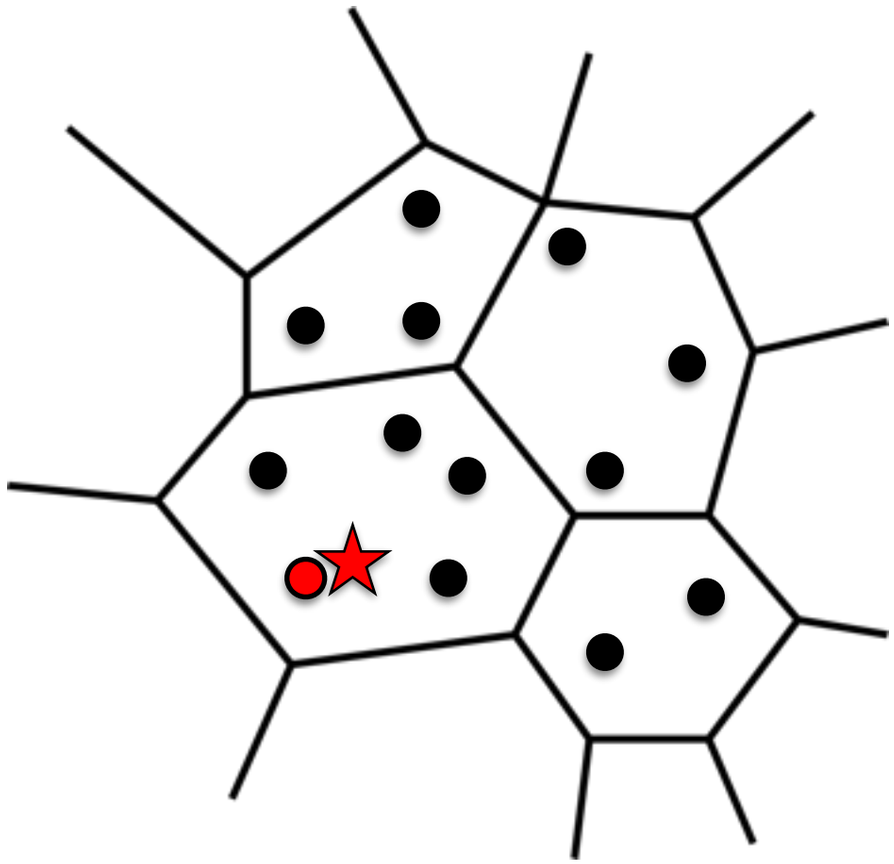
Probabilistic 2D-to-3D Search



Efficient computation of p_i, c_i

- Precompute probabilities for **regions** in descriptor space

Probabilistic 2D-to-3D Search



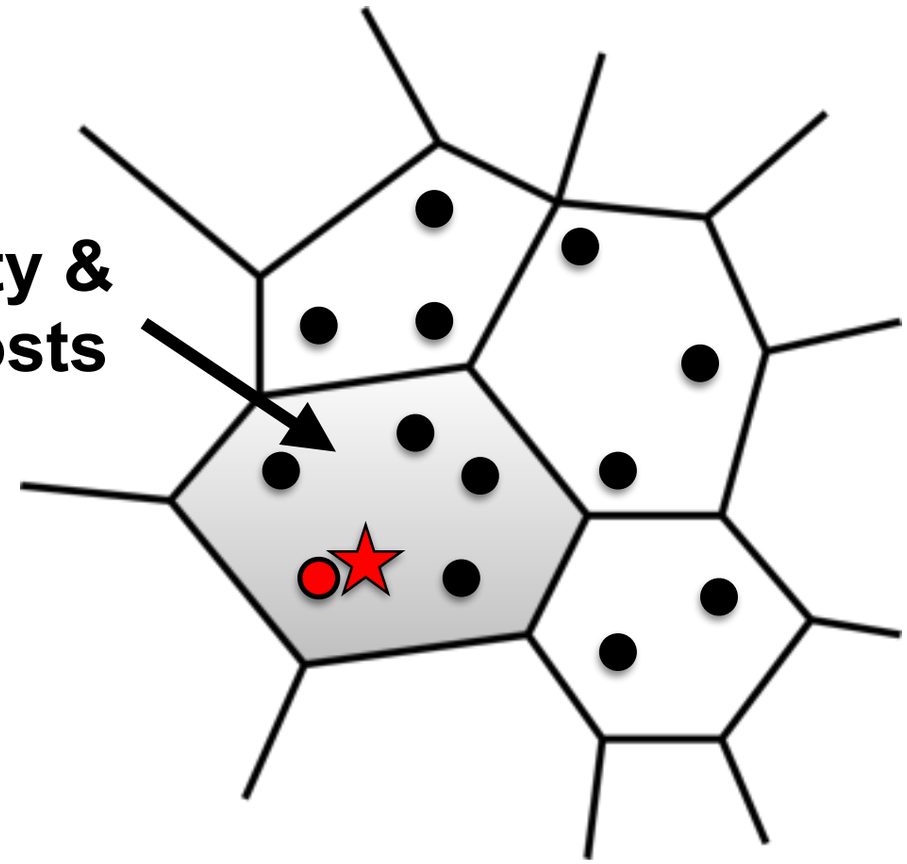
Fixed & Precomputed Visual Vocabulary

Efficient computation of p_i, c_i

- Precompute probabilities for **regions** in descriptor space

Probabilistic 2D-to-3D Search

Constant probability & constant search costs



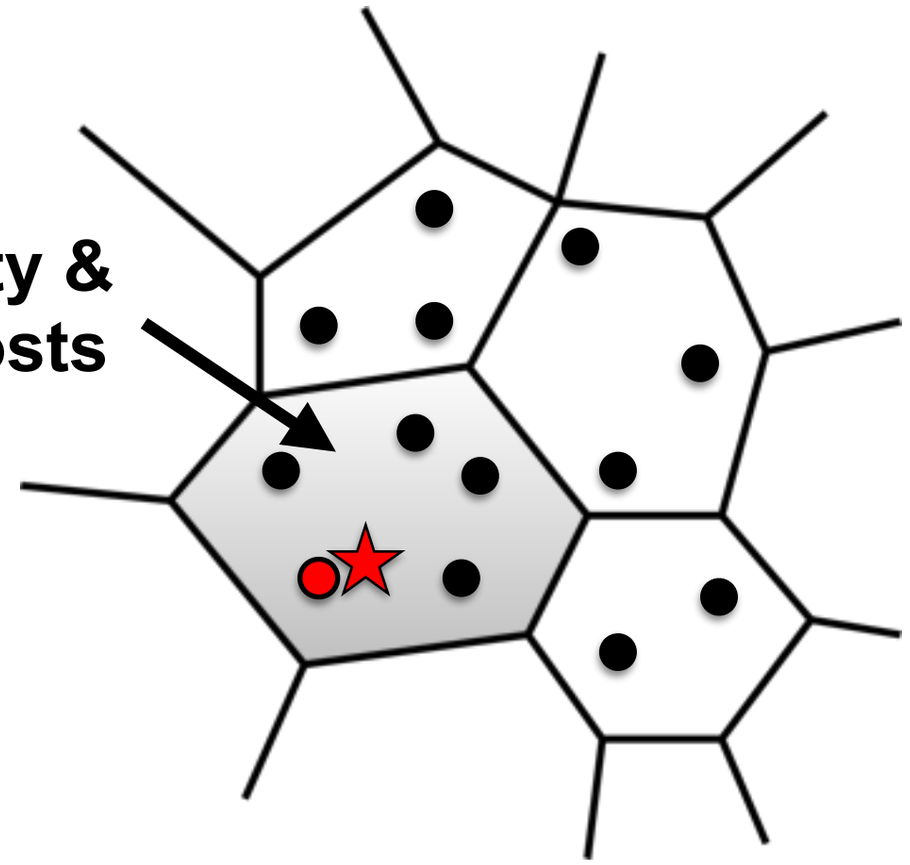
Fixed & Precomputed Visual Vocabulary

Efficient computation of p_i, c_i

- Precompute probabilities for **regions** in descriptor space

Probabilistic 2D-to-3D Search

Constant probability & constant search costs



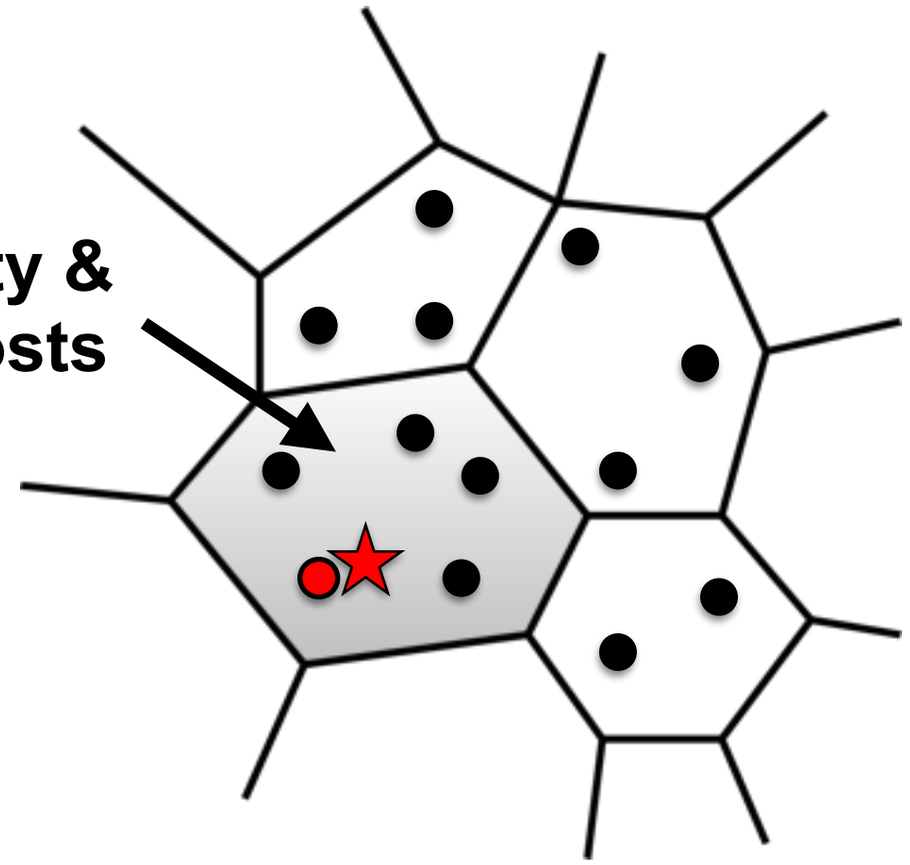
Fixed & Precomputed Visual Vocabulary

Efficient computation of p_i, c_i

- Precompute probabilities for **regions** in descriptor space
- Limit nearest neighbor search to same cell (**Quantized Search**)

Probabilistic 2D-to-3D Search

Constant probability & constant search costs



Fixed & Precomputed Visual Vocabulary

Efficient computation of p_i, c_i

- Precompute probabilities for **regions** in descriptor space
- Limit nearest neighbor search to same cell (**Quantized Search**)
- Computation in constant time for fixed-size vocabulary

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:

Probabilistic 2D-to-3D Search

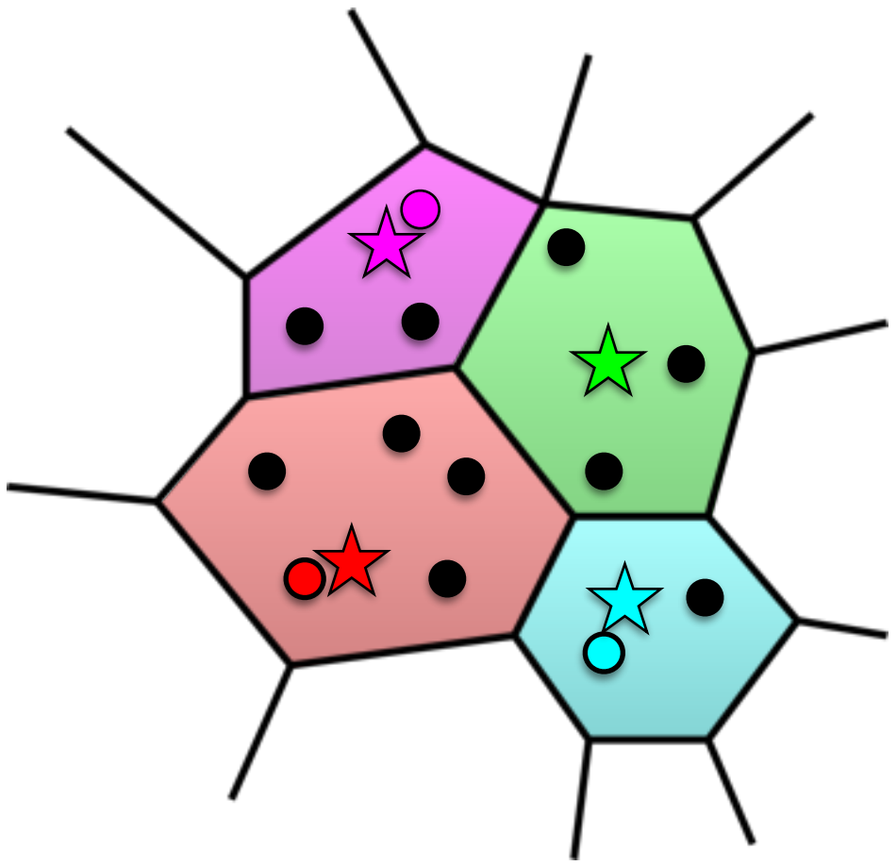
- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with $p_i > 0$ in ascending order of search costs

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with $p_i > 0$ in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$

Probabilistic 2D-to-3D Search

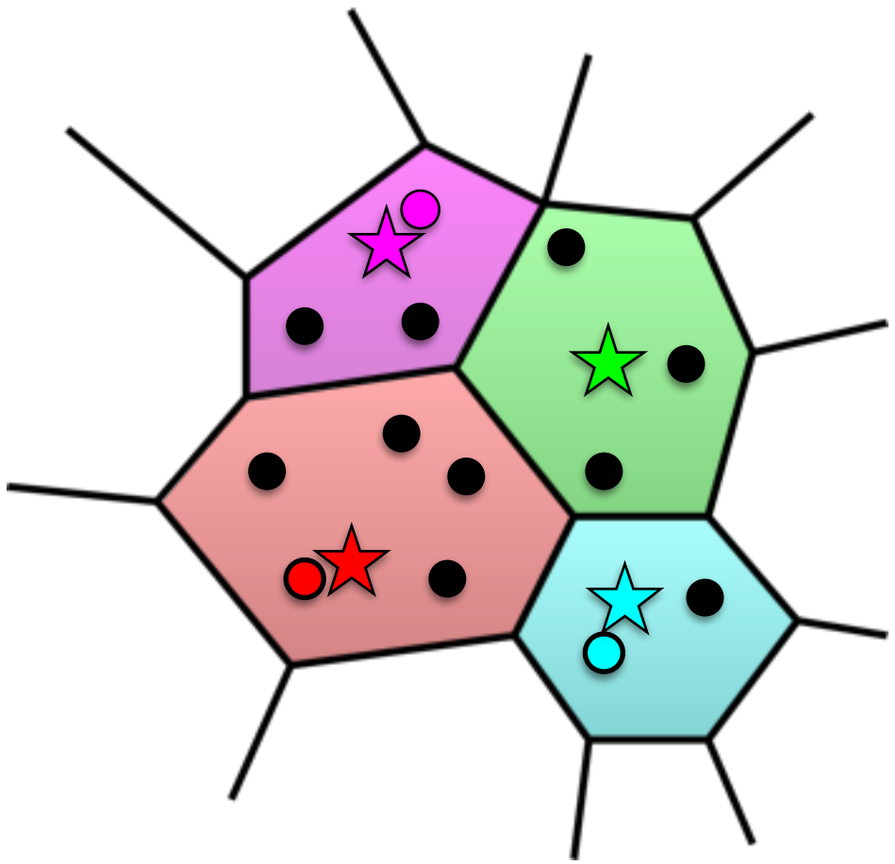
- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$



Resulting order:

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$

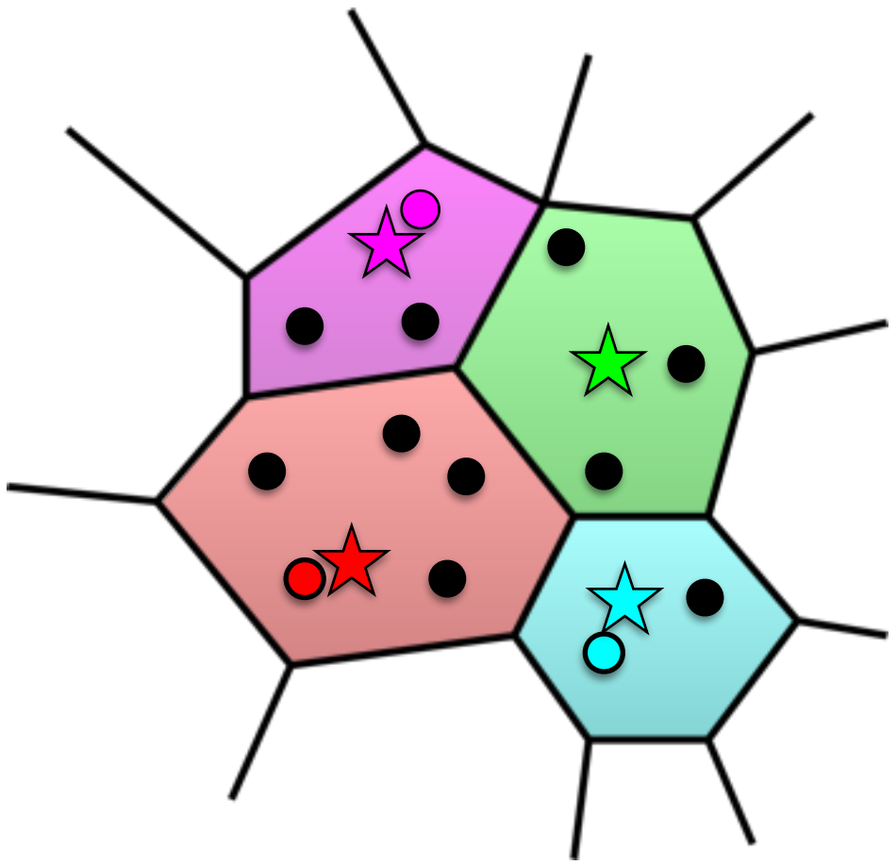


Resulting order:



Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$

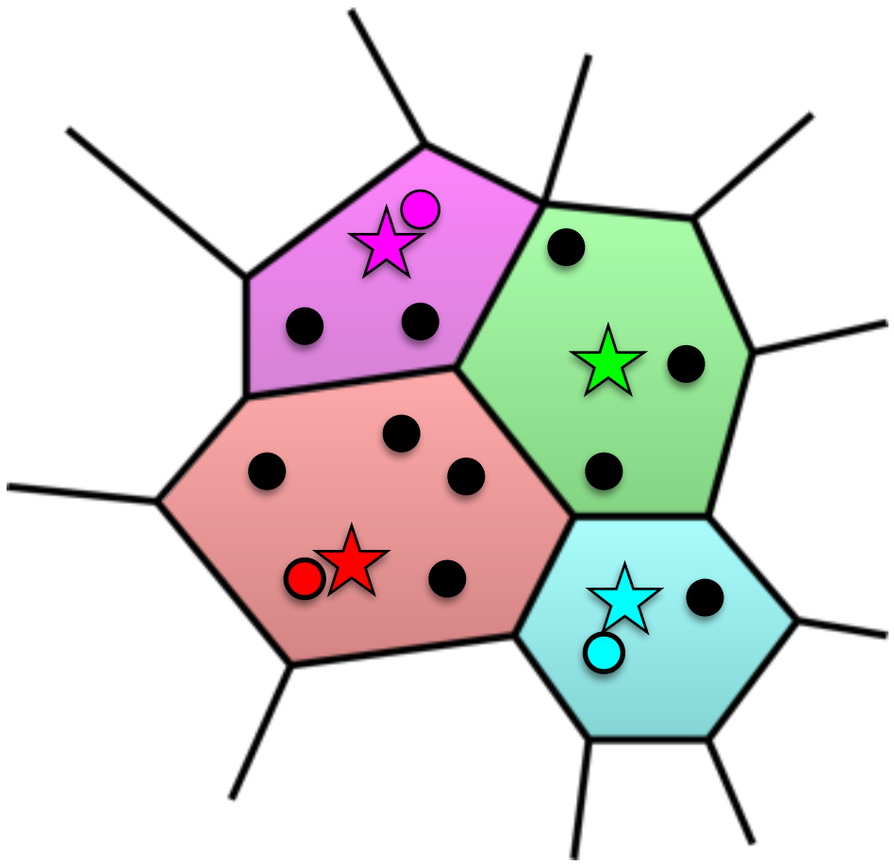


Resulting order:



Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$

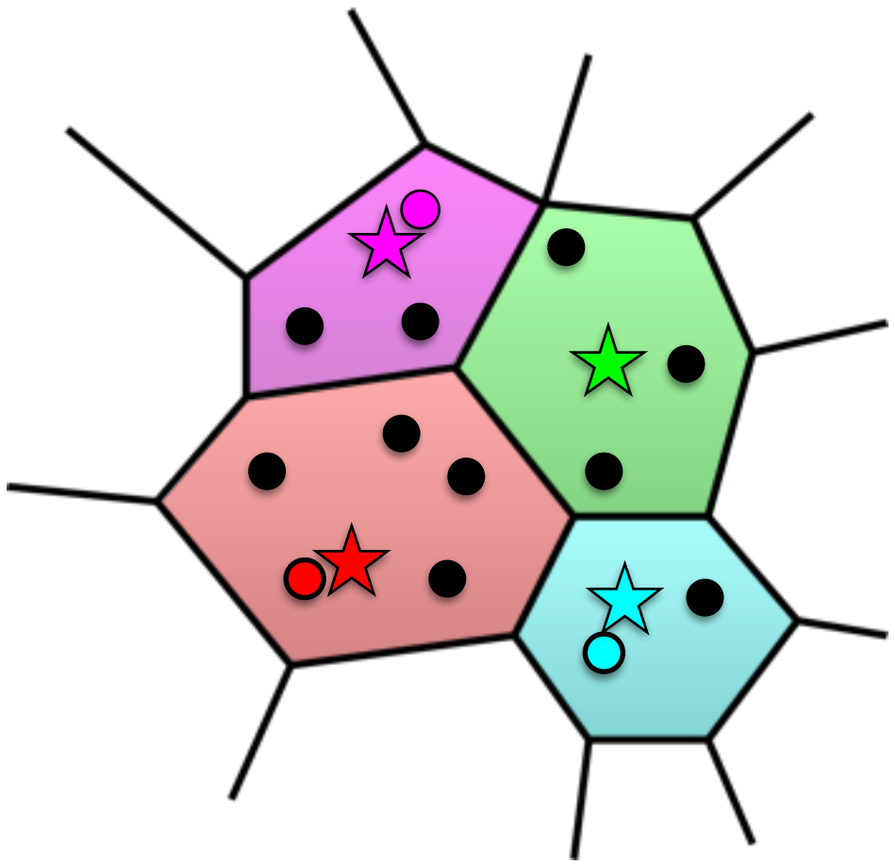


Resulting order:



Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$

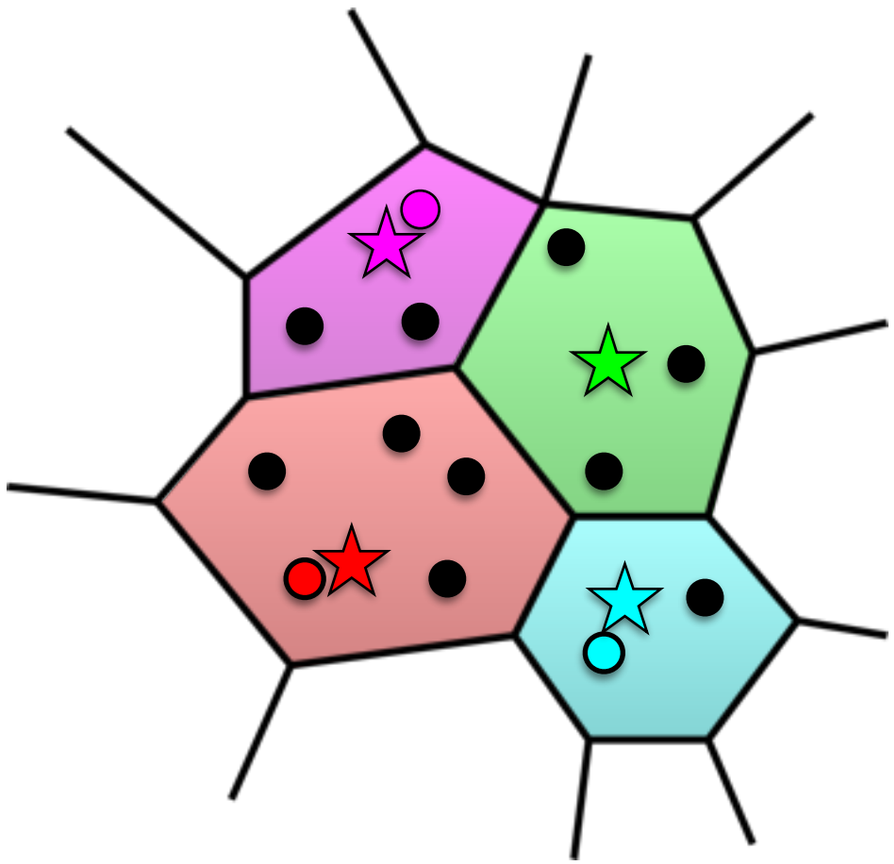


Resulting order:



Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$



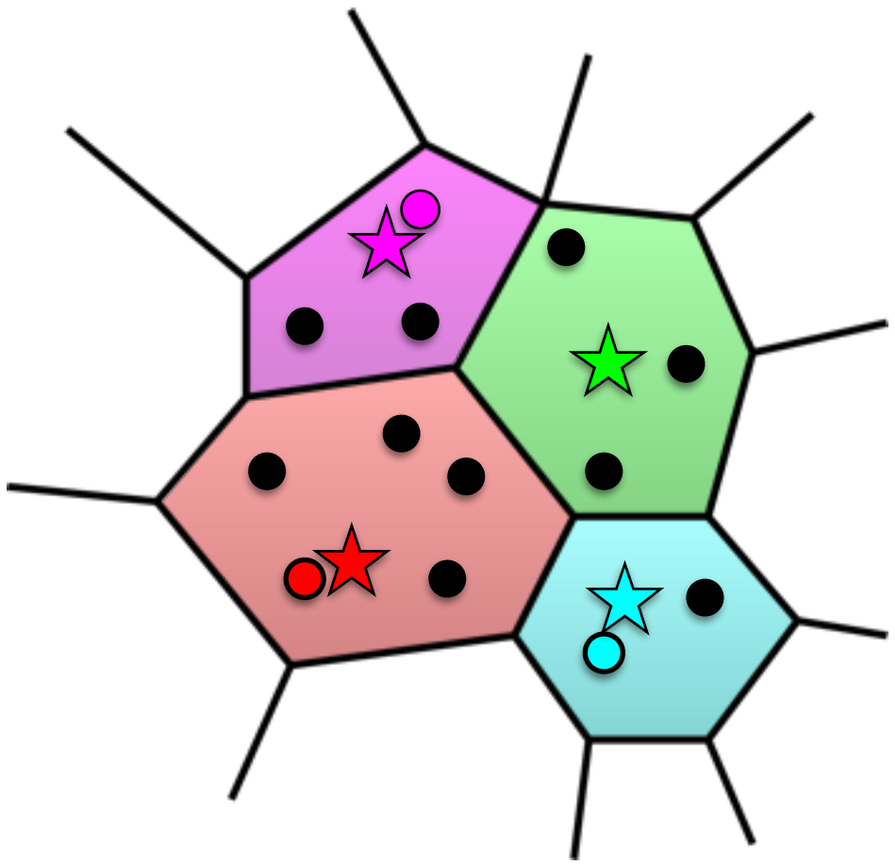
Resulting order:



Globally unique

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with $p_i > 0$ in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$



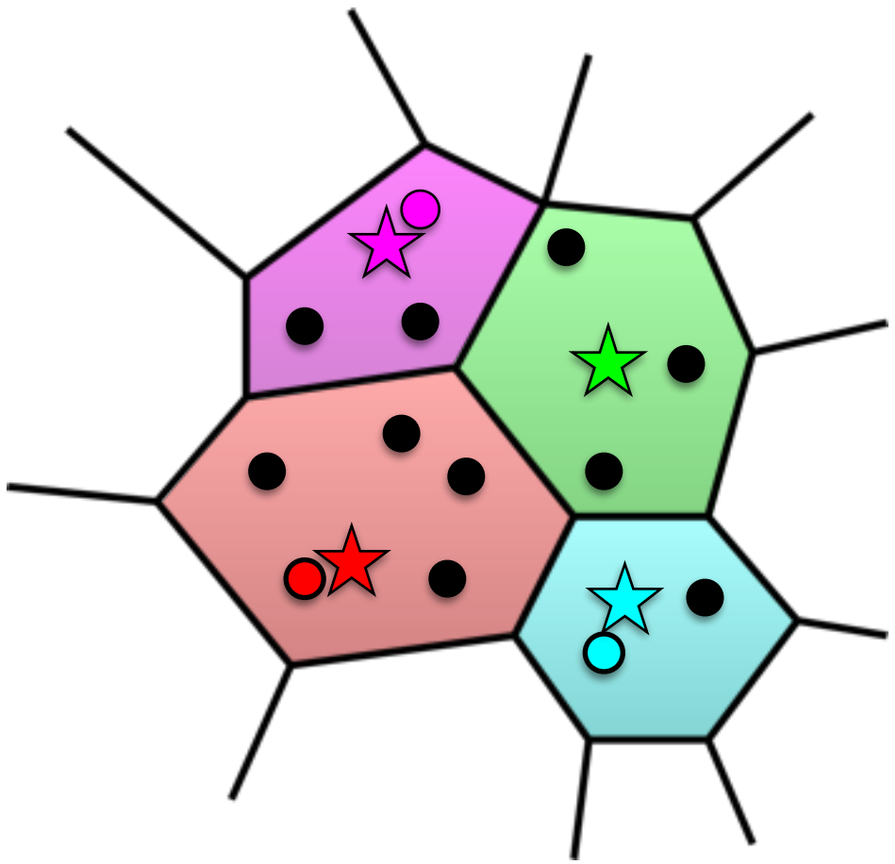
Resulting order:



Globally unique → **Highly ambiguous**

Probabilistic 2D-to-3D Search

- Solving $\min \sum_i X_i c_i \text{ s.t. } \sum_i X_i p_i \geq N_t$ is **NP-complete**
- Simple Greedy strategy:
  - Sort features with p_i>0 in ascending order of search costs
  - Select first m features such that $\sum_{i=1}^{m} p_i \geq N_t$



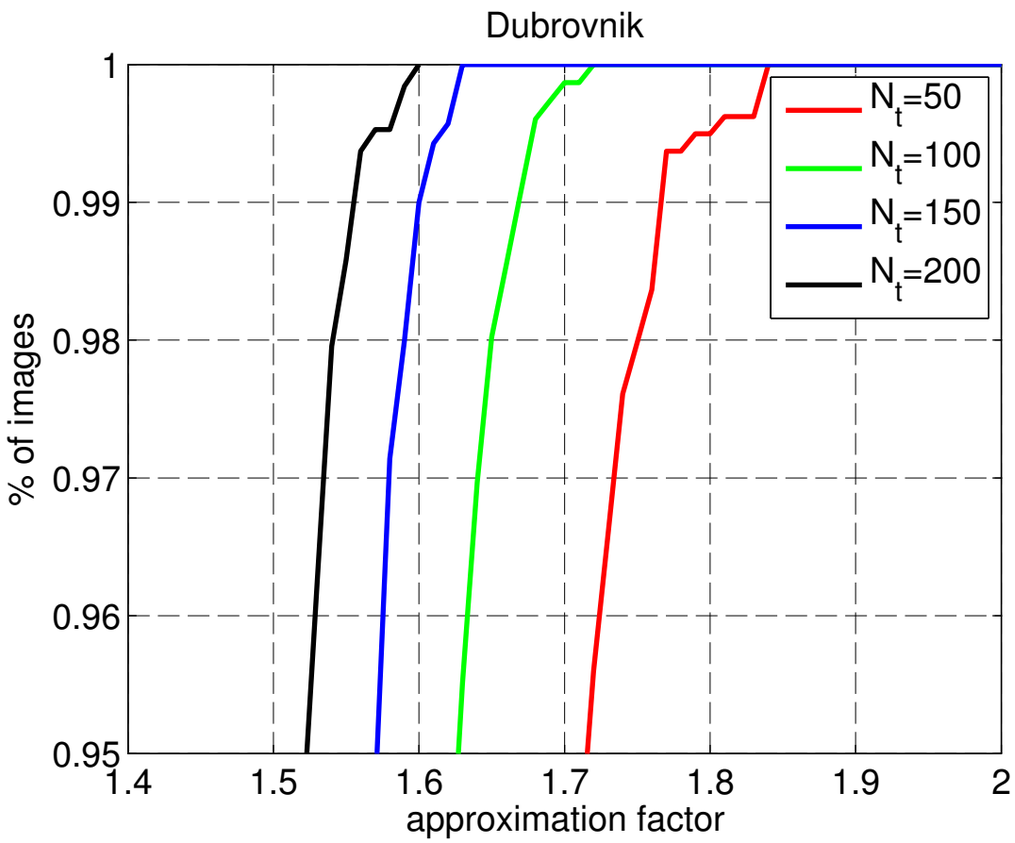
Resulting order:

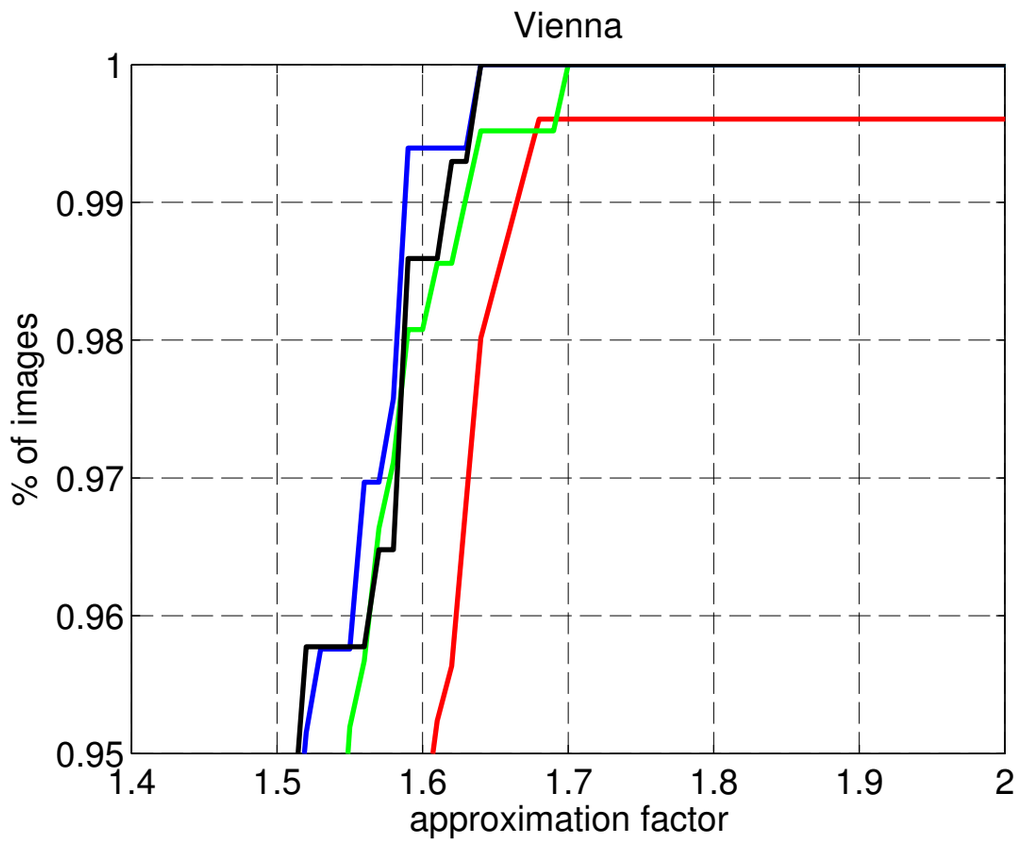


Globally unique → **Highly ambiguous**

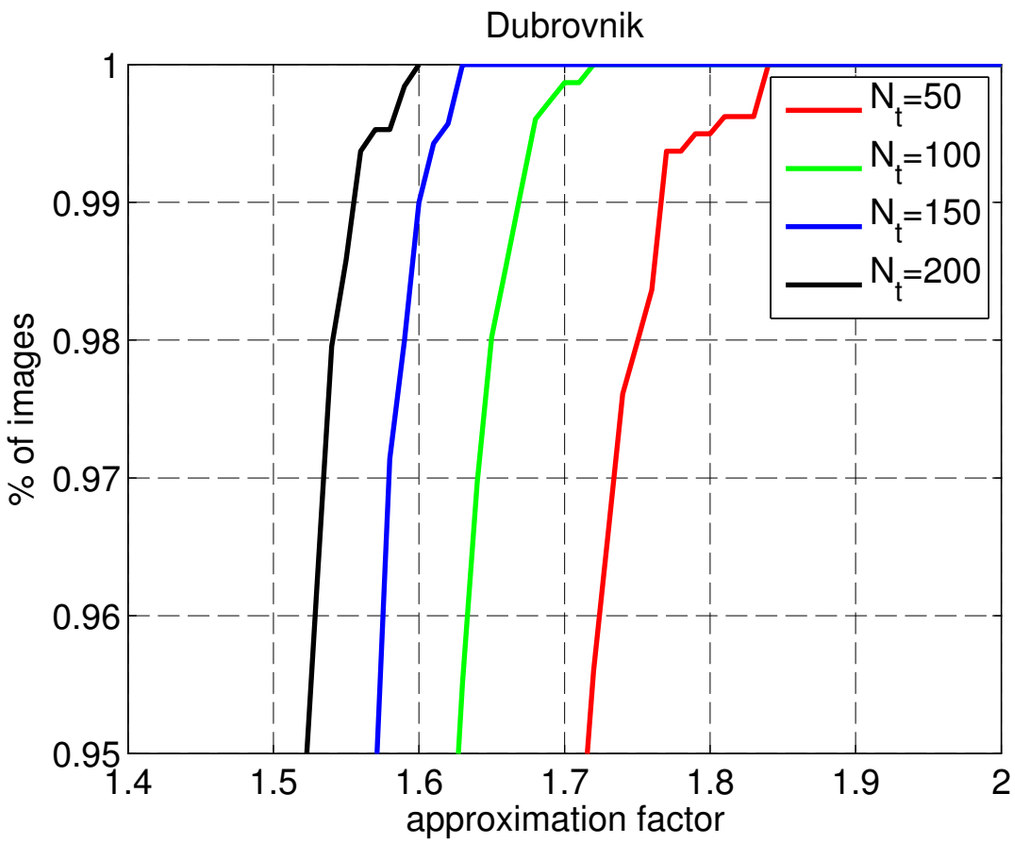
Similar to idf weighting [Sivic, Zisserman, ICCV'03]

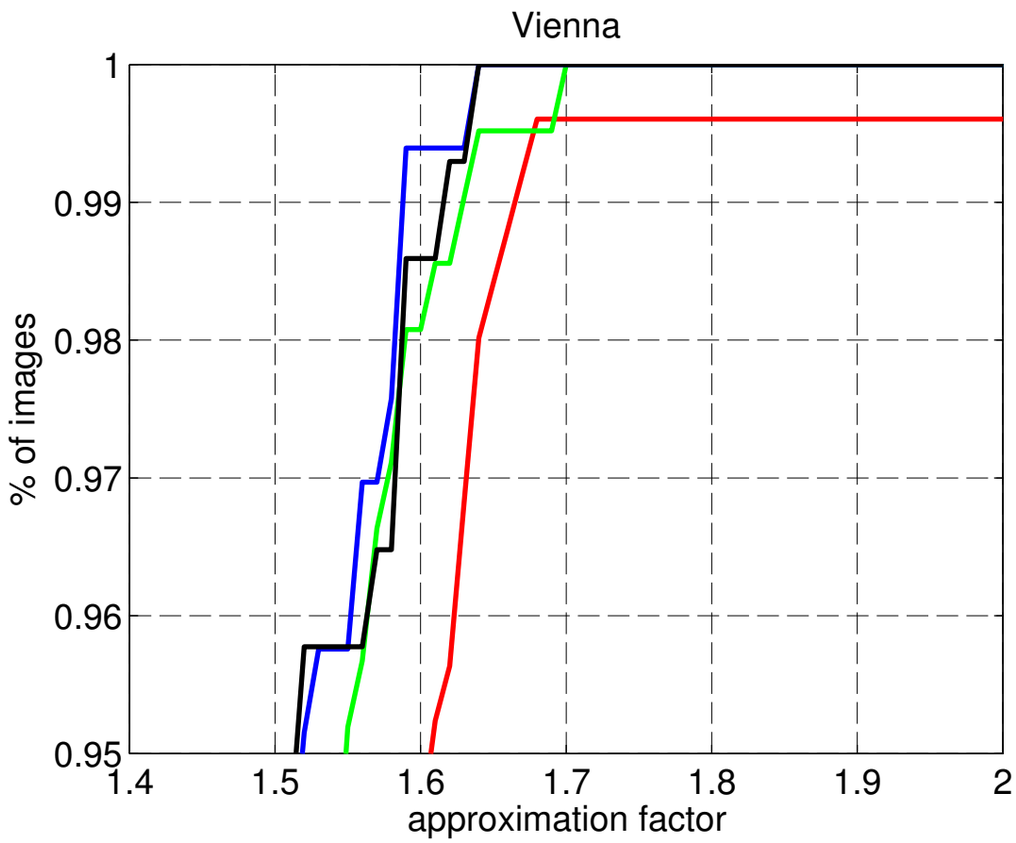
Greedy vs. OPT





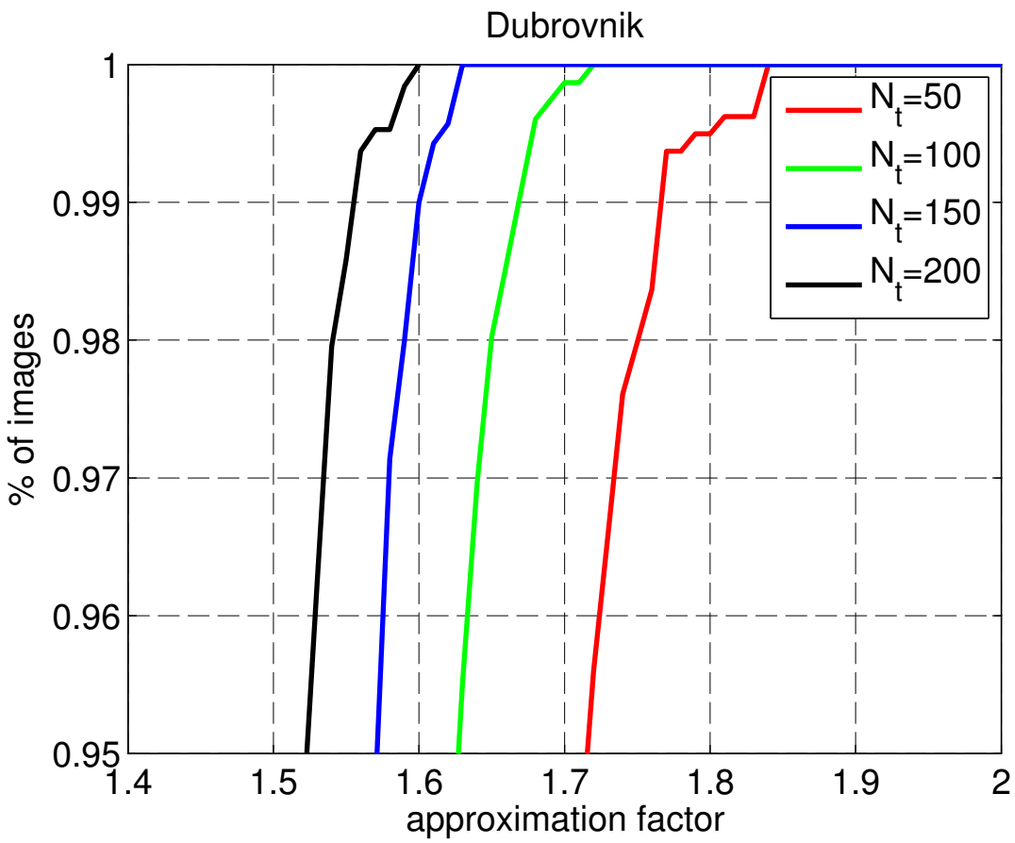
Greedy vs. OPT

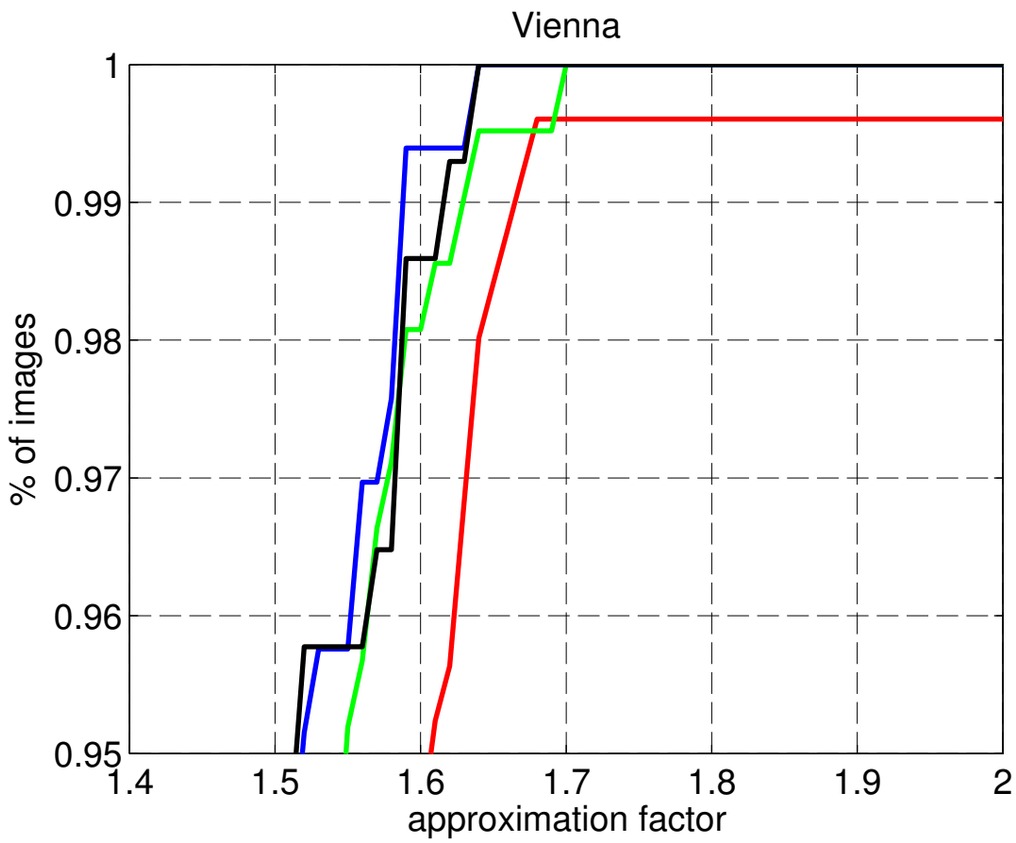




- Greedy performs close to optimal!

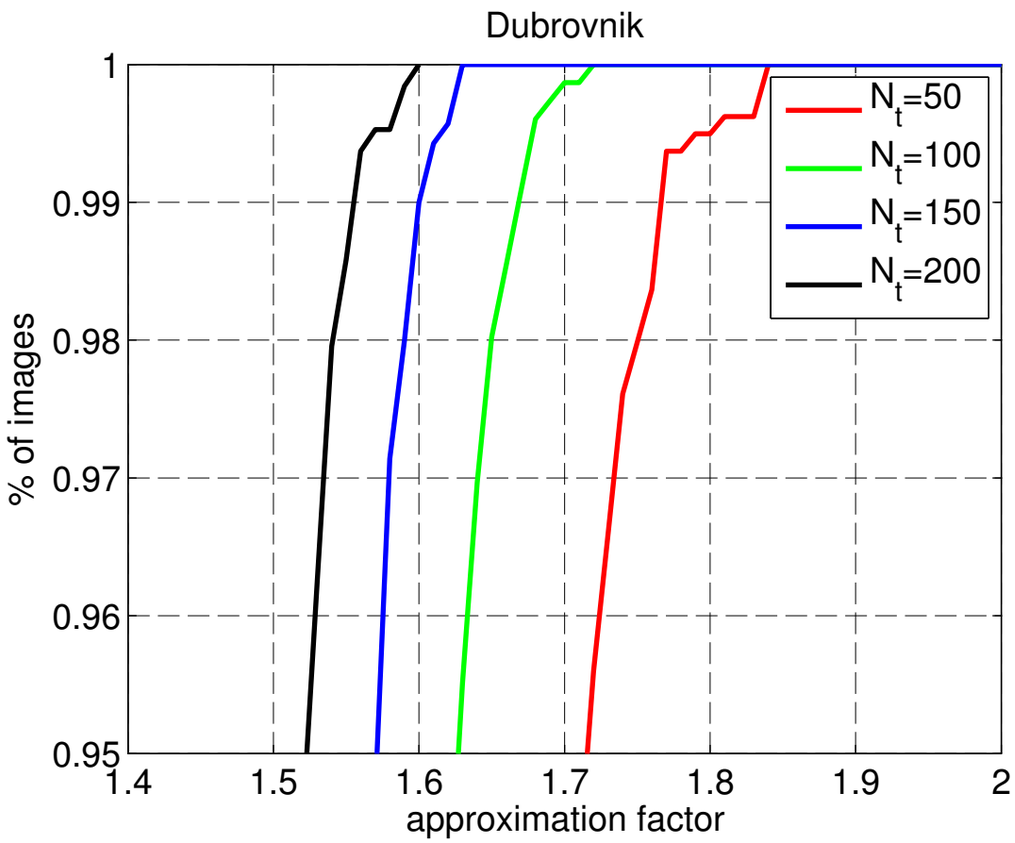
Greedy vs. OPT

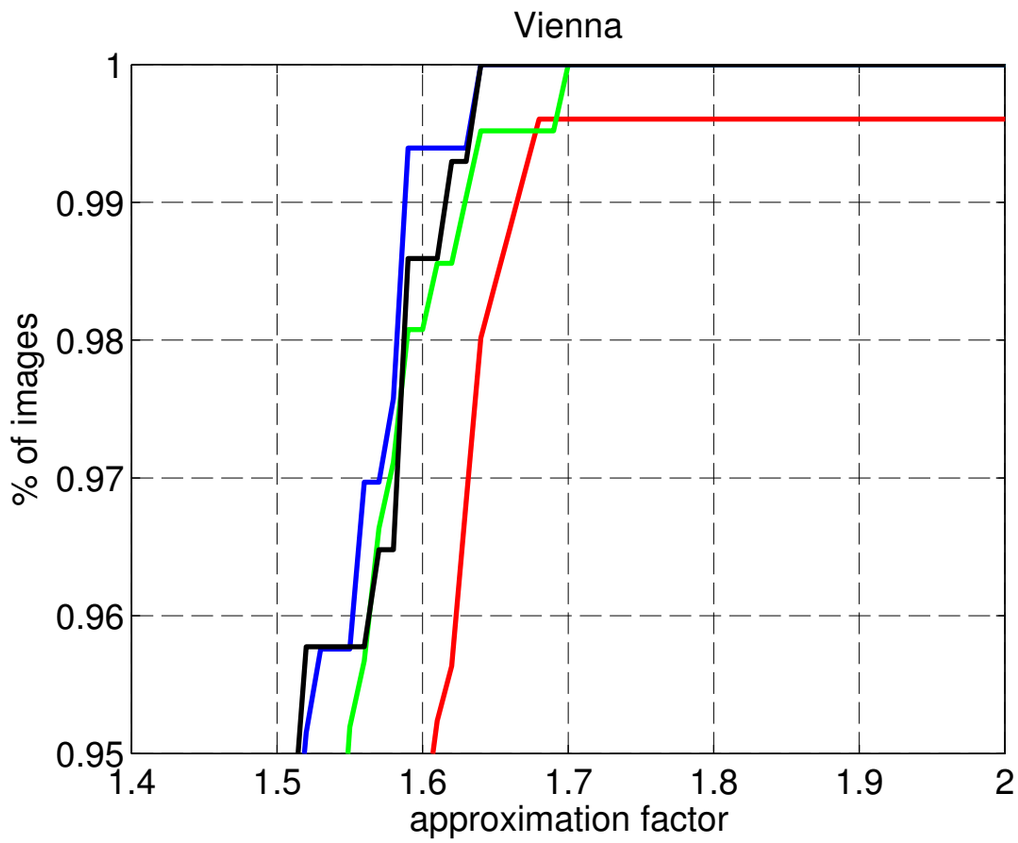




- Greedy performs close to optimal!
- Here: Probabilities learnt from query images

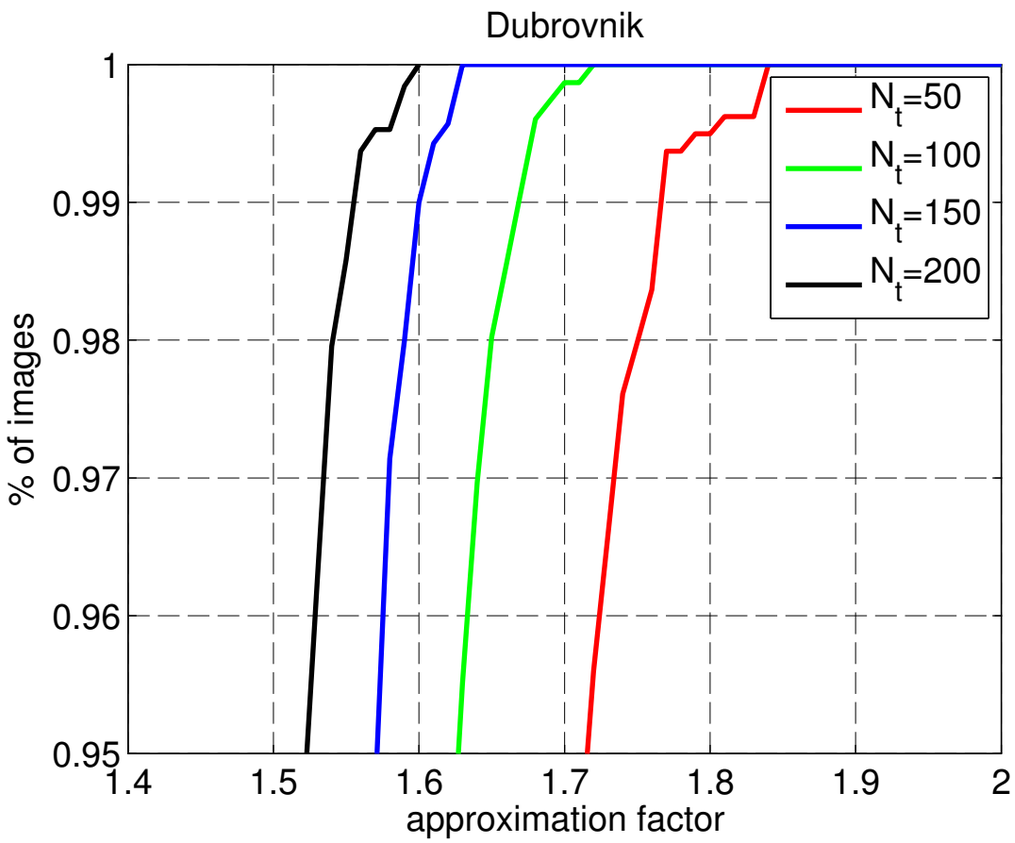
Greedy vs. OPT

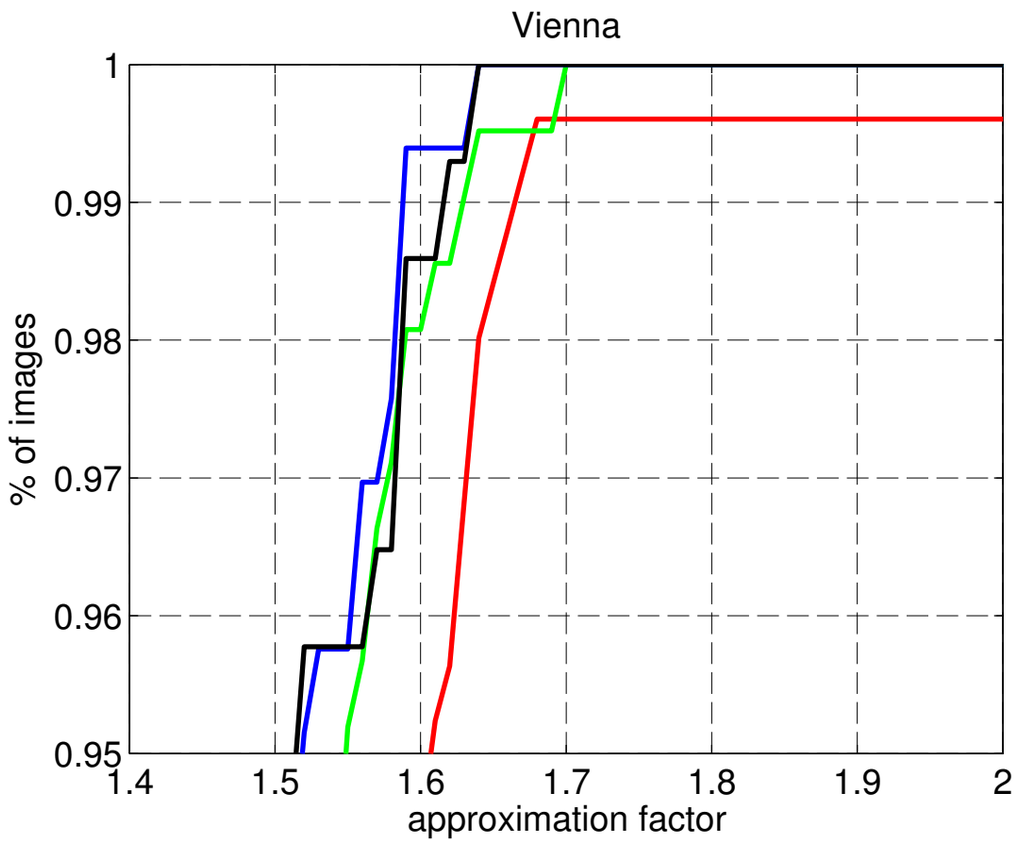




- Greedy performs close to optimal!
- Here: Probabilities learnt from query images
- In practice: Hard to find good training data

Greedy vs. OPT



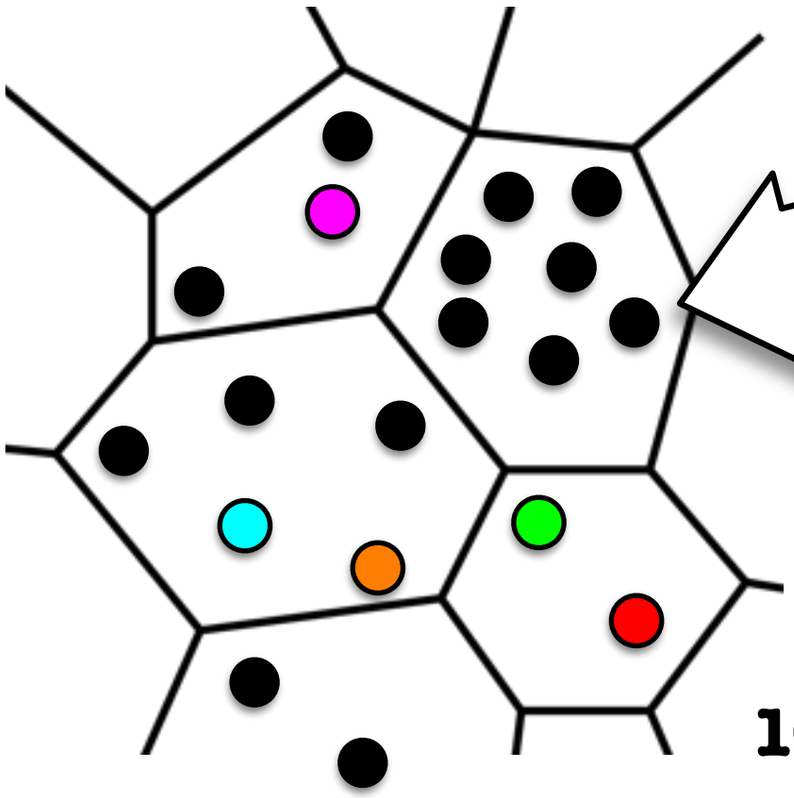


- Greedy performs close to optimal!
- Here: Probabilities learnt from query images
- In practice: Hard to find good training data
  - … but Greedy does not really need it!

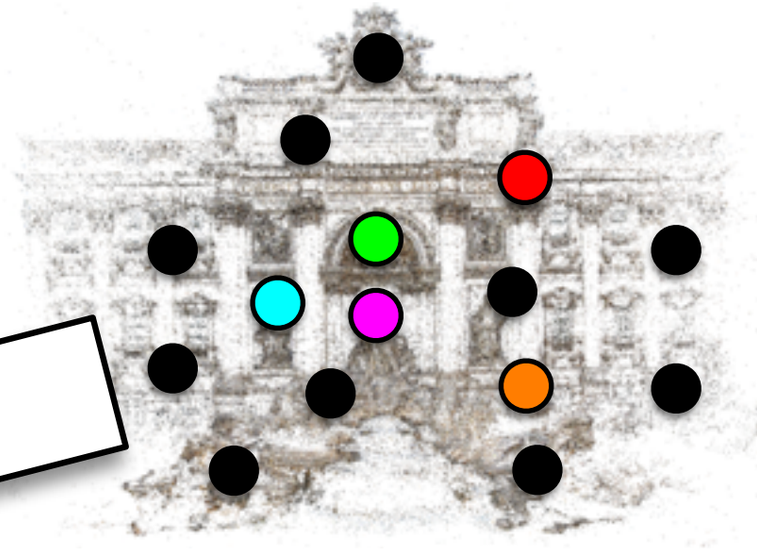
Vocabulary-Based Prioritized Search (VPS)



Query Image



100k words



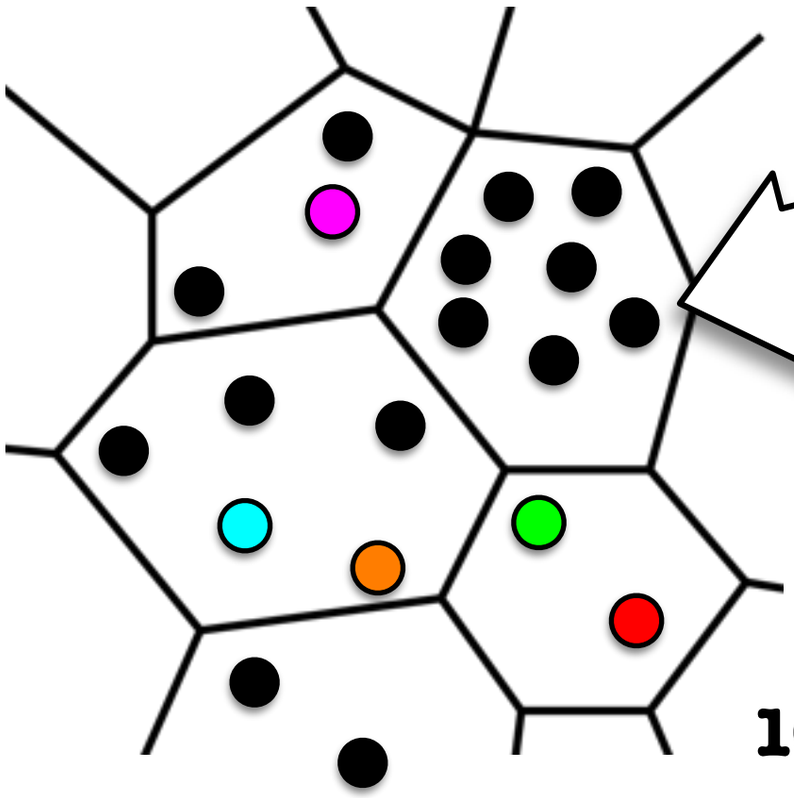
3D Model

[Sattler et al., ICCV'11] [code]

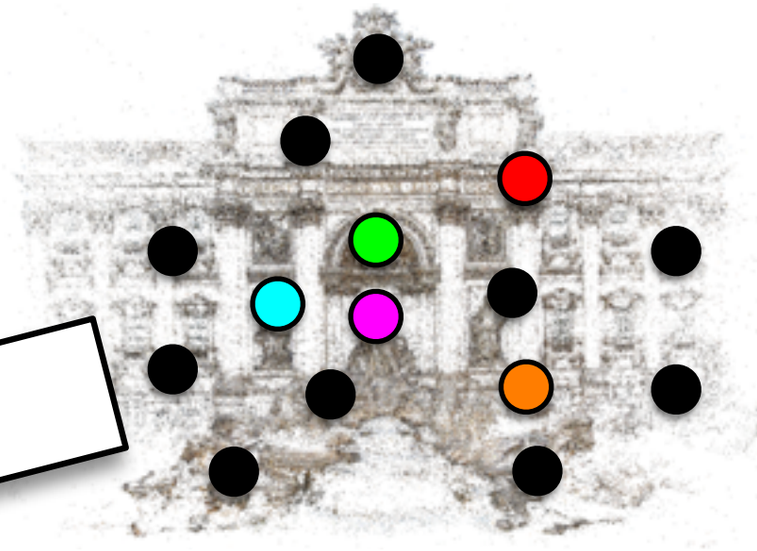
Vocabulary-Based Prioritized Search (VPS)



Query Image



100k words



3D Model

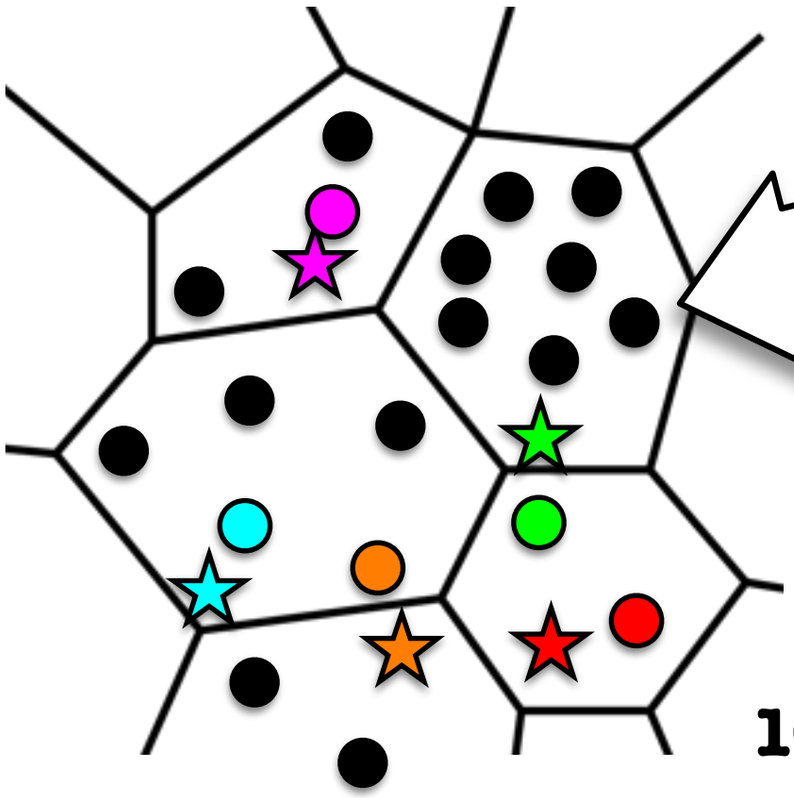
Assign features to words

[Sattler et al., ICCV'11] [code]

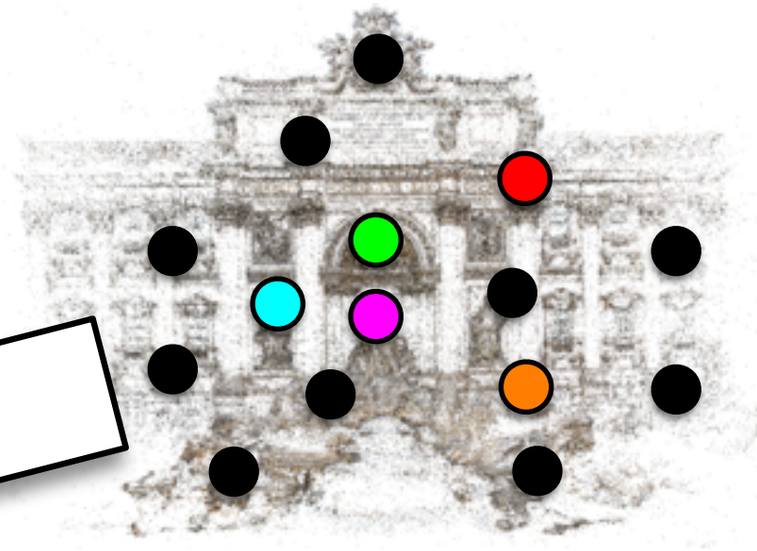
Vocabulary-Based Prioritized Search (VPS)



Query Image



100k words



3D Model

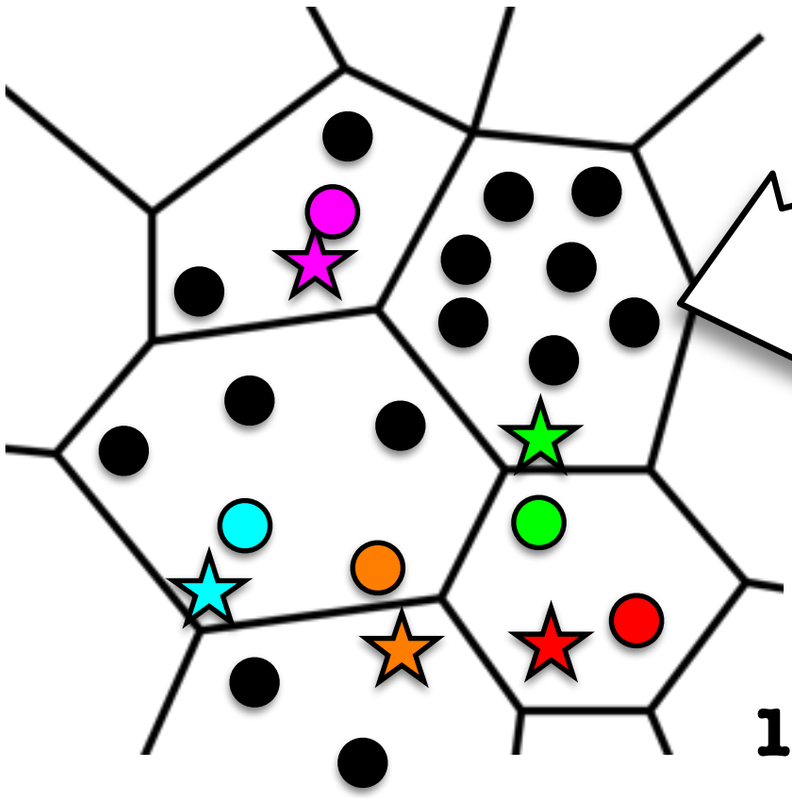
Assign features to words

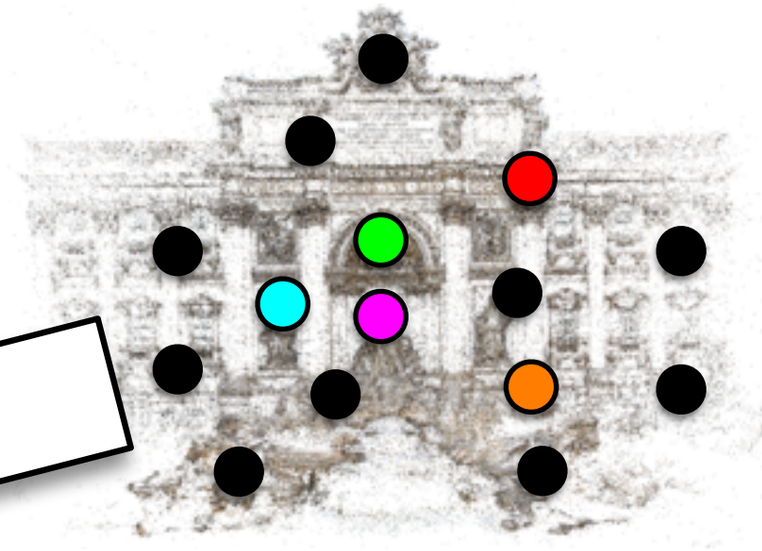
[Sattler et al., ICCV'11] [code]

Vocabulary-Based Prioritized Search (VPS)



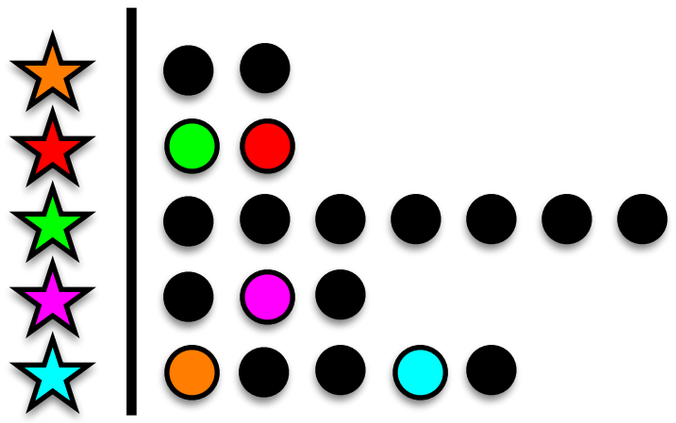
Query Image





3D Model

100k words



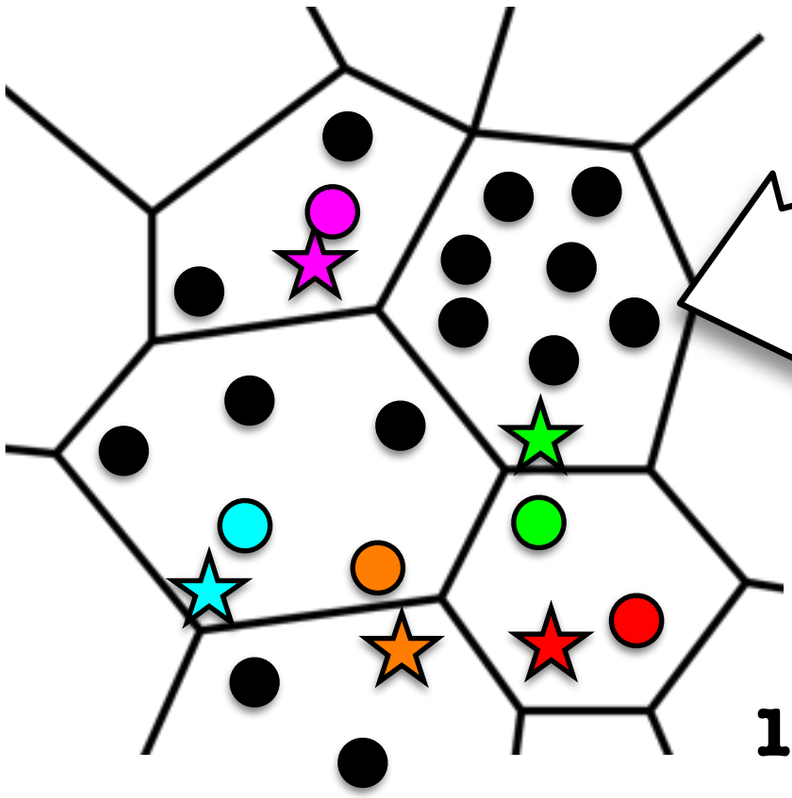
Assign features to words

[Sattler et al., ICCV'11] [code]

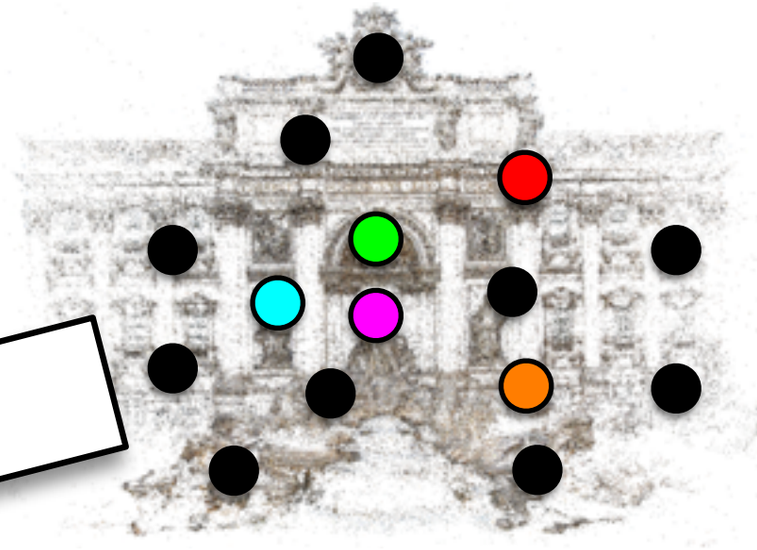
Vocabulary-Based Prioritized Search (VPS)



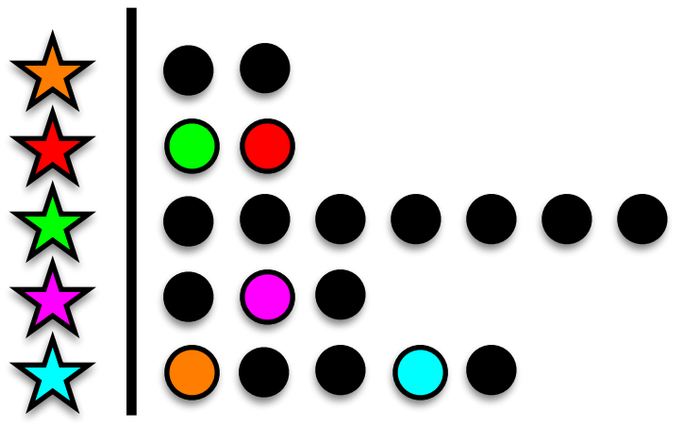
Query Image



100k words



3D Model



Assign features to words

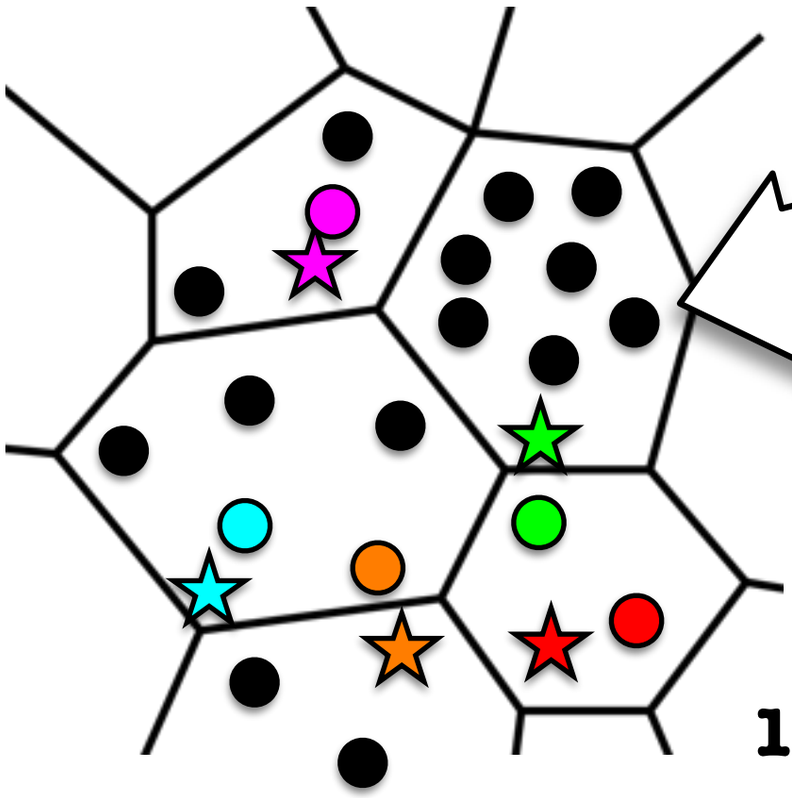
Sort based on costs

[Sattler et al., ICCV'11] [code]

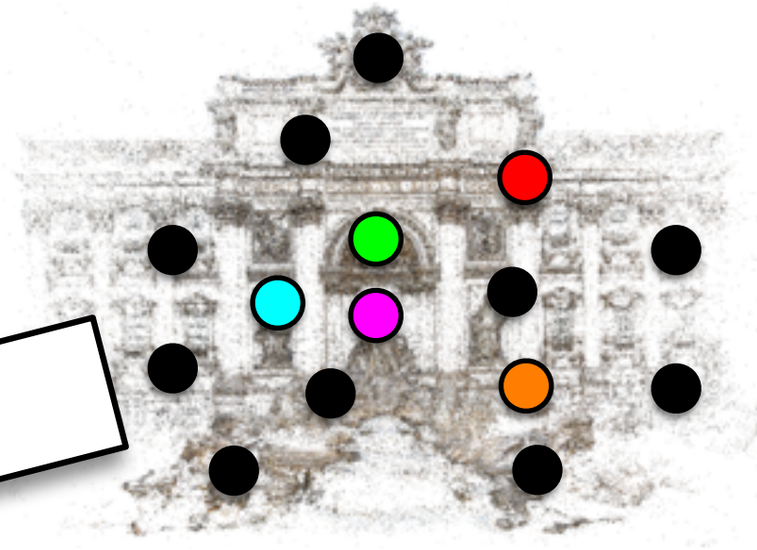
Vocabulary-Based Prioritized Search (VPS)



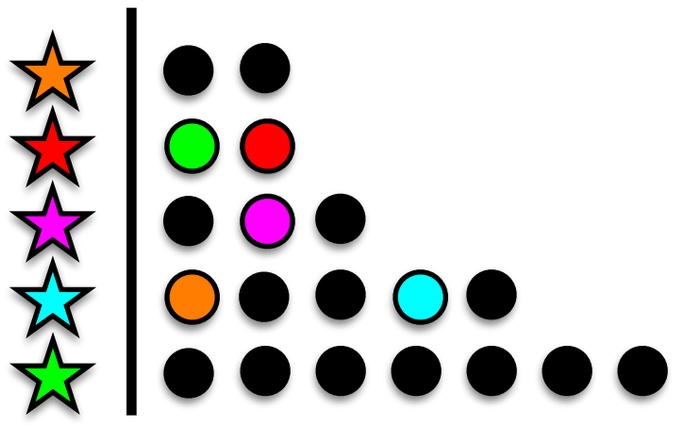
Query Image



100k words



3D Model



Assign features to words

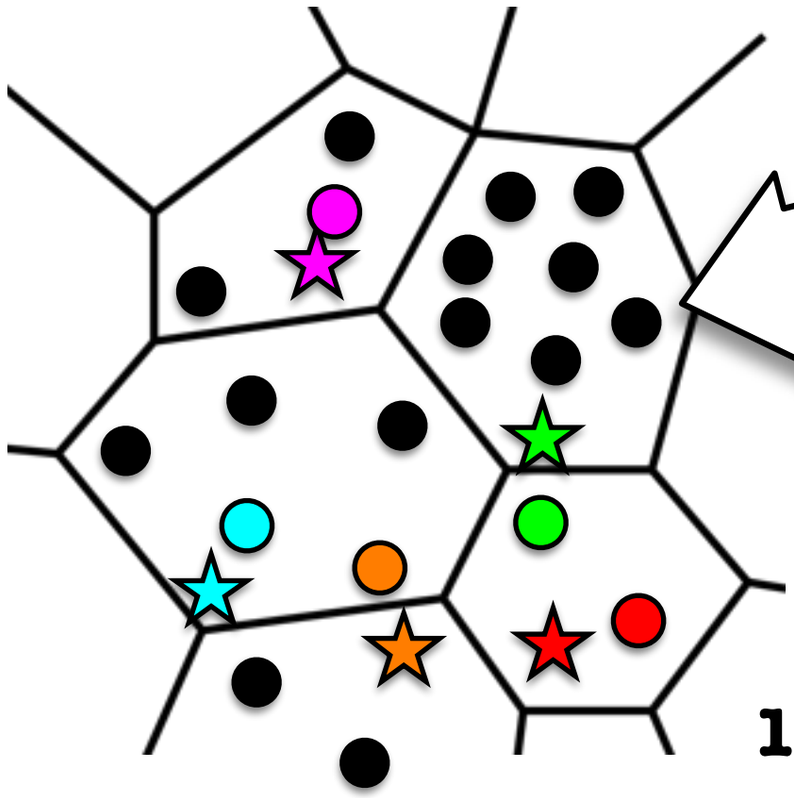
Sort based on costs

[Sattler et al., ICCV'11] [code]

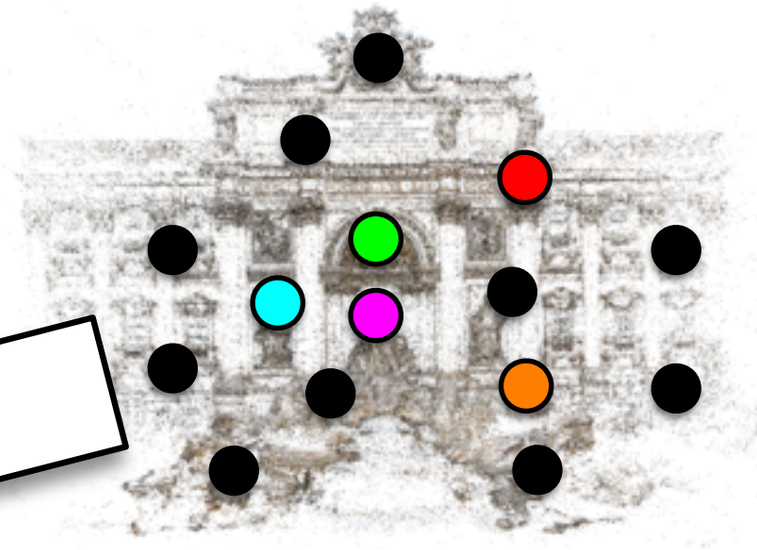
Vocabulary-Based Prioritized Search (VPS)



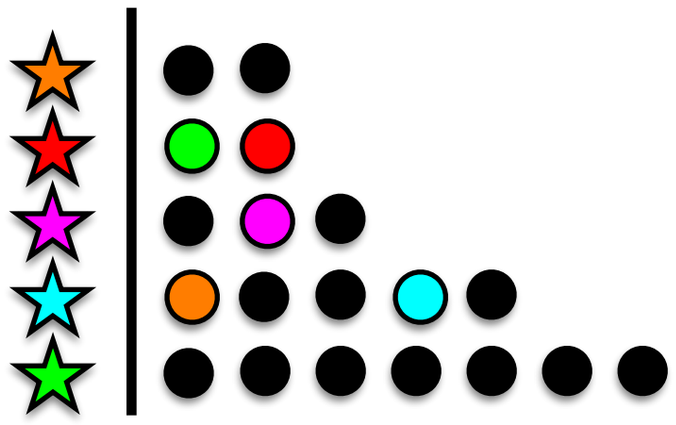
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

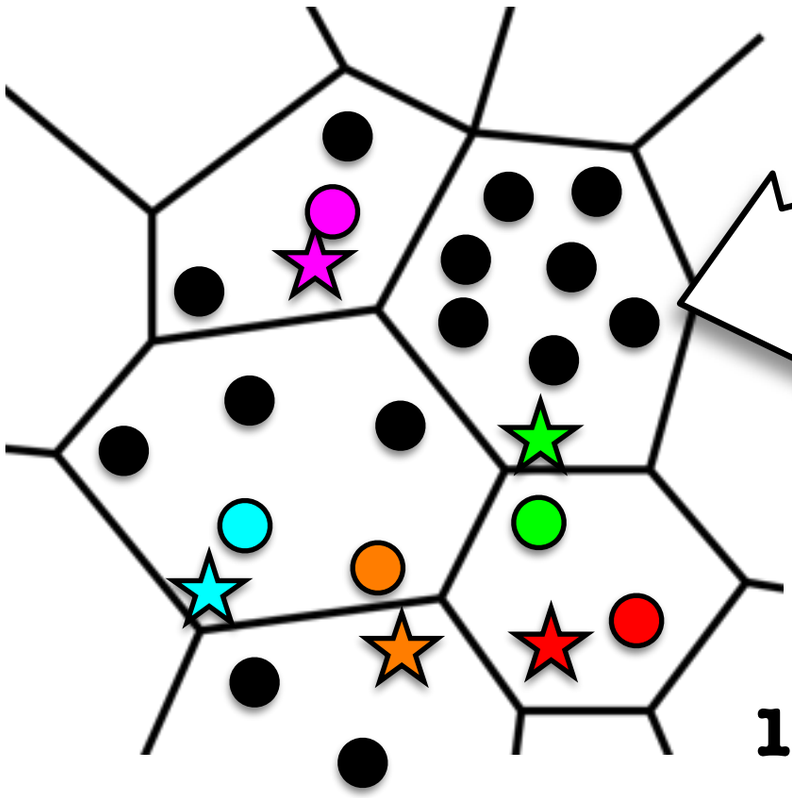
Linear search through words

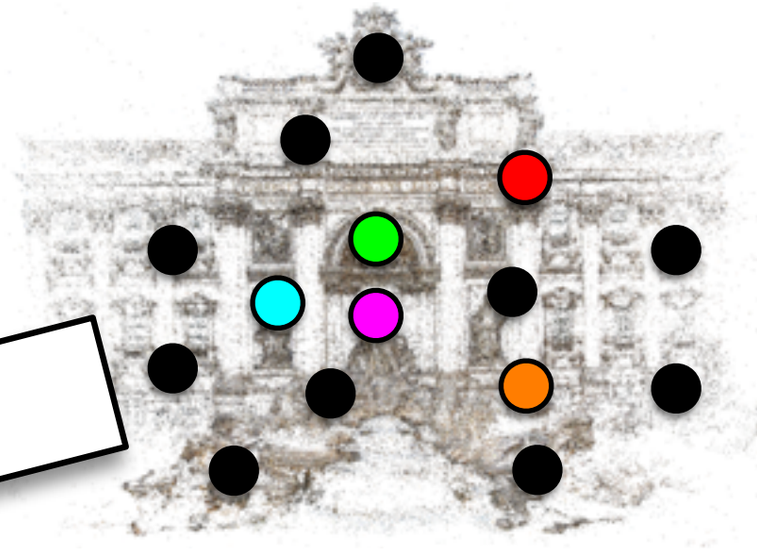
[Sattler et al., ICCV'11] [code]

Vocabulary-Based Prioritized Search (VPS)



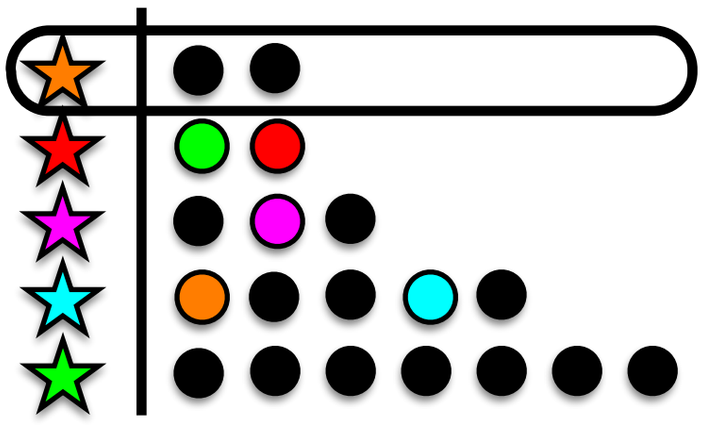
Query Image





3D Model

100k words



Assign features to words

Sort based on costs

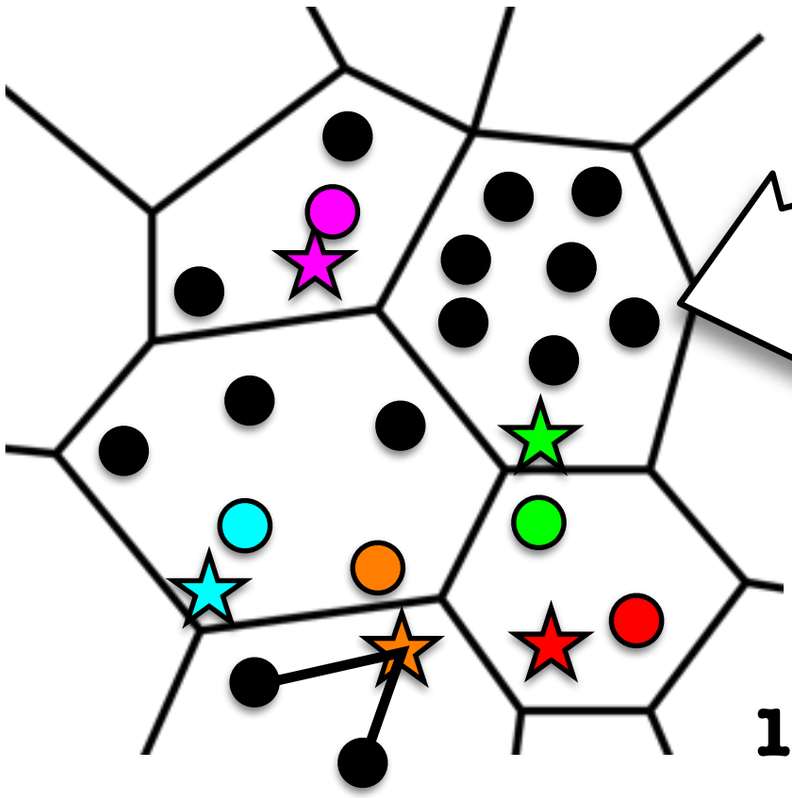
Linear search through words

[Sattler et al., ICCV'11] [code]

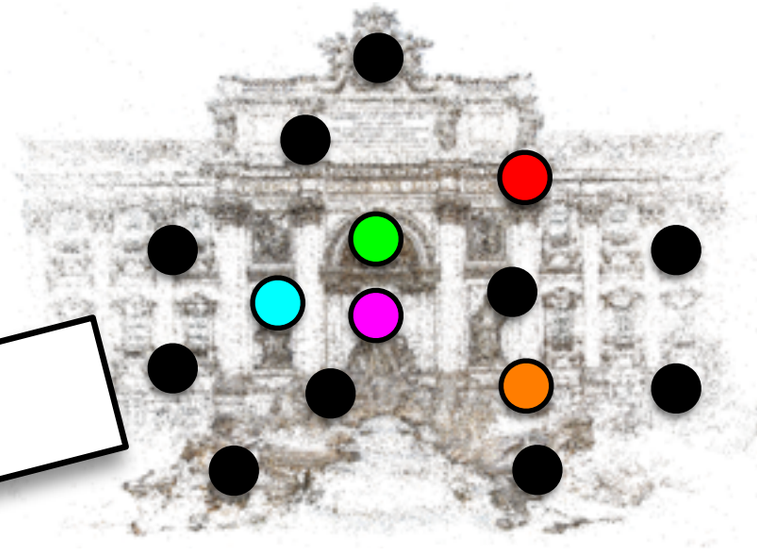
Vocabulary-Based Prioritized Search (VPS)



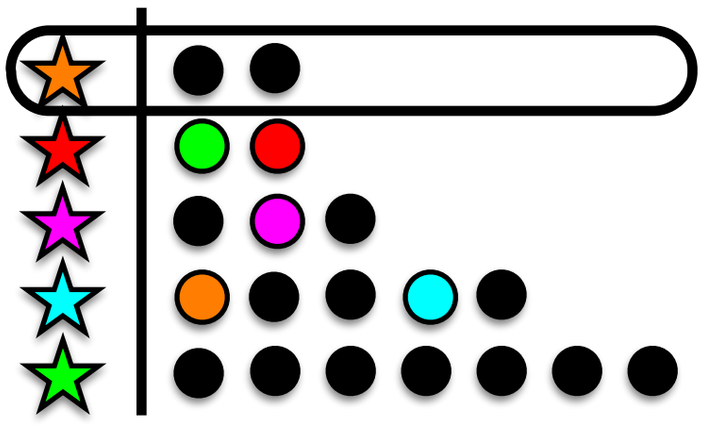
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

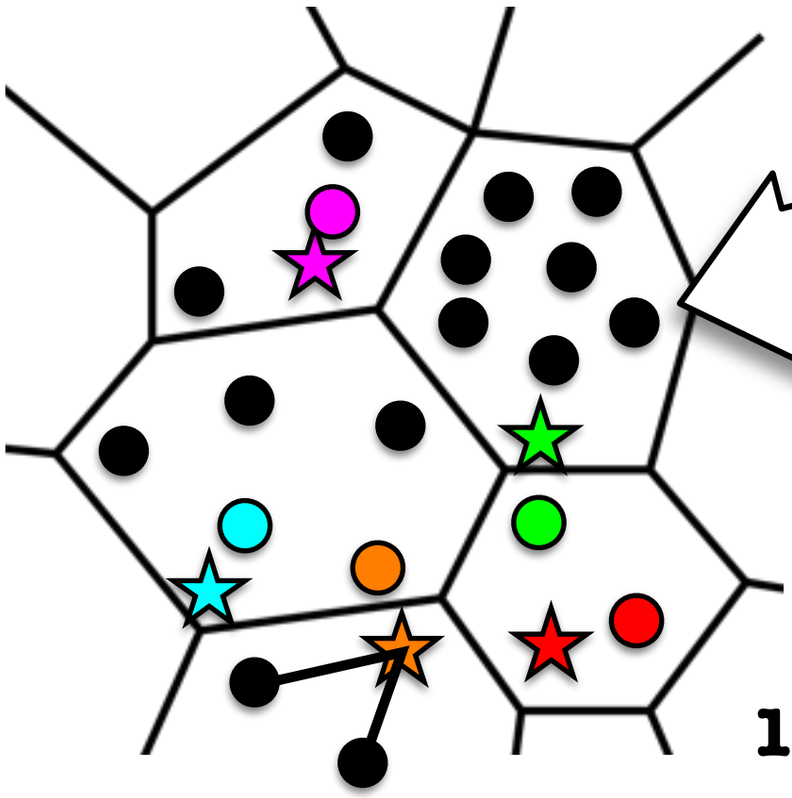
Linear search through words

[Sattler et al., ICCV'11] [code]

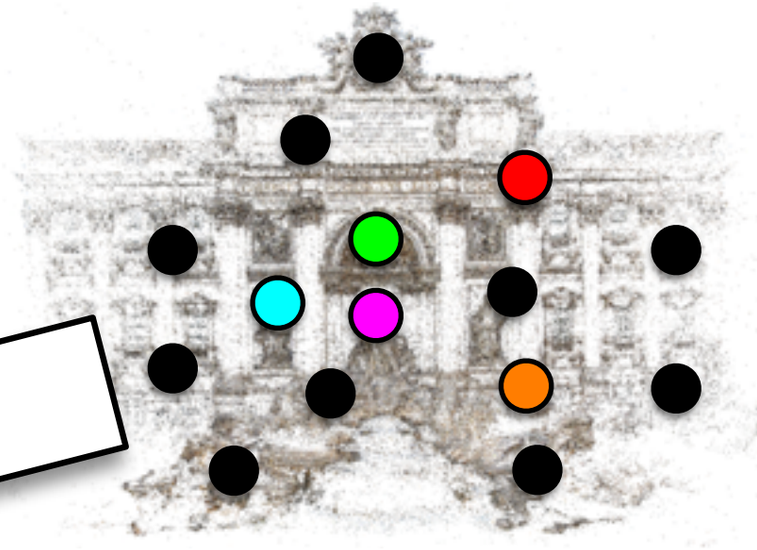
Vocabulary-Based Prioritized Search (VPS)



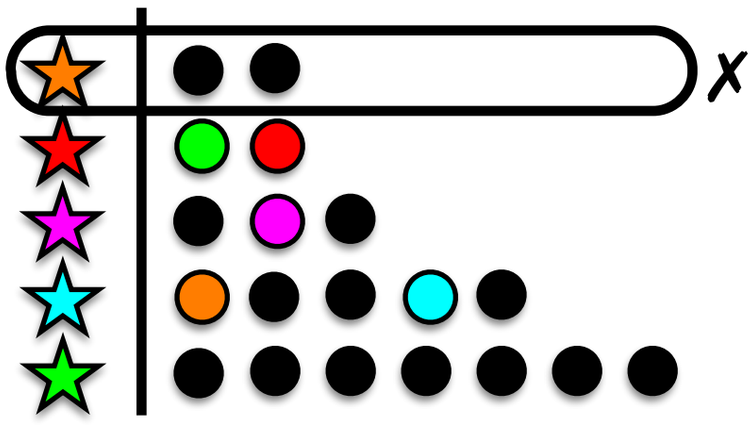
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

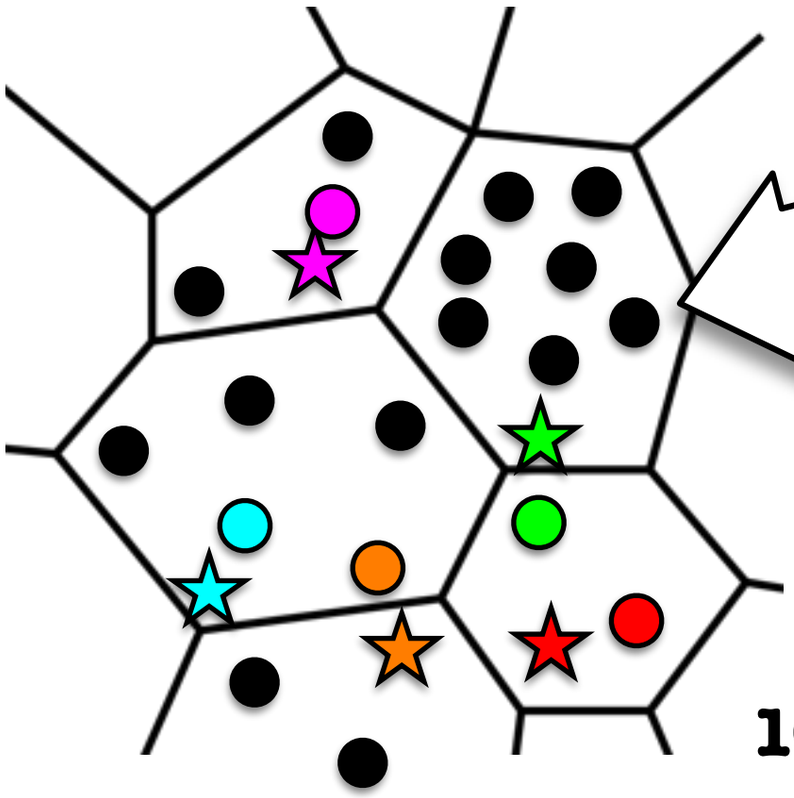
Linear search through words

[Sattler et al., ICCV'11] [code]

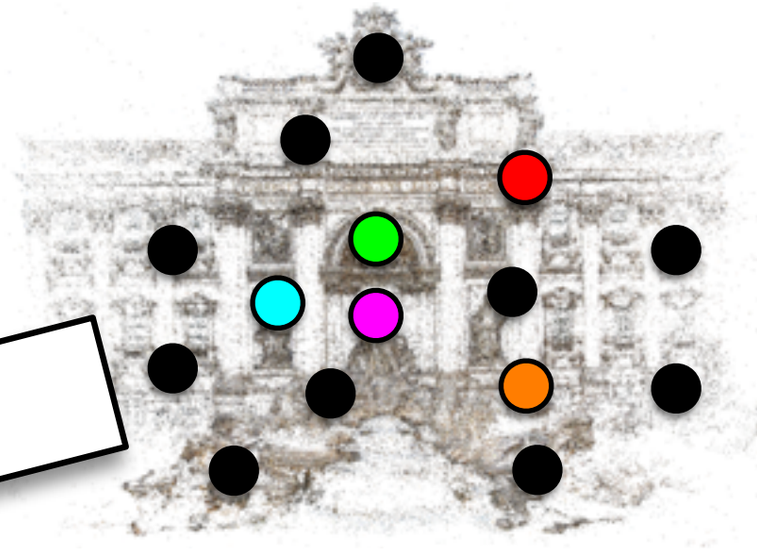
Vocabulary-Based Prioritized Search (VPS)



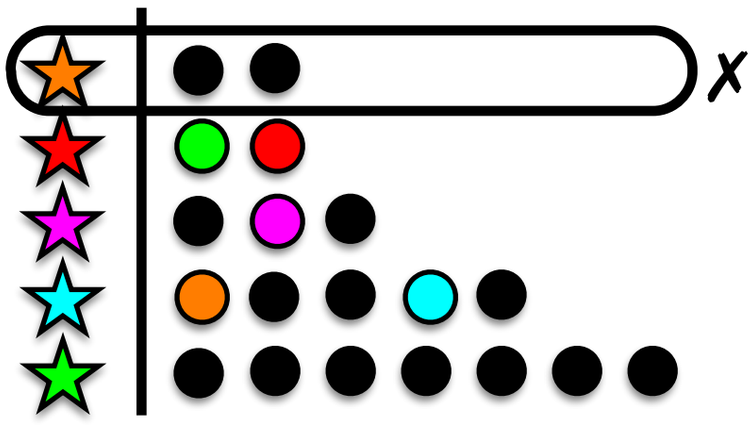
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

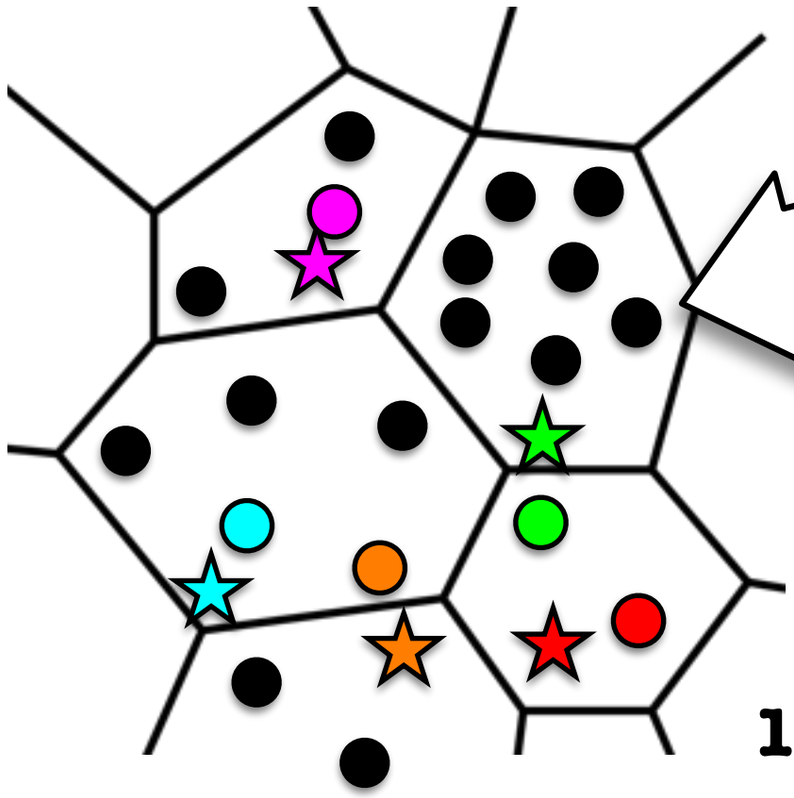
Linear search through words

[Sattler et al., ICCV'11] [code]

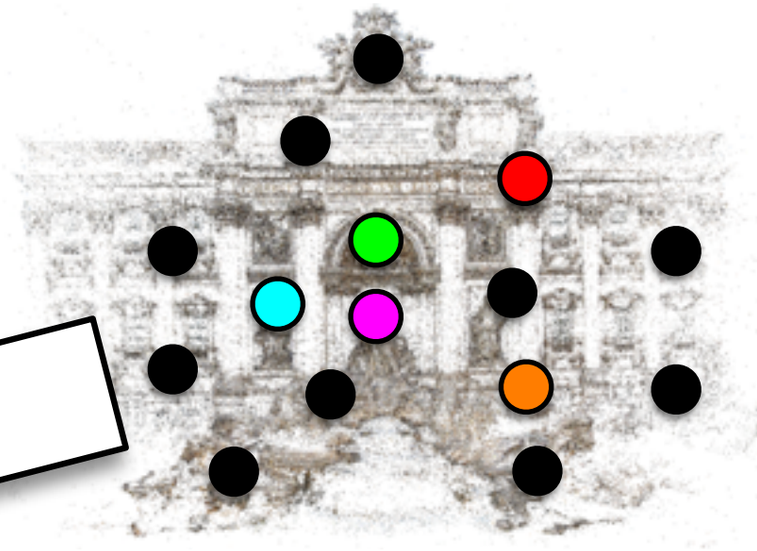
Vocabulary-Based Prioritized Search (VPS)



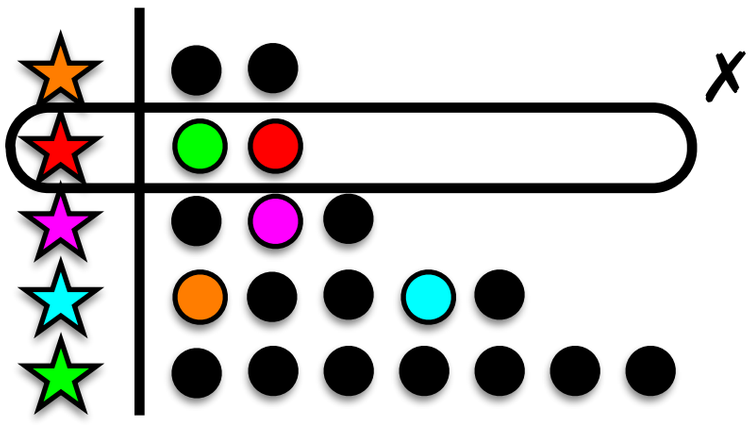
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

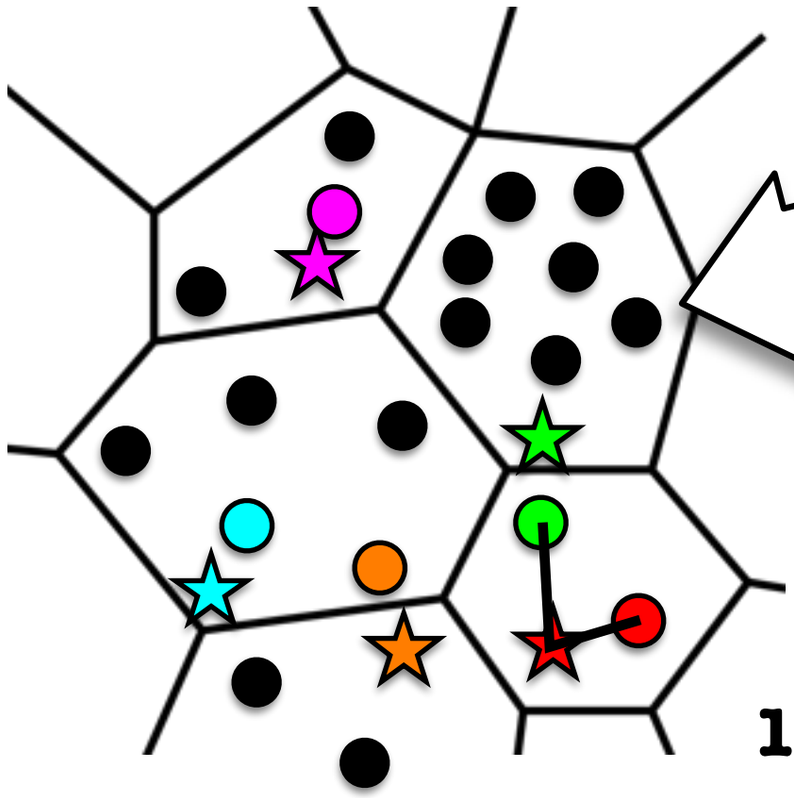
Linear search through words

[Sattler et al., ICCV'11] [code]

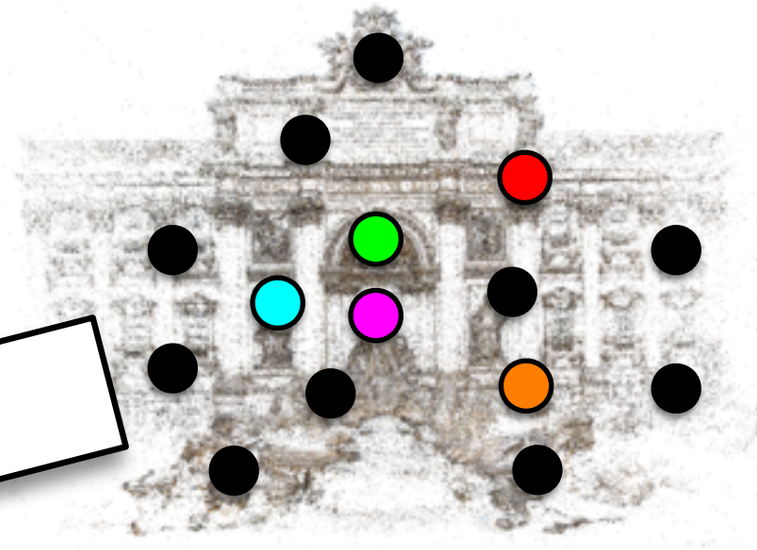
Vocabulary-Based Prioritized Search (VPS)



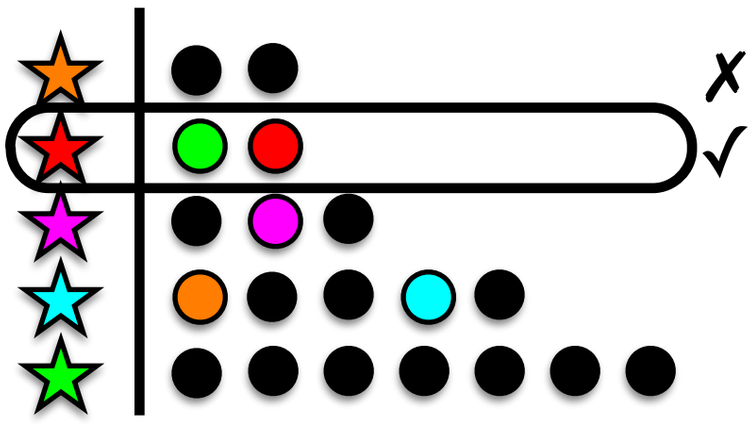
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

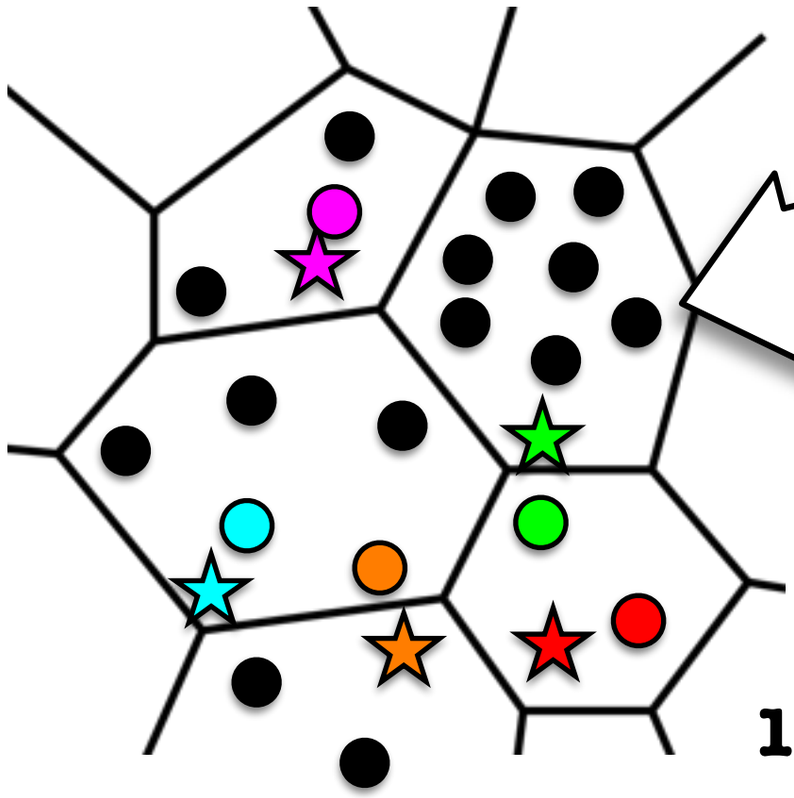
Linear search through words

[Sattler et al., ICCV'11] [code]

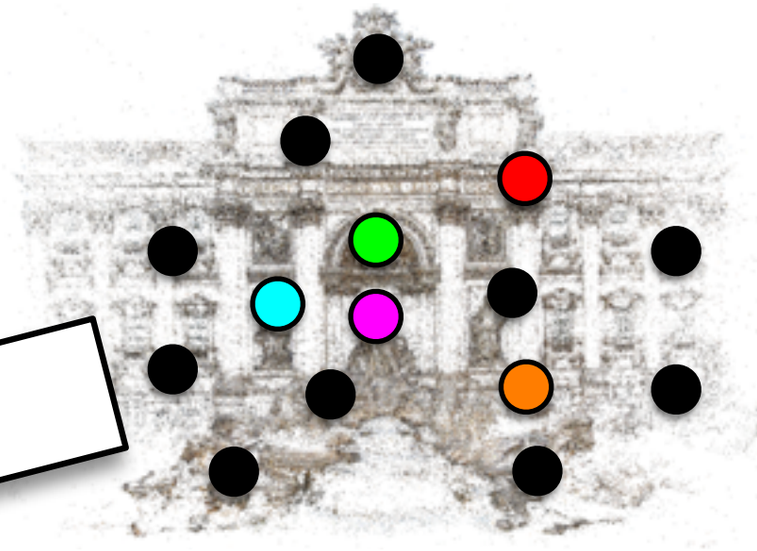
Vocabulary-Based Prioritized Search (VPS)



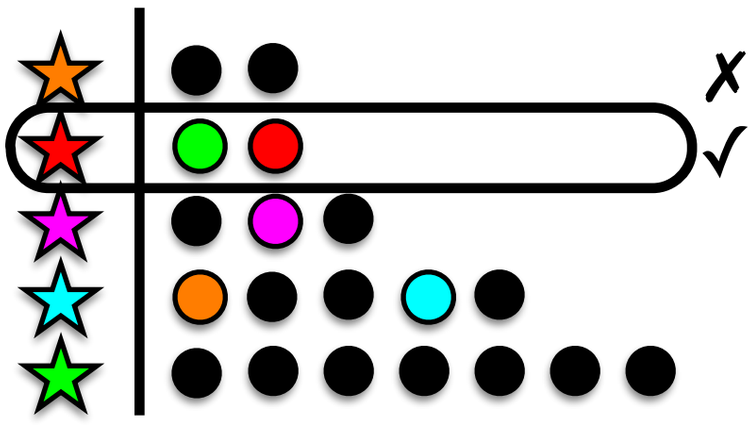
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

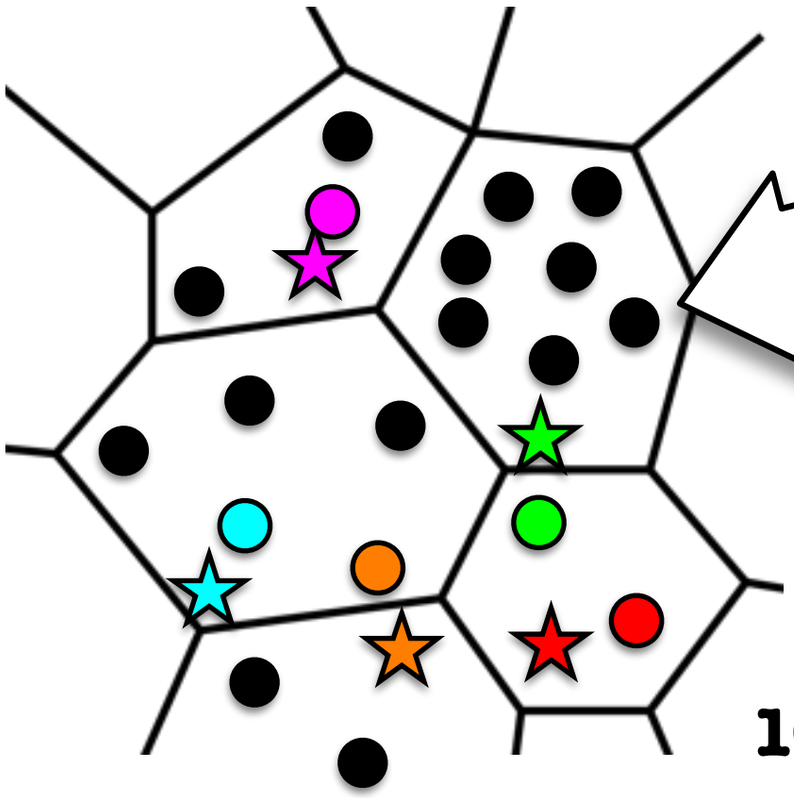
Linear search through words

[Sattler et al., ICCV'11] [code]

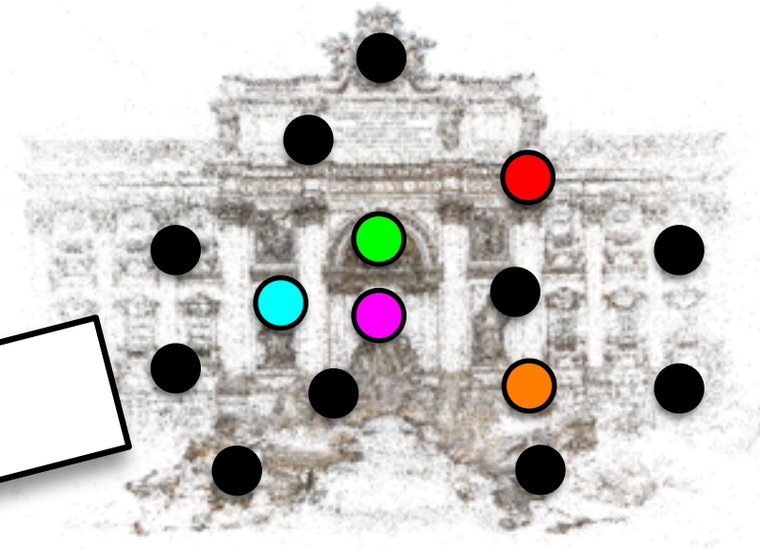
Vocabulary-Based Prioritized Search (VPS)



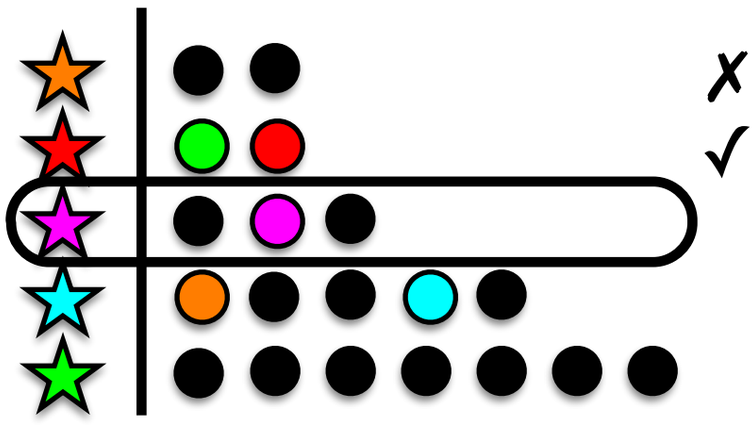
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

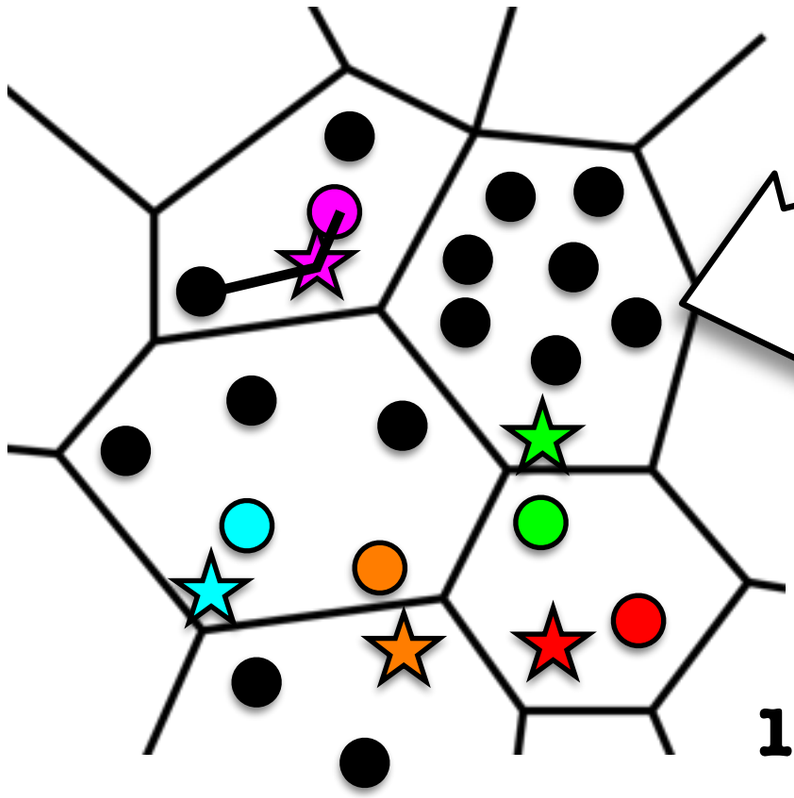
Linear search through words

[Sattler et al., ICCV'11] [code]

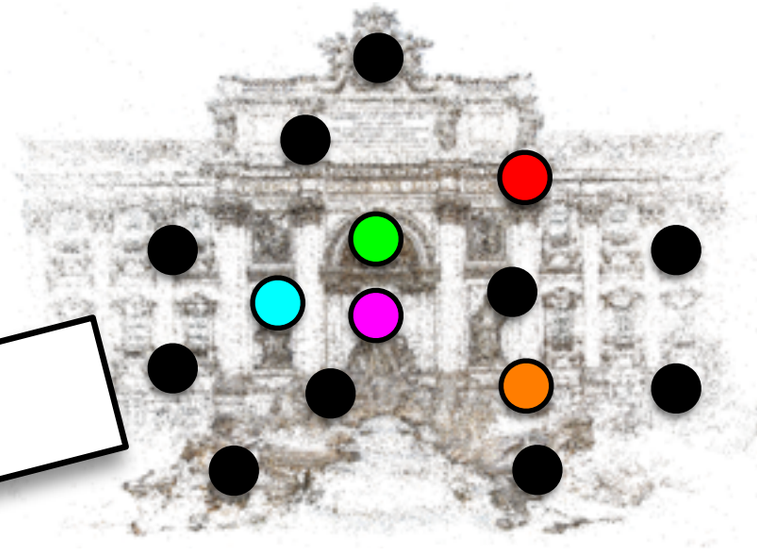
Vocabulary-Based Prioritized Search (VPS)



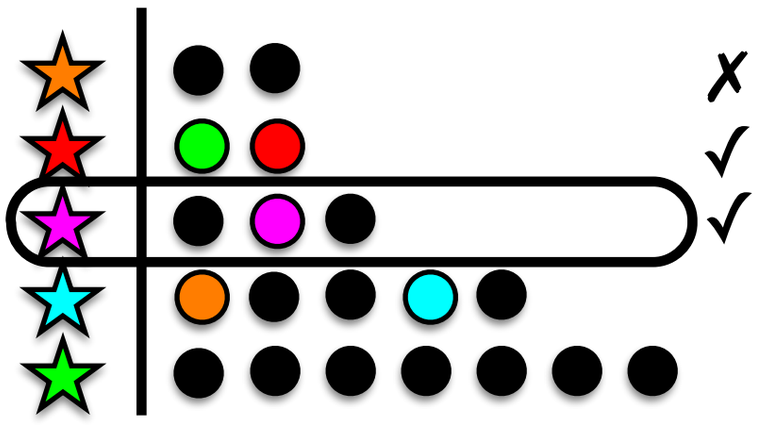
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

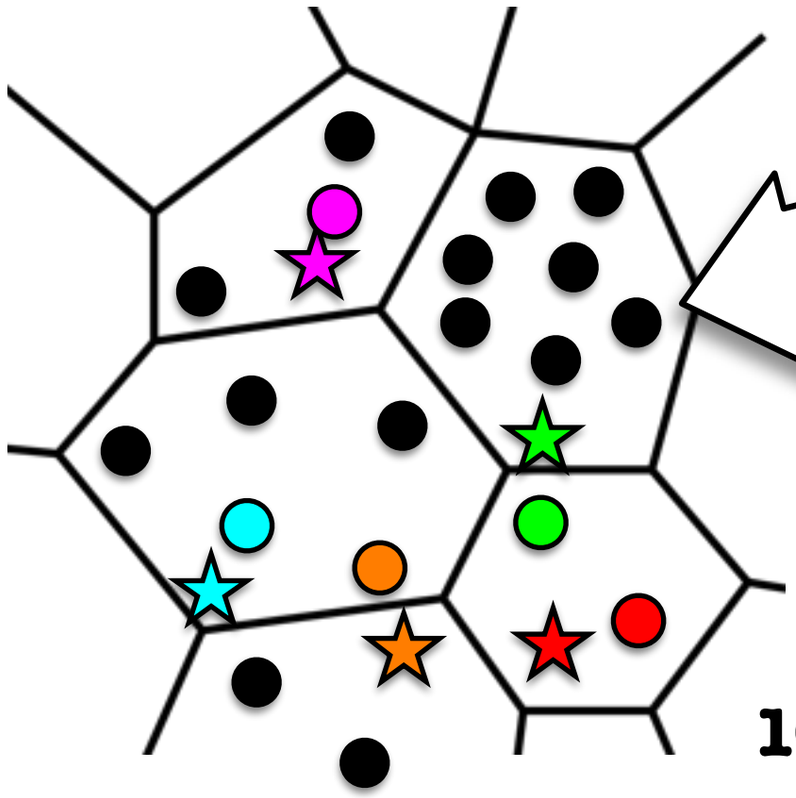
Linear search through words

[Sattler et al., ICCV'11] [code]

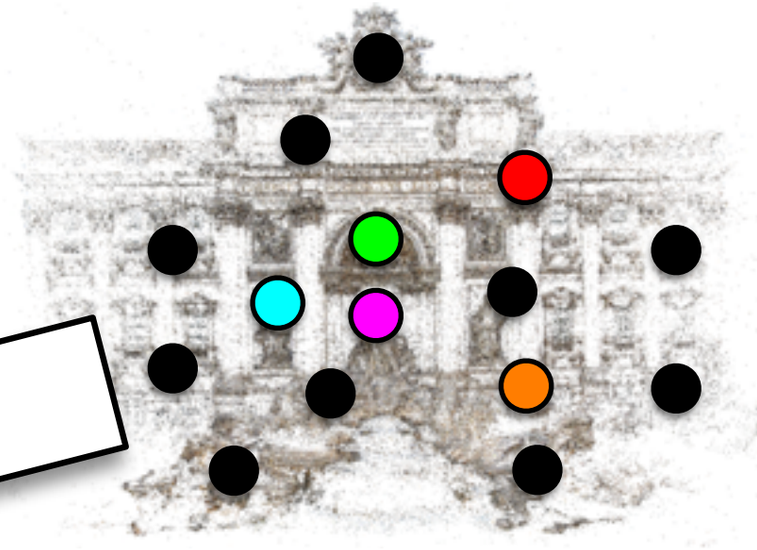
Vocabulary-Based Prioritized Search (VPS)



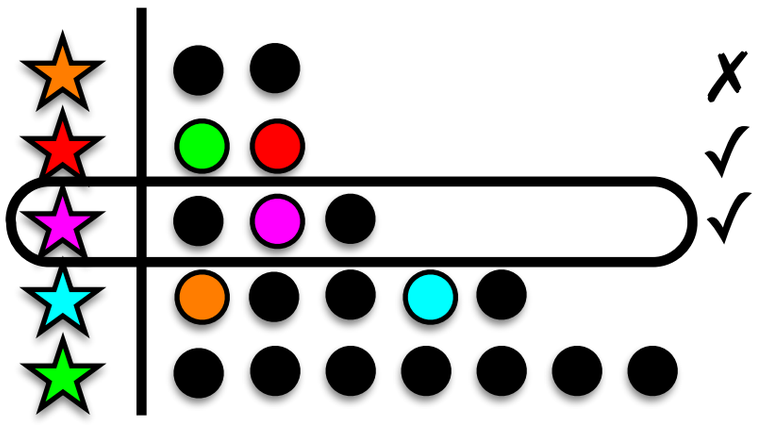
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

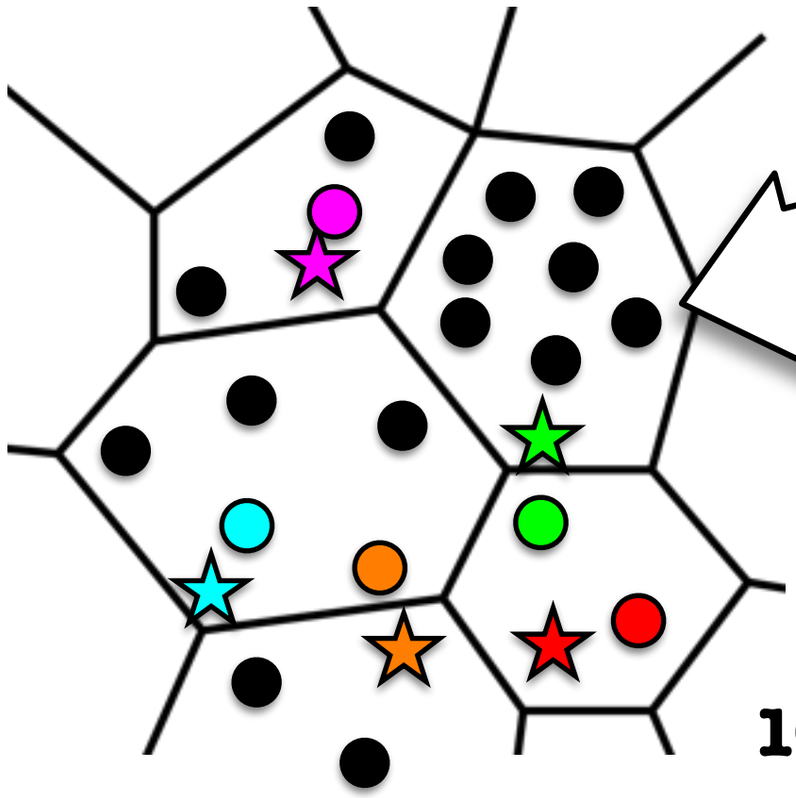
Linear search through words

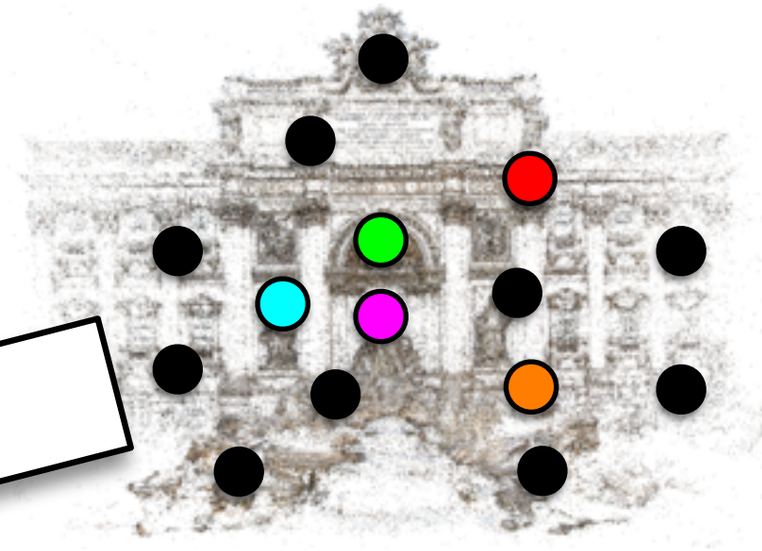
[Sattler et al., ICCV'11] [code]

Vocabulary-Based Prioritized Search (VPS)



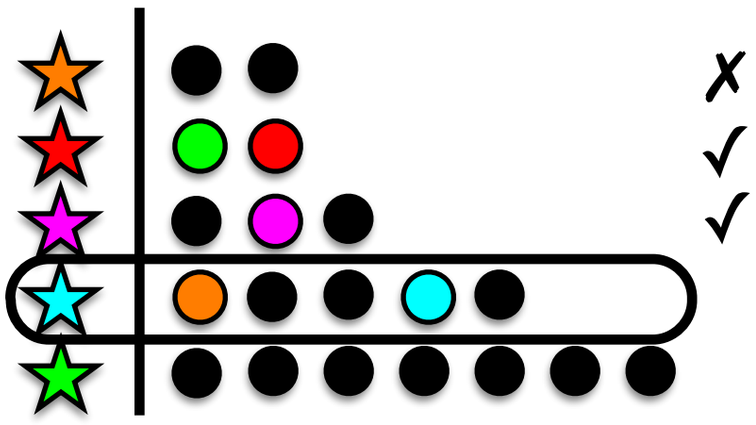
Query Image





3D Model

100k words



Assign features to words

Sort based on costs

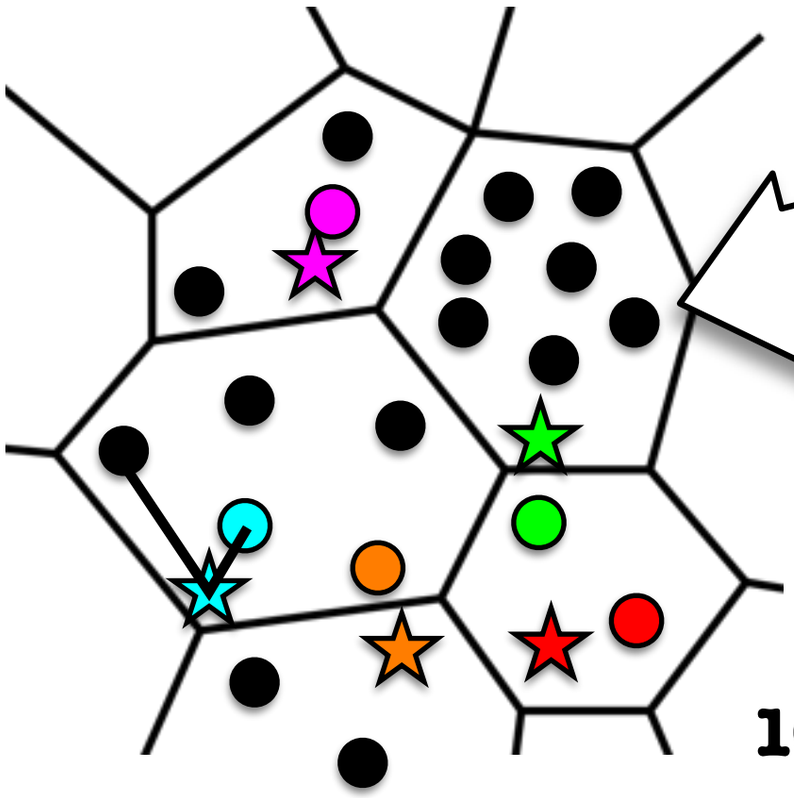
Linear search through words

[Sattler et al., ICCV'11] [code]

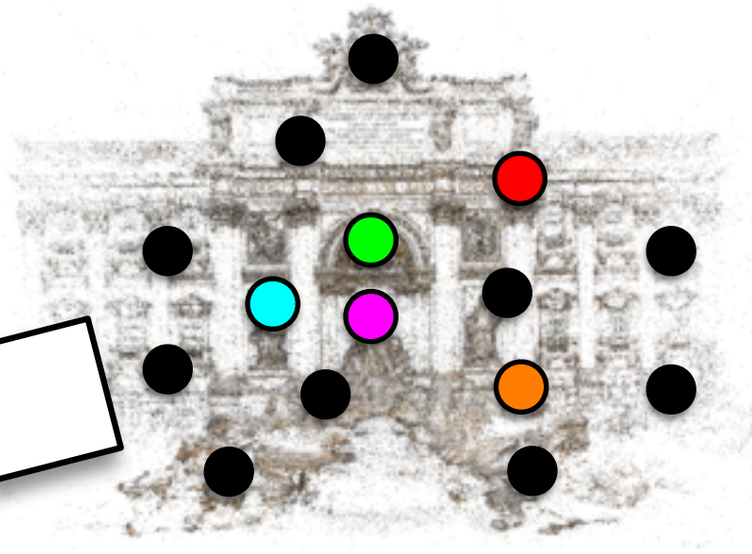
Vocabulary-Based Prioritized Search (VPS)



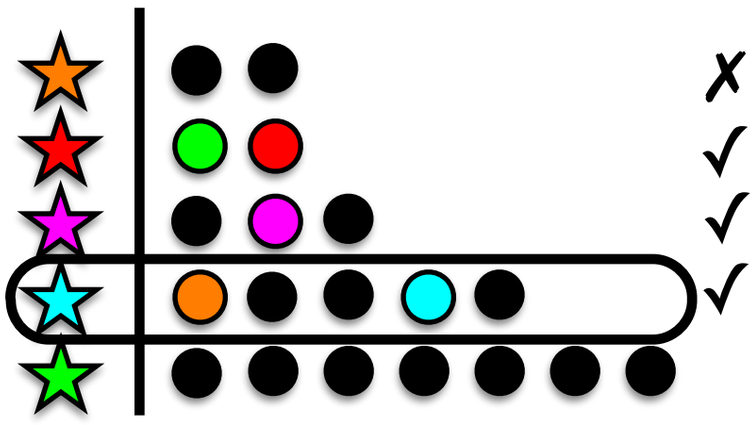
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

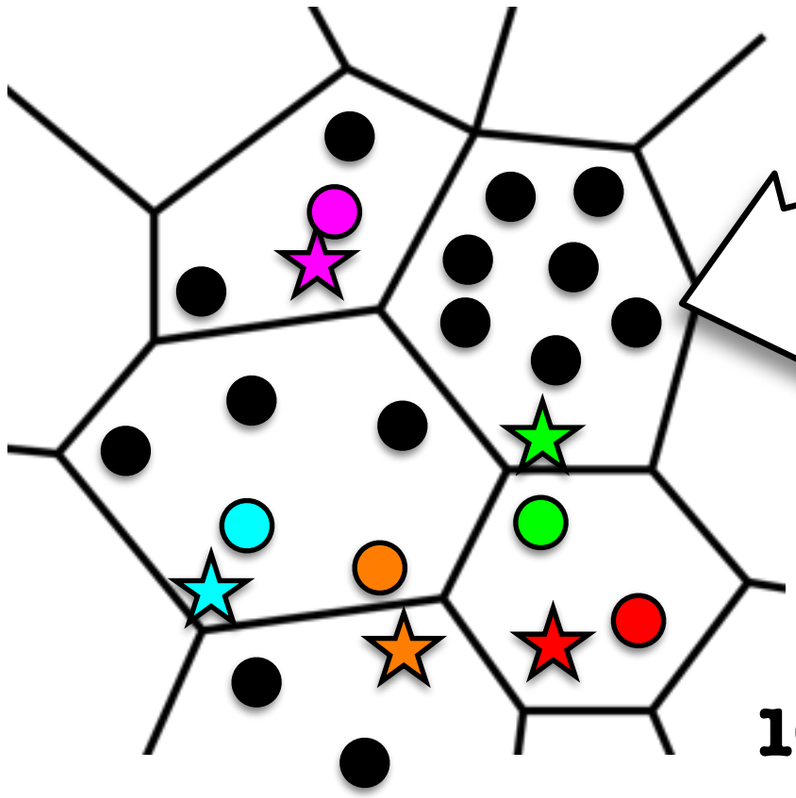
Linear search through words

[Sattler et al., ICCV'11] [code]

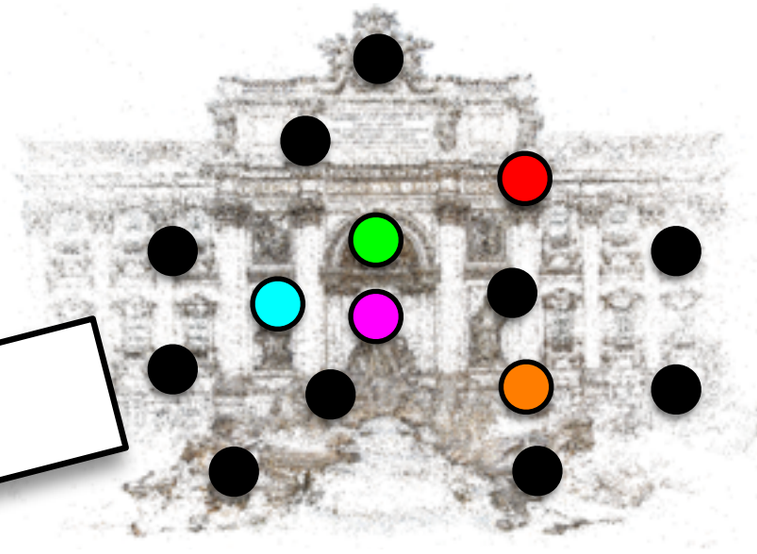
Vocabulary-Based Prioritized Search (VPS)



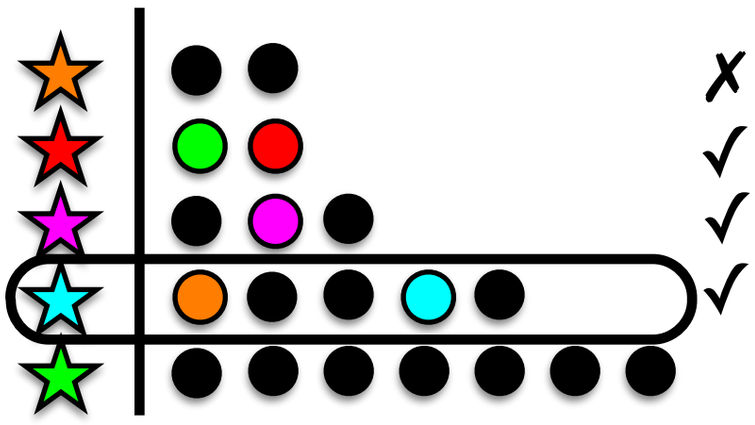
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

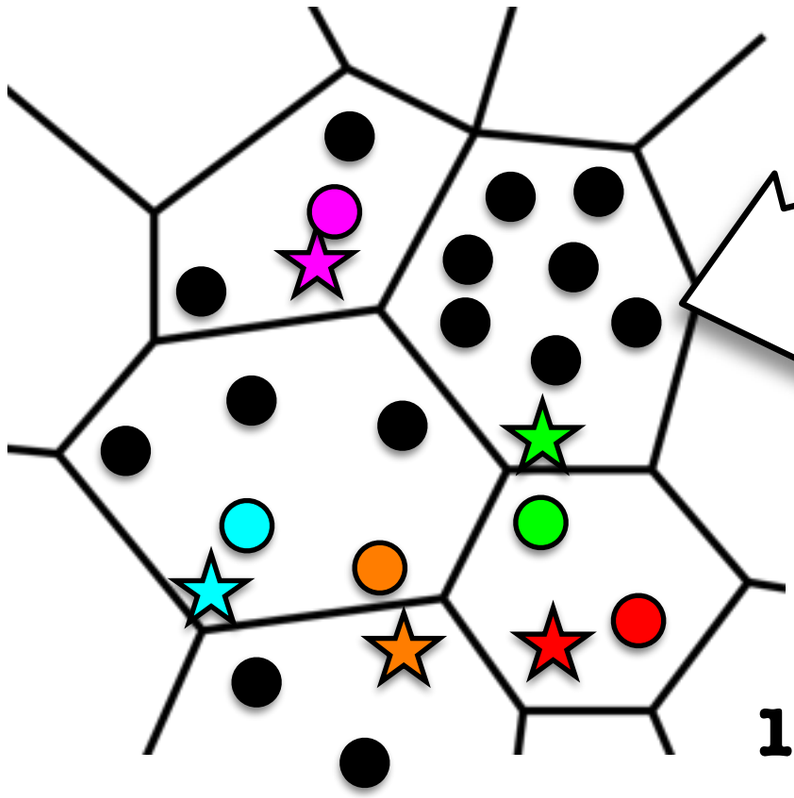
Linear search through words

[Sattler et al., ICCV'11] [code]

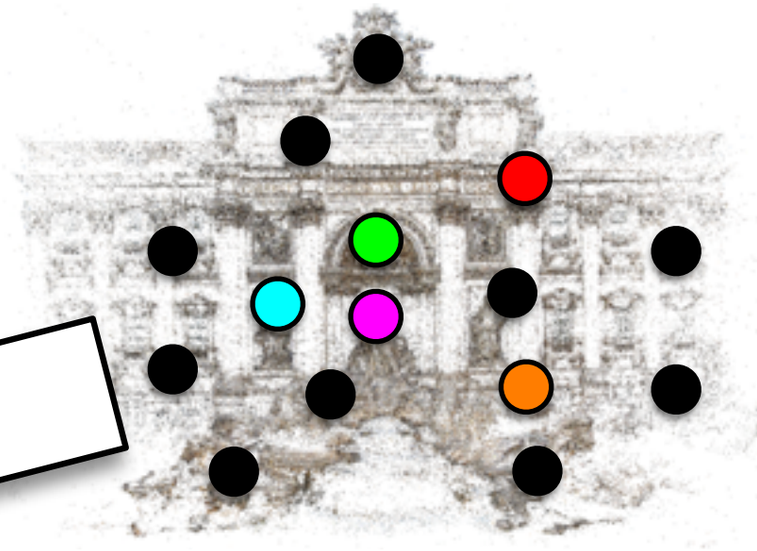
Vocabulary-Based Prioritized Search (VPS)



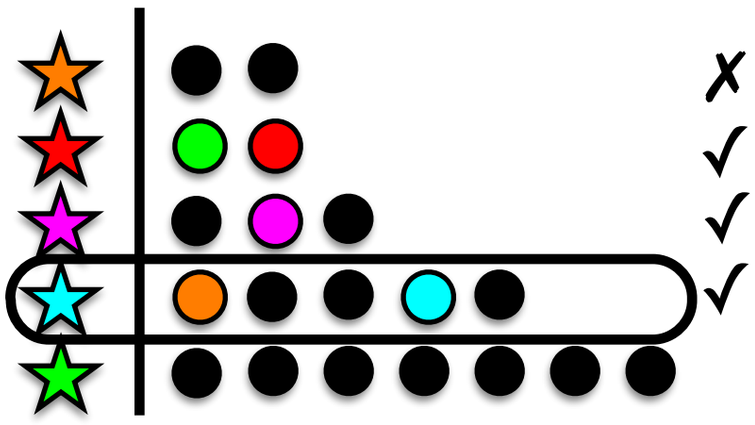
Query Image



100k words



3D Model



Assign features to words

Sort based on costs

Linear search through words

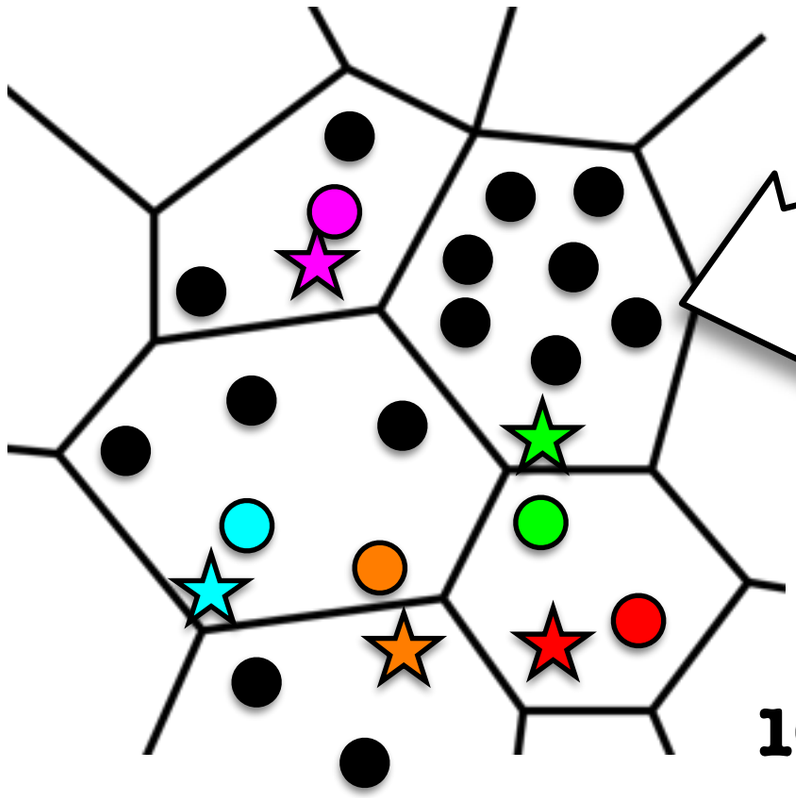
Stop after 100 **matches**

[Sattler et al., ICCV'11] [code]

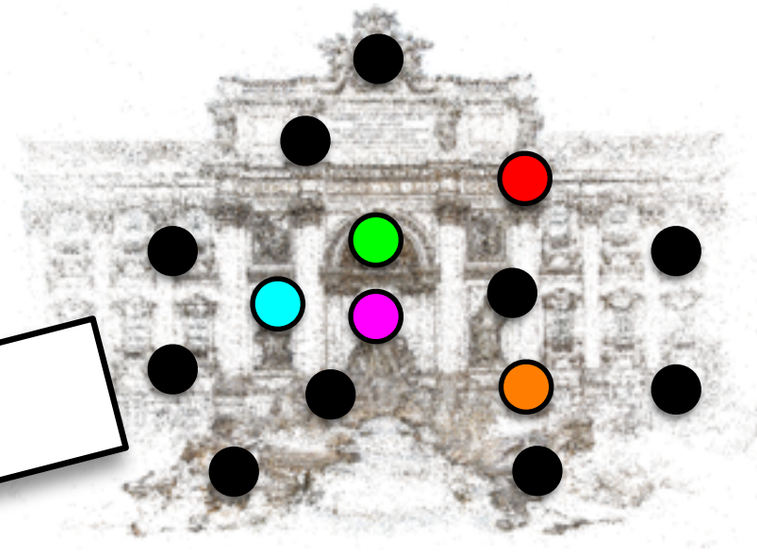
Vocabulary-Based Prioritized Search (VPS)



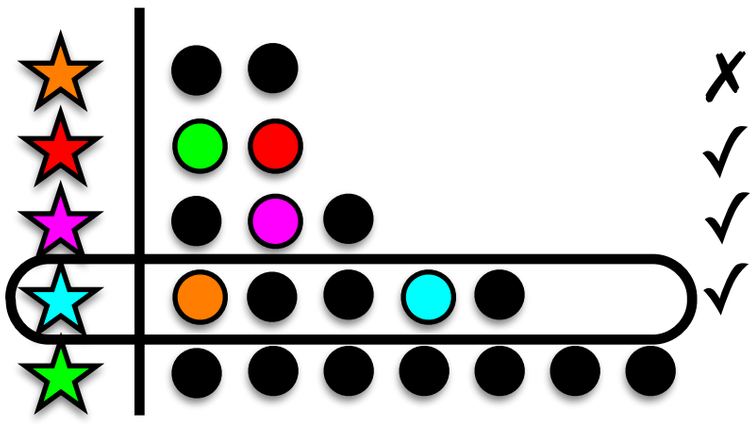
Query Image



100k words



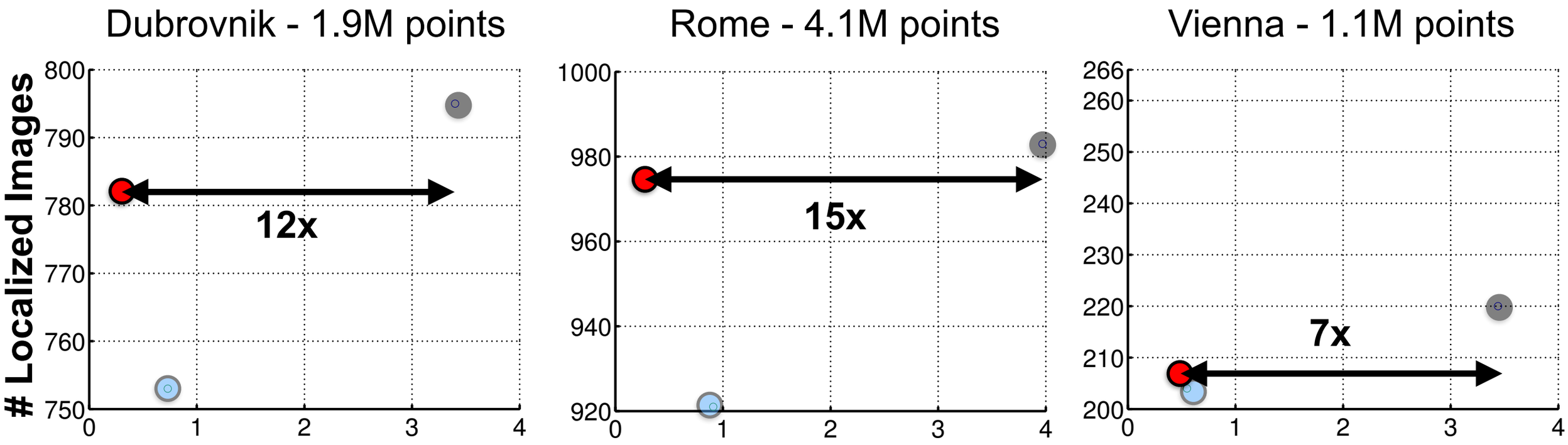
3D Model



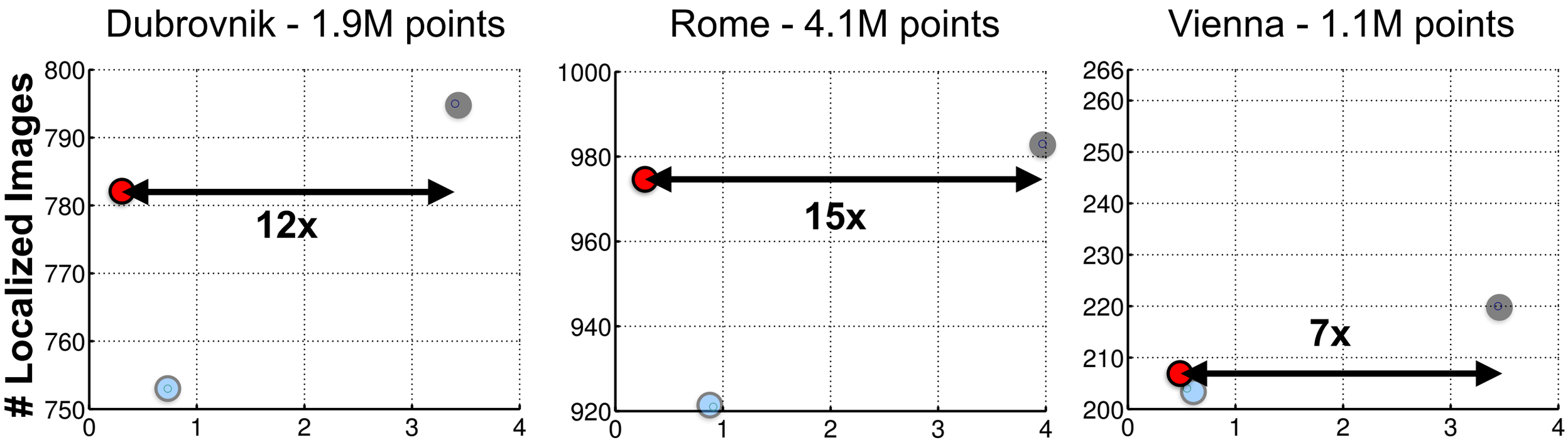
Assign features to words	**Sort based on costs**	**Linear search** through words	**Stop after** 100 **matches**	**Pose estimation:** RANSAC + p6p

[Sattler et al., ICCV'11] [code]

Results


Dubrovnik - 1.9M points
Rome - 4.1M points
Vienna - 1.1M points
Localized Images
800
790
780
770
760
750
1000
980
960
940
920
266
260
250
240
230
220
210
200
0
1
2
3
4
12x
15x
7x
Mean localization time per image [s] (excluding feature extraction)
kd-tree
P2F - compact model
VPS

Results

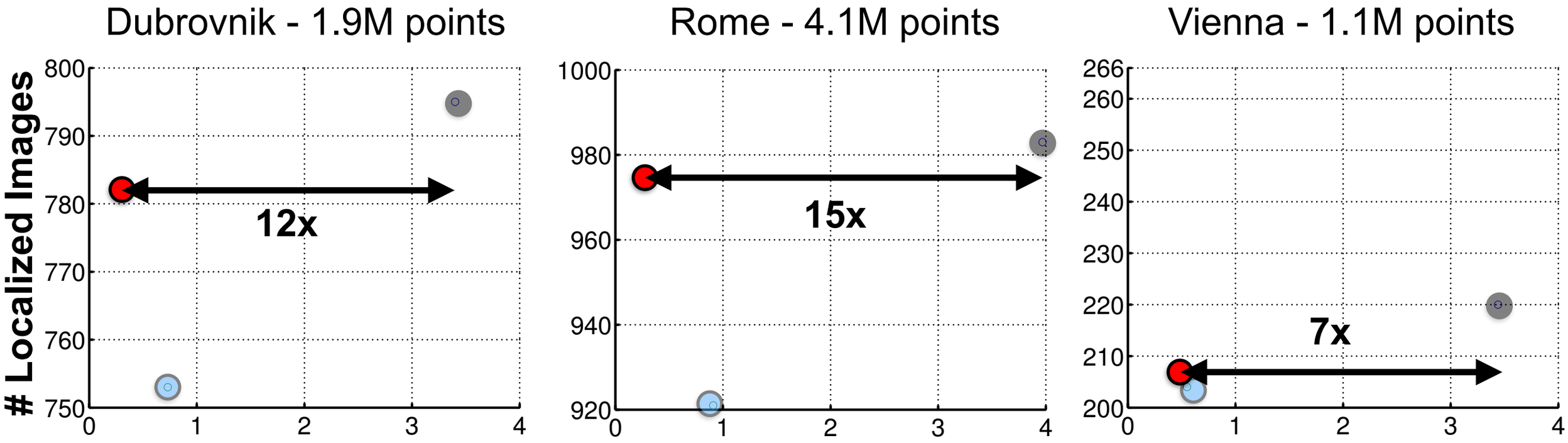


Mean localization time per image [s] (excluding feature extraction)

- kd-tree
- P2F - compact model
- **VPS**

✓ 10x faster than kd-tree, faster than P2F

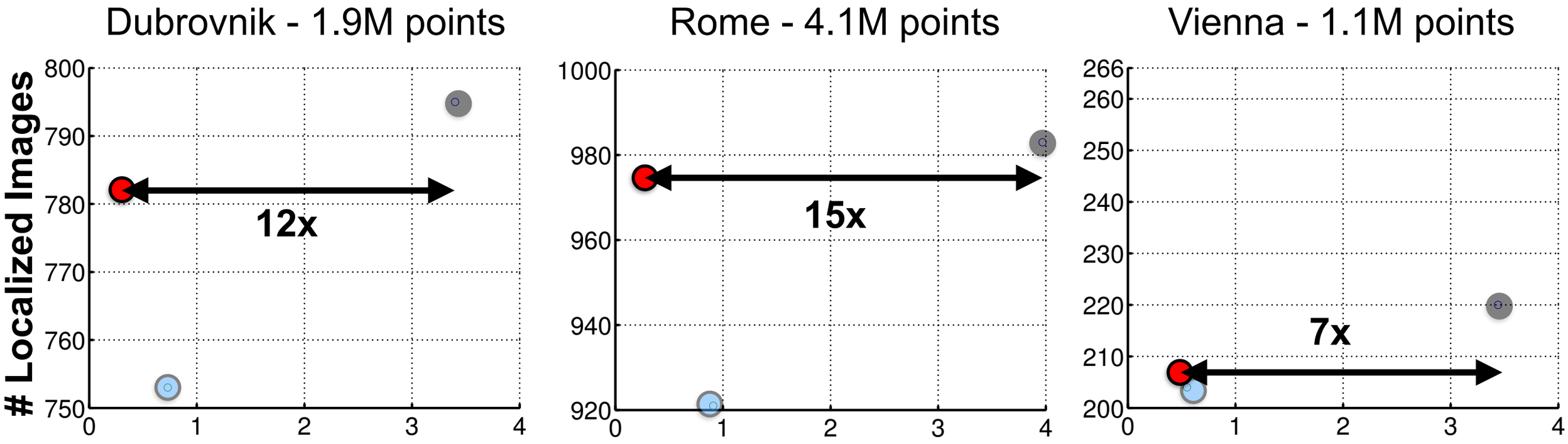
Results



✓ 10x faster than kd-tree, faster than P2F

✓ More effective than P2F …

Results



Mean localization time per image [s] (excluding feature extraction)

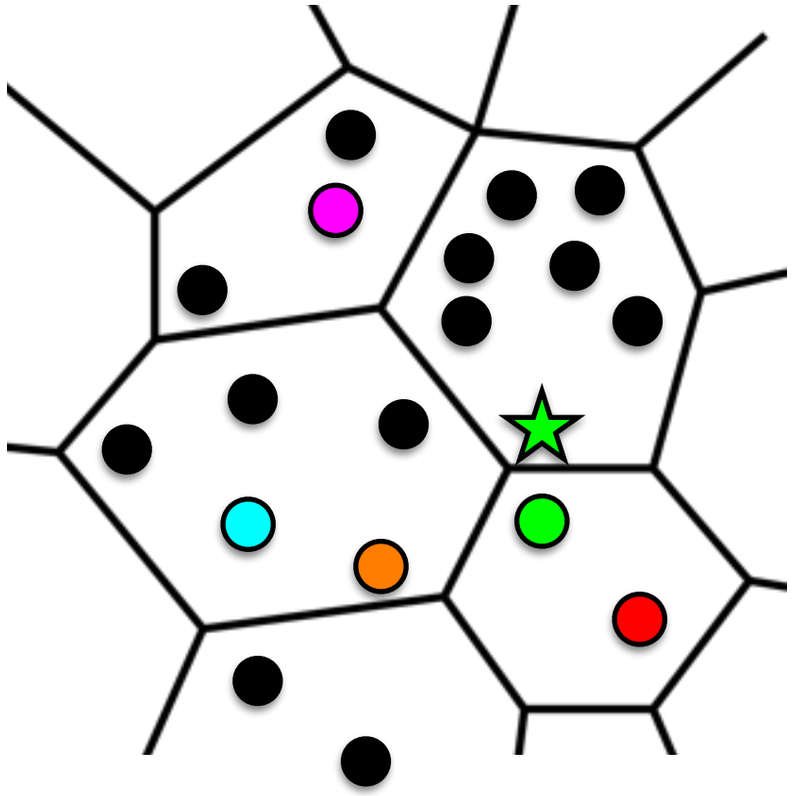
- kd-tree
- P2F - compact model
- **VPS**

✓ 10x faster than kd-tree, faster than P2F

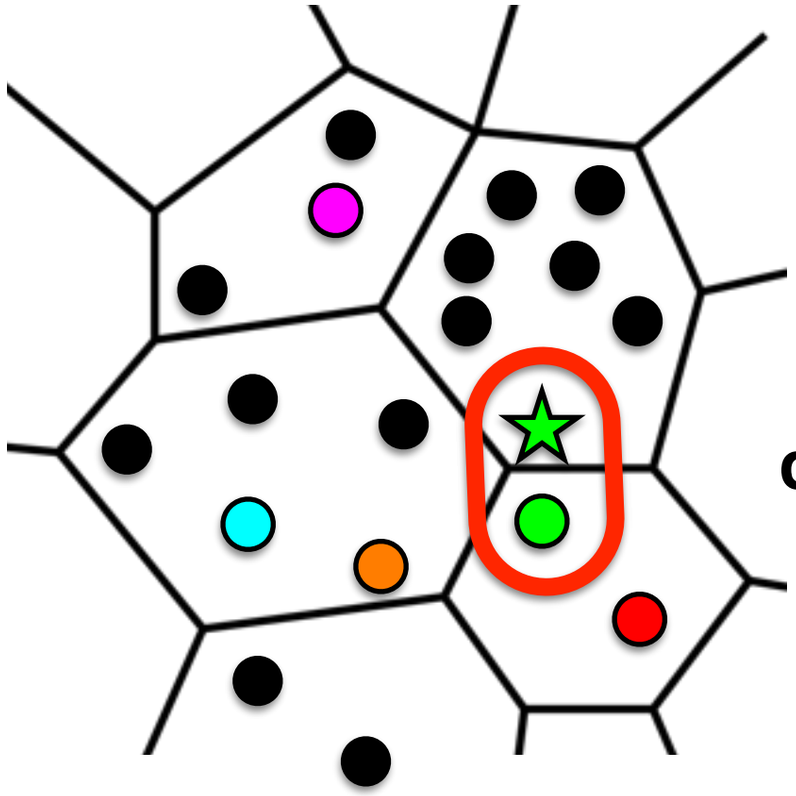
✓ More effective than P2F …

✗ … but still less effective than kd-tree

Quantization Artifacts

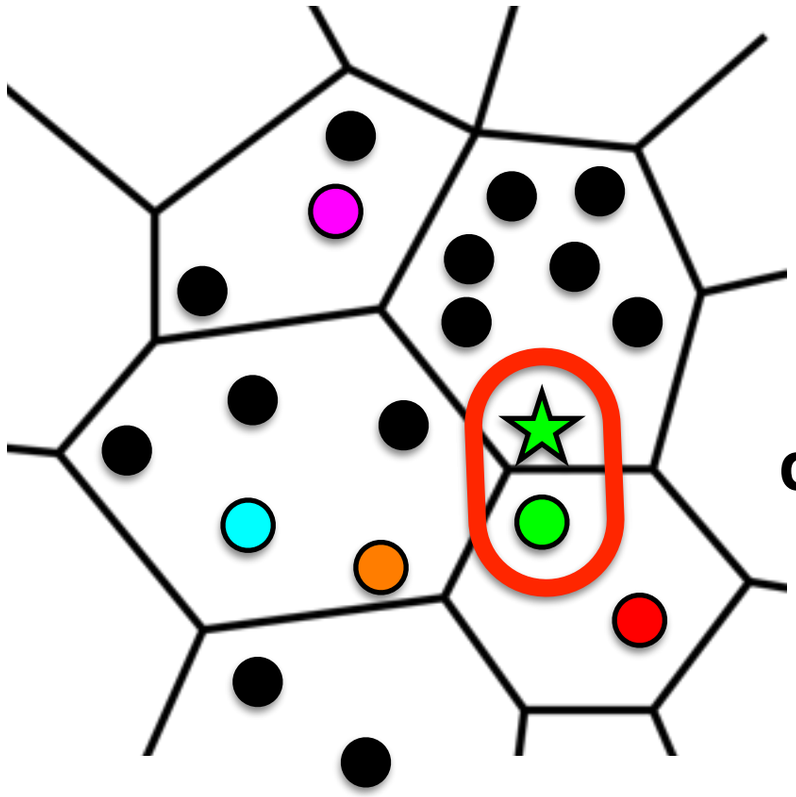


Quantization Artifacts



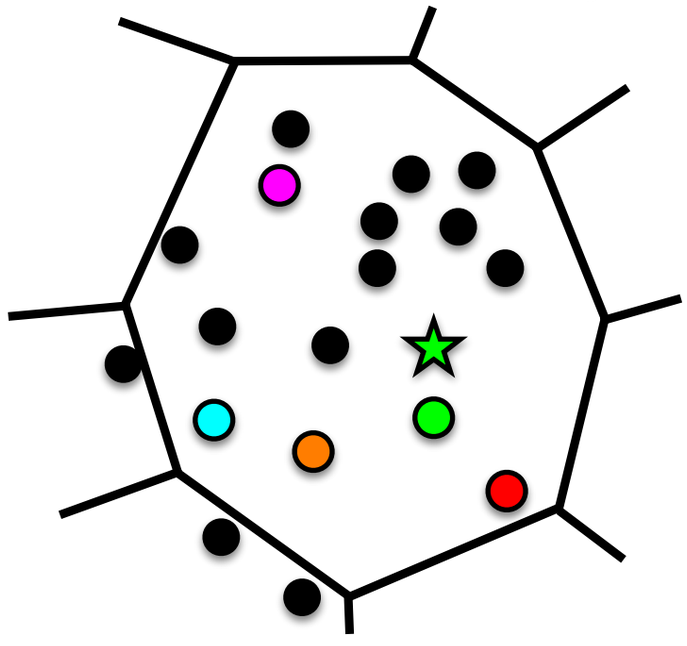
Match missed due to quantization!

Quantization Artifacts

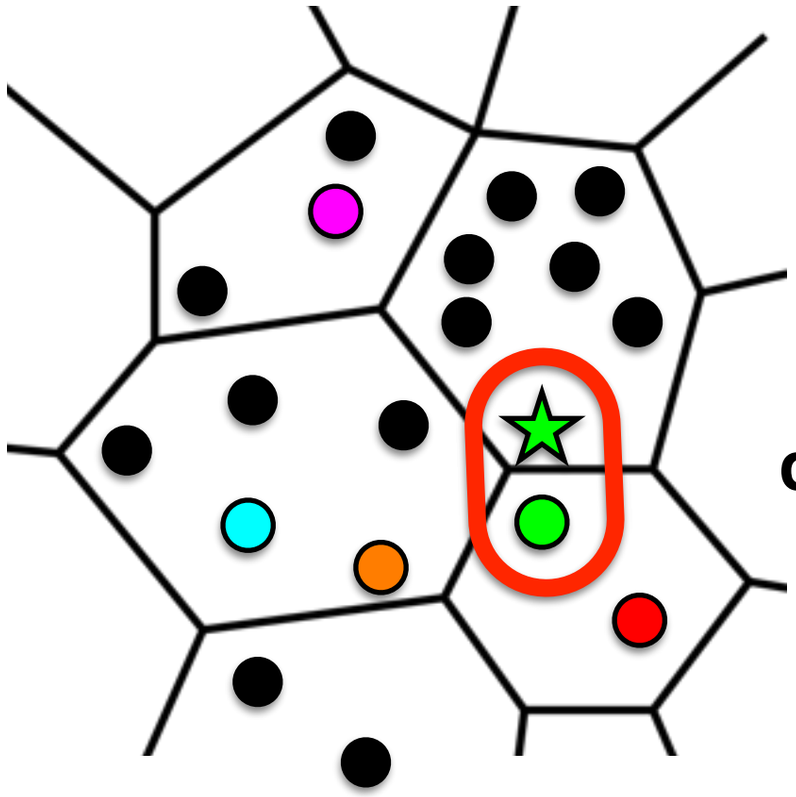


Match missed due to quantization!

Smaller Vocabulary

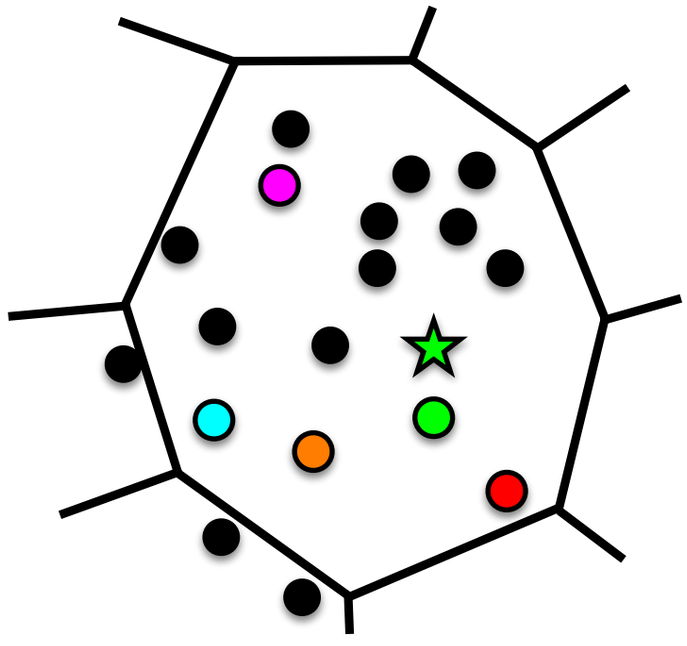


Quantization Artifacts

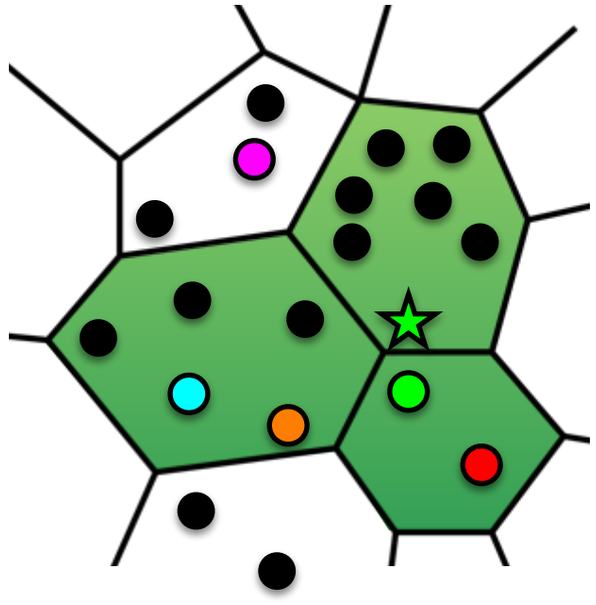


Match missed due to quantization!

Smaller Vocabulary

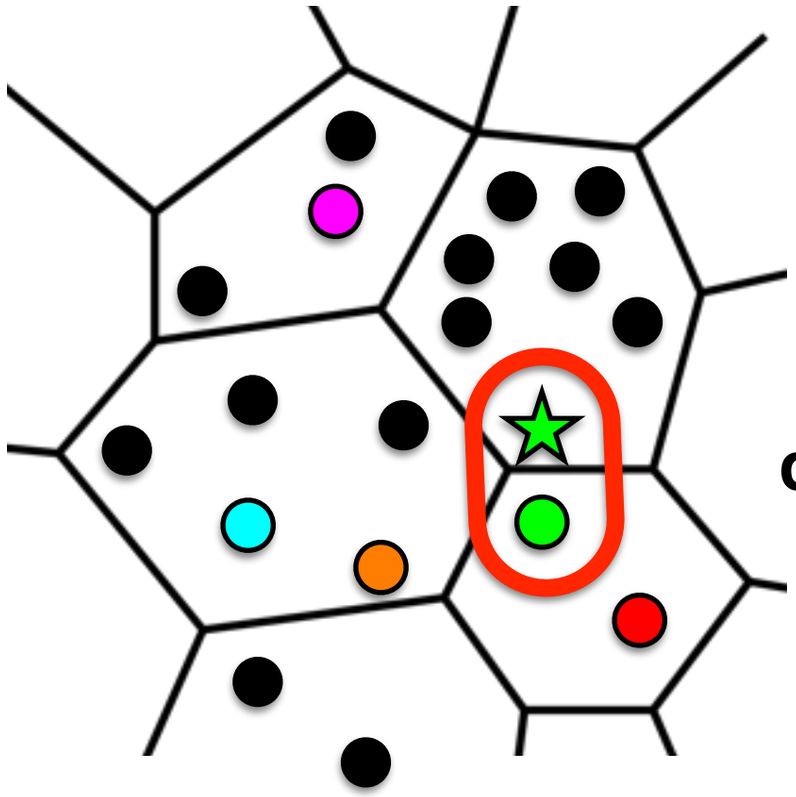


Soft Assignments



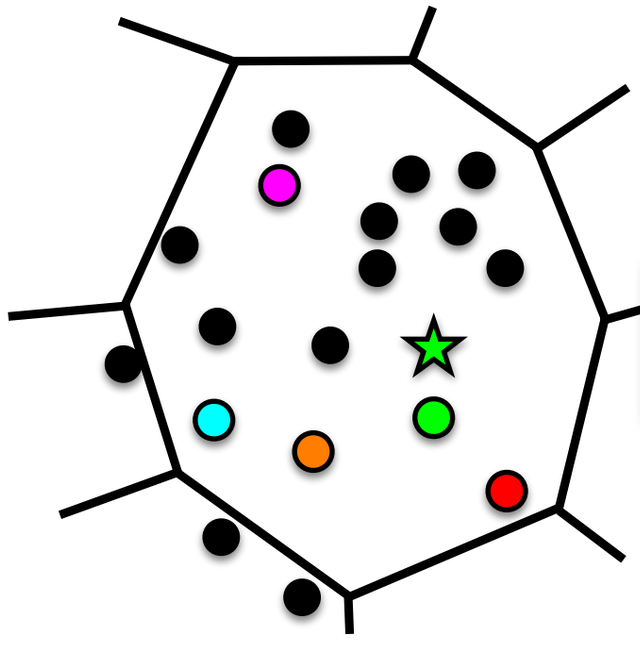
[Philbin et al., CPVR'08]

Quantization Artifacts



Match missed due to quantization!

Smaller Vocabulary



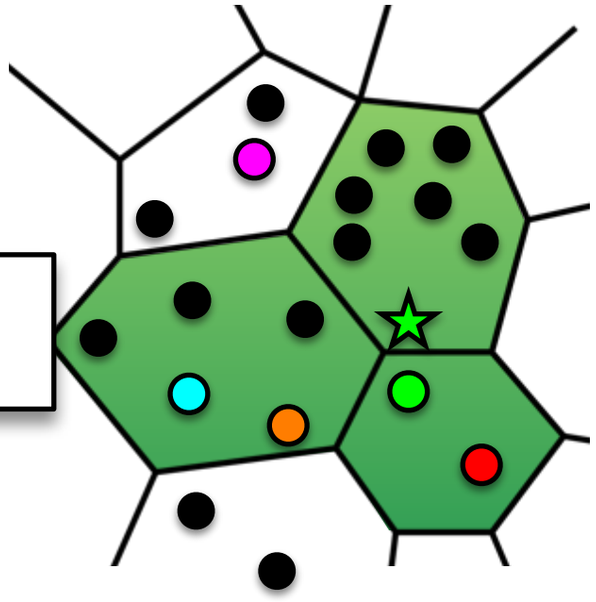
Additional costs

for each feature

$$\mathcal{O}(P/W)$$

W words, P points

Soft Assignments



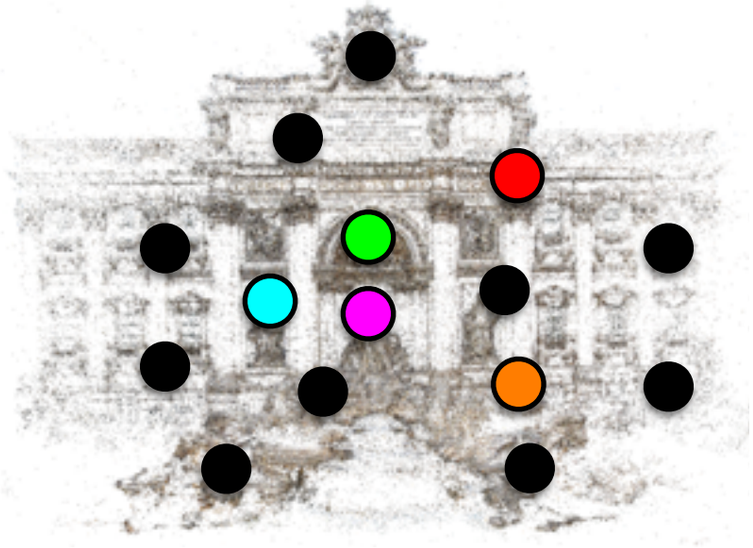
[Philbin et al., CPVR'08]

Handling Quantization Artifacts

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



3D Model

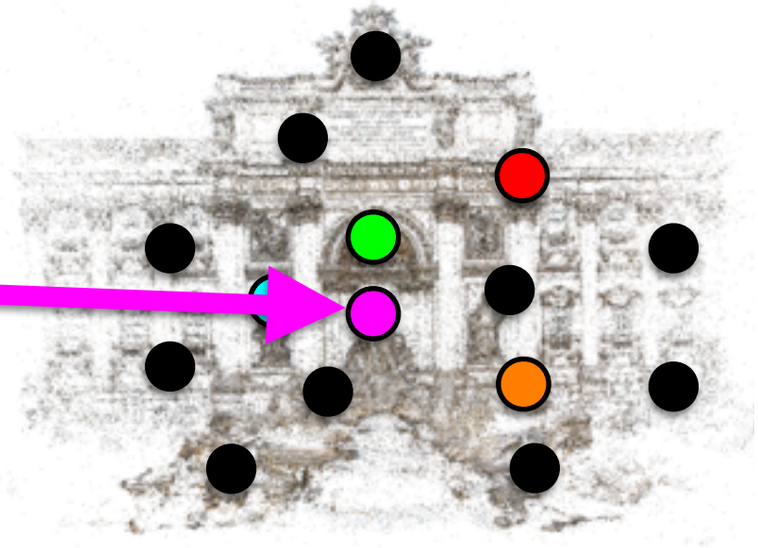
- Points surrounding 2D-to-3D match should also be visible

Handling Quantization Artifacts

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



3D Model

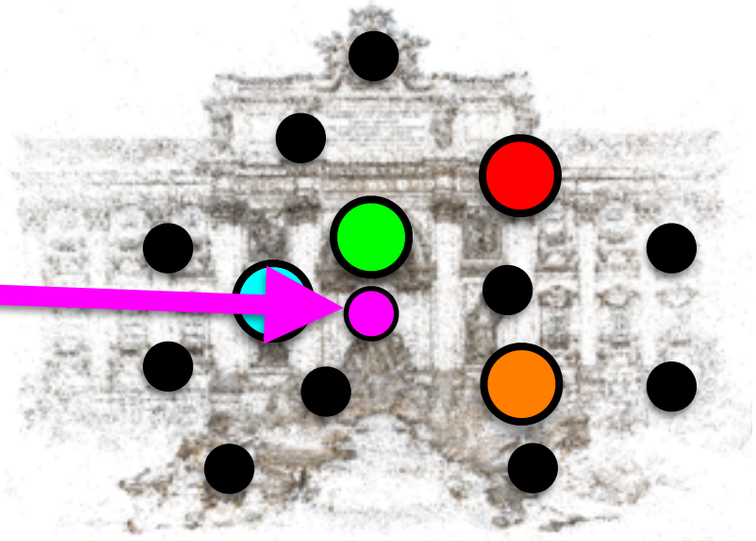
- Points surrounding 2D-to-3D match should also be visible

Handling Quantization Artifacts

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



3D Model

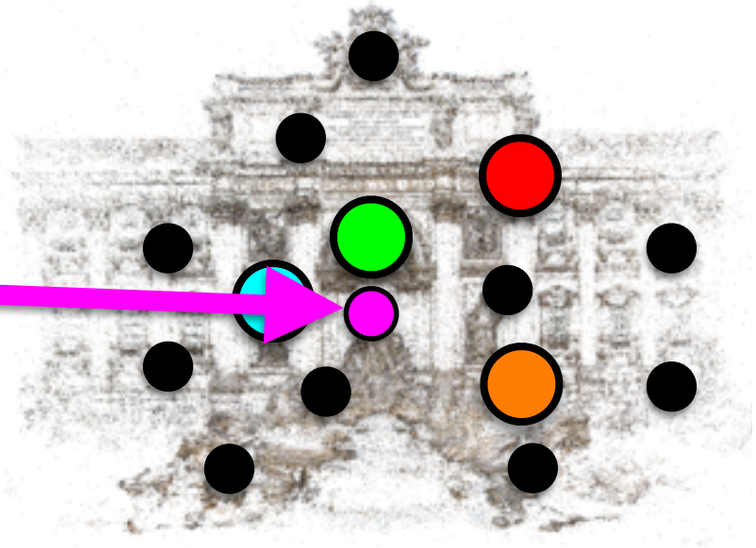
- Points surrounding 2D-to-3D match should also be visible

Handling Quantization Artifacts

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image

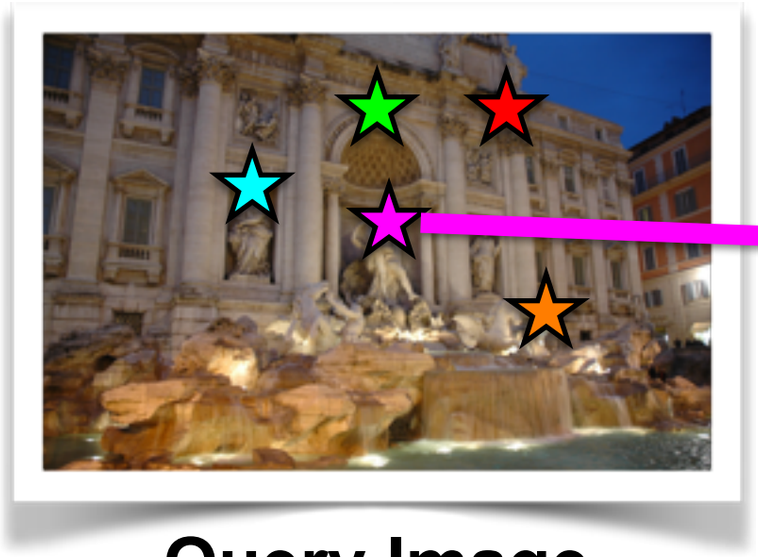


3D Model

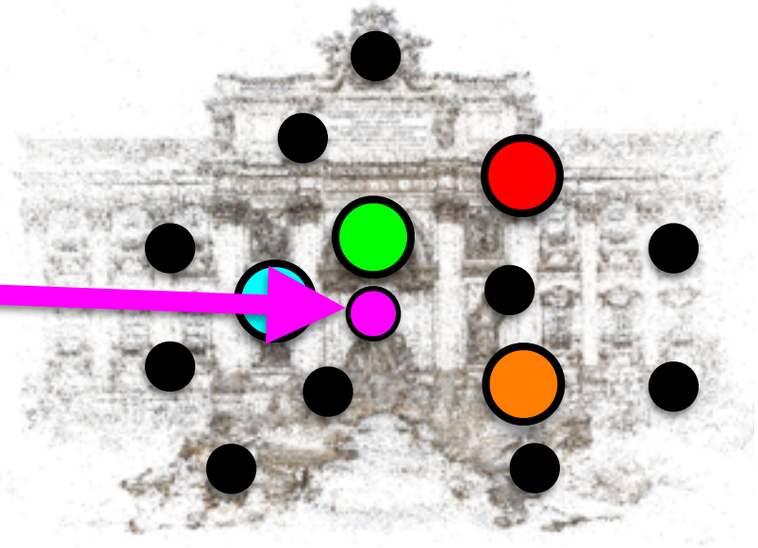
- Points surrounding 2D-to-3D match should also be visible
  - [Choudhary & Narayanan, ECCV'12] : 2D-to-3D matches seed 3D-to-2D search

Handling Quantization Artifacts

Idea: Exploit **co-occurrence of matches** to recover matches



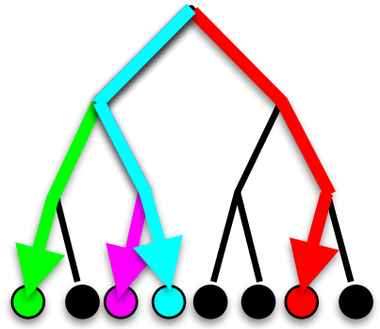
Query Image



3D Model

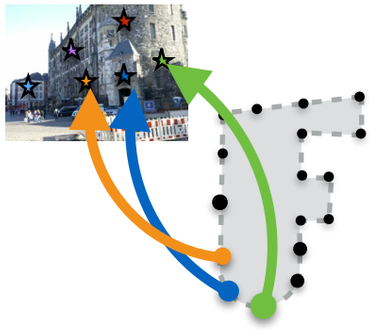
- Points surrounding 2D-to-3D match should also be visible
  - [Choudhary & Narayanan, ECCV'12] : 2D-to-3D matches seed 3D-to-2D search
  - [Sattler et al., ECCV'12] : Integrate 3D-to-2D matching into VPS

Localization - Overview



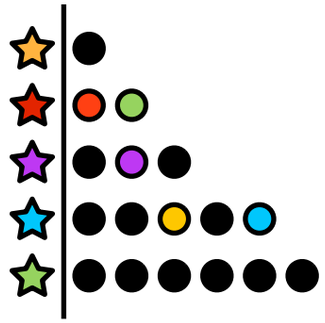
Baseline: kd-tree search

[Sattler et al., ICCV'11]



P2F

[Li et al., ECCV'10]



VPS

[Sattler et al., ICCV'11]



Active Search

[Sattler et al., ECCV'12]

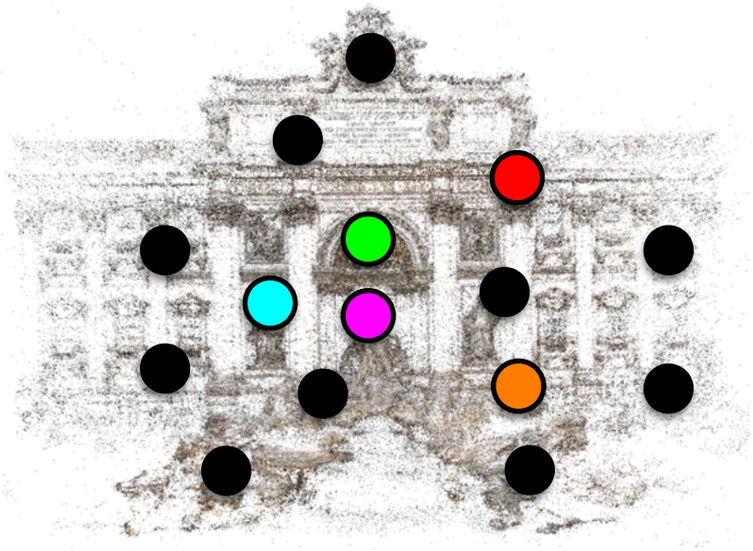
effectivness	✓✓	✗✗	✗	✓✓✓
efficiency	✗✗	✓	✓	✓✓

Active Search

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



3D Model

- Points surrounding 2D-to-3D match should also be visible:

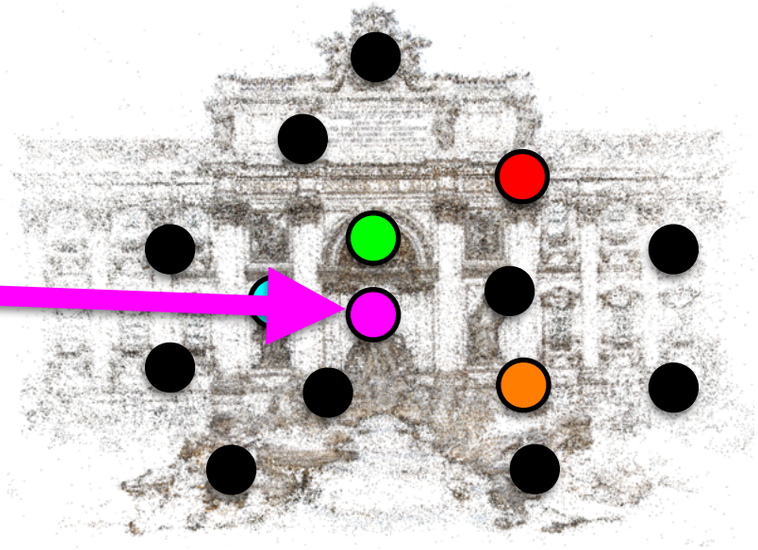
[Sattler et al., ECCV'12] [code]

Active Search

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



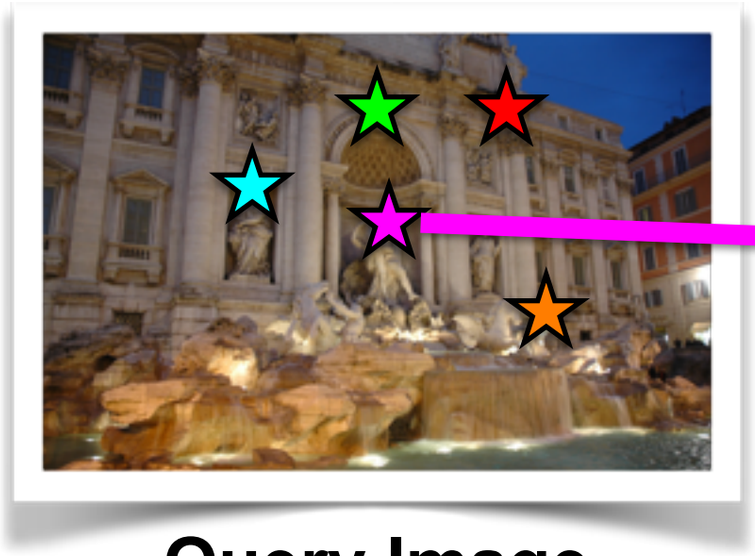
3D Model

- Points surrounding 2D-to-3D match should also be visible:

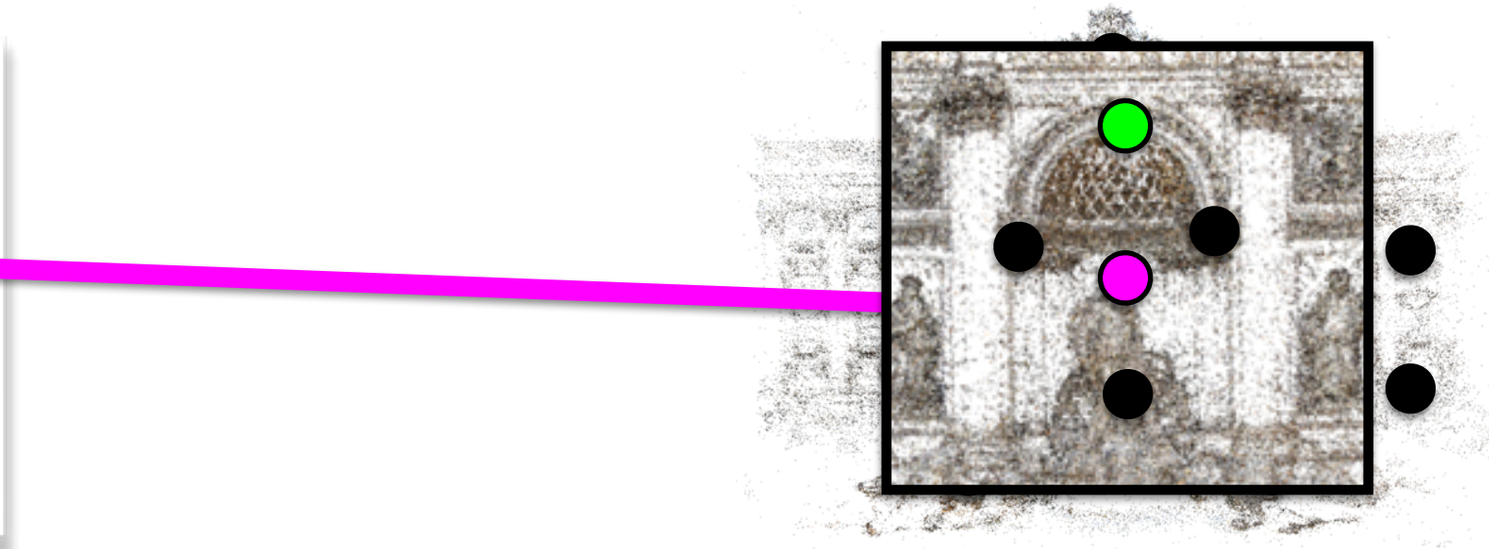
[Sattler et al., ECCV'12] [code]

Active Search

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



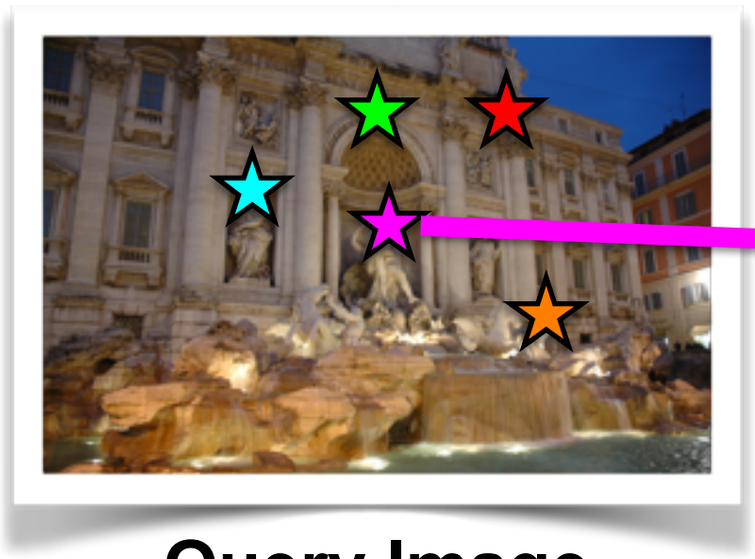
3D Model

- Points surrounding 2D-to-3D match should also be visible:

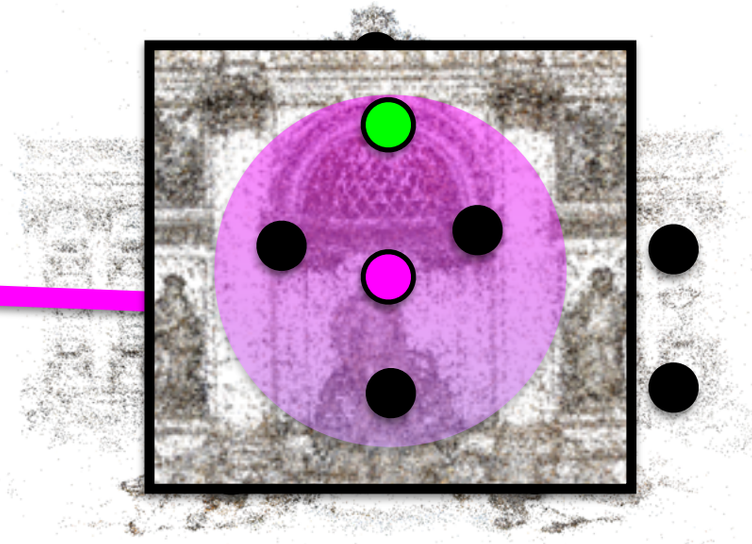
[Sattler et al., ECCV'12] [code]

Active Search

Idea: Exploit **co-occurrence of matches** to recover matches



Query Image



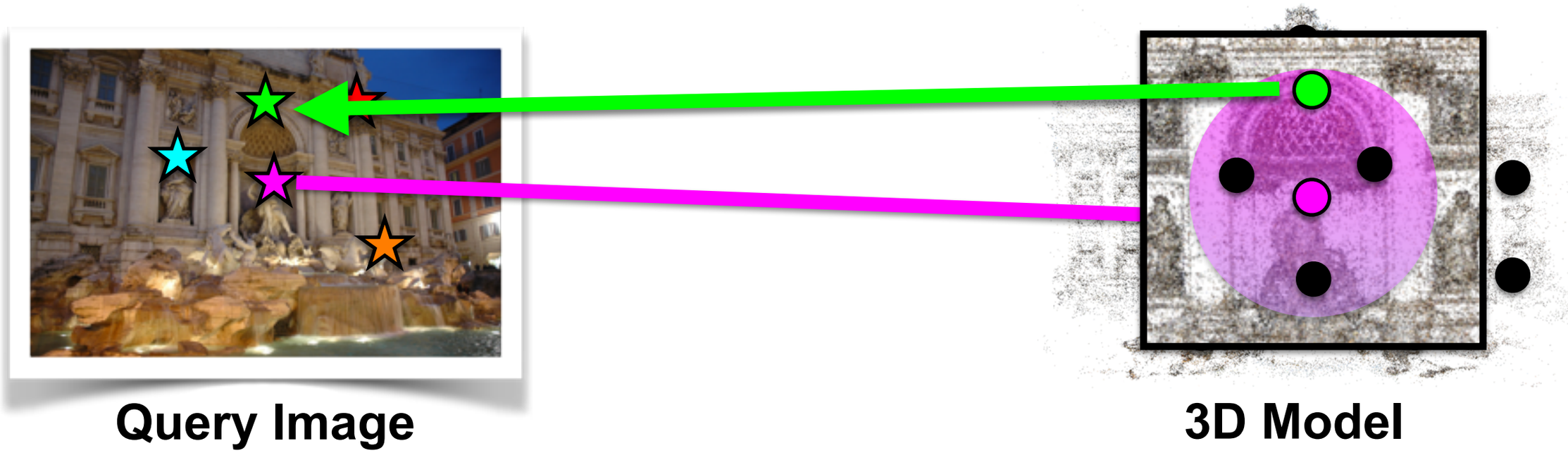
3D Model

- Points surrounding 2D-to-3D match should also be visible:
  - Find nearest neighbors in 3D around matching point

[Sattler et al., ECCV'12] [code]

Active Search

Idea: Exploit **co-occurrence of matches** to recover matches



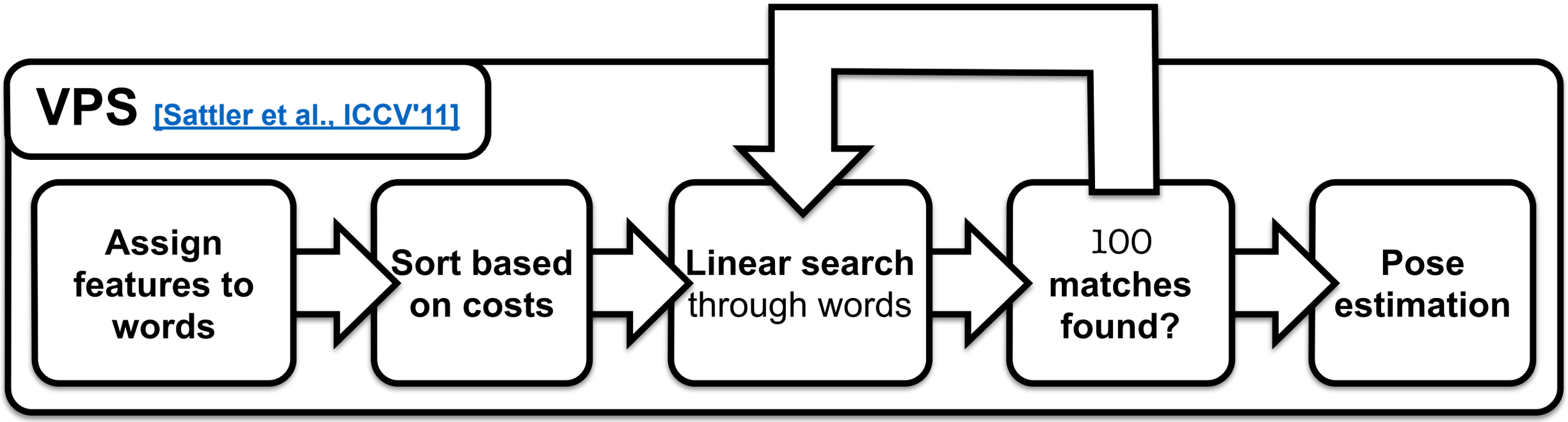
Query Image

3D Model

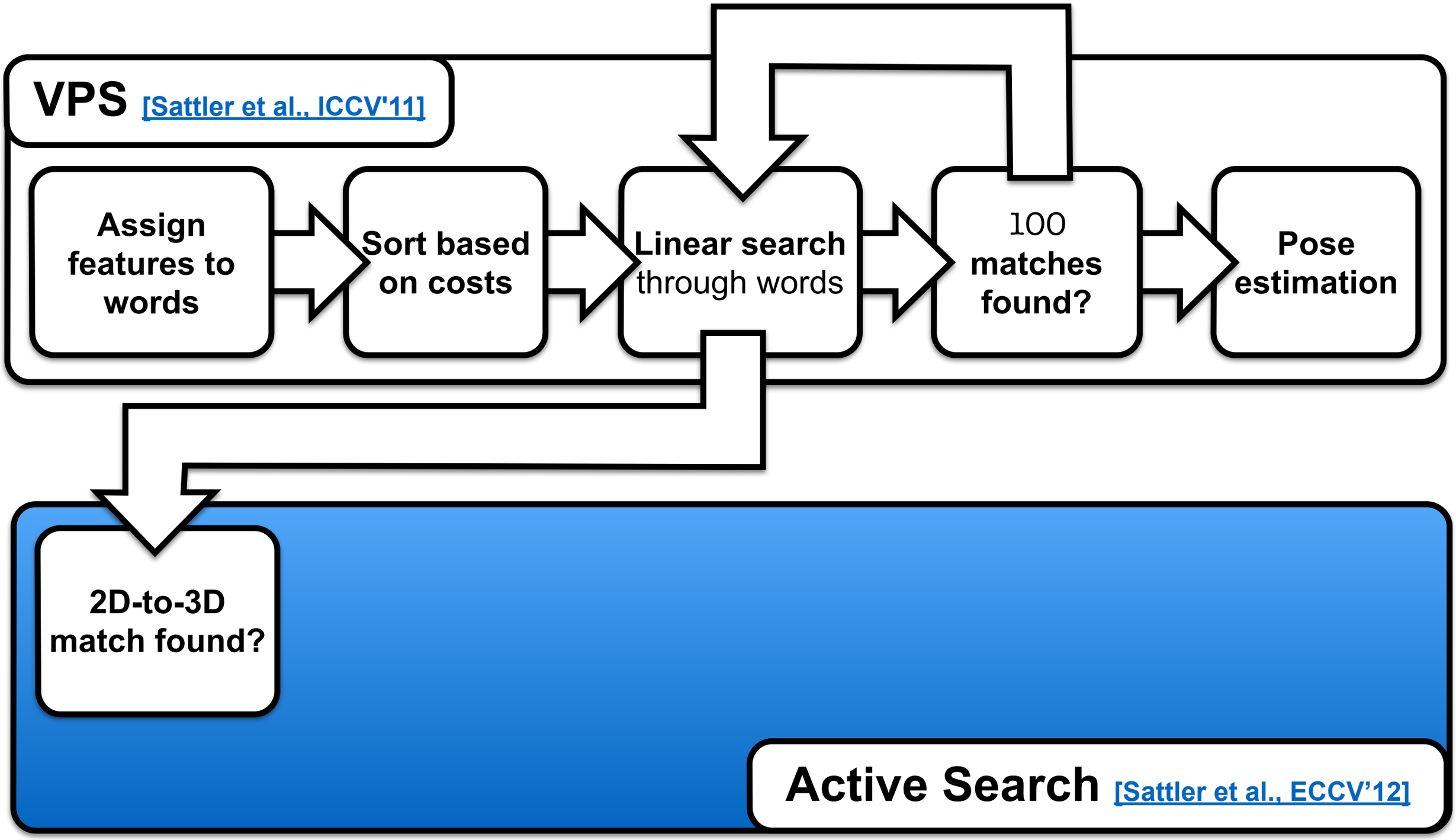
- Points surrounding 2D-to-3D match should also be visible:
  - Find nearest neighbors in 3D around matching point
  - Perform 3D-to-2D search for neighbors

[Sattler et al., ECCV'12] [code]

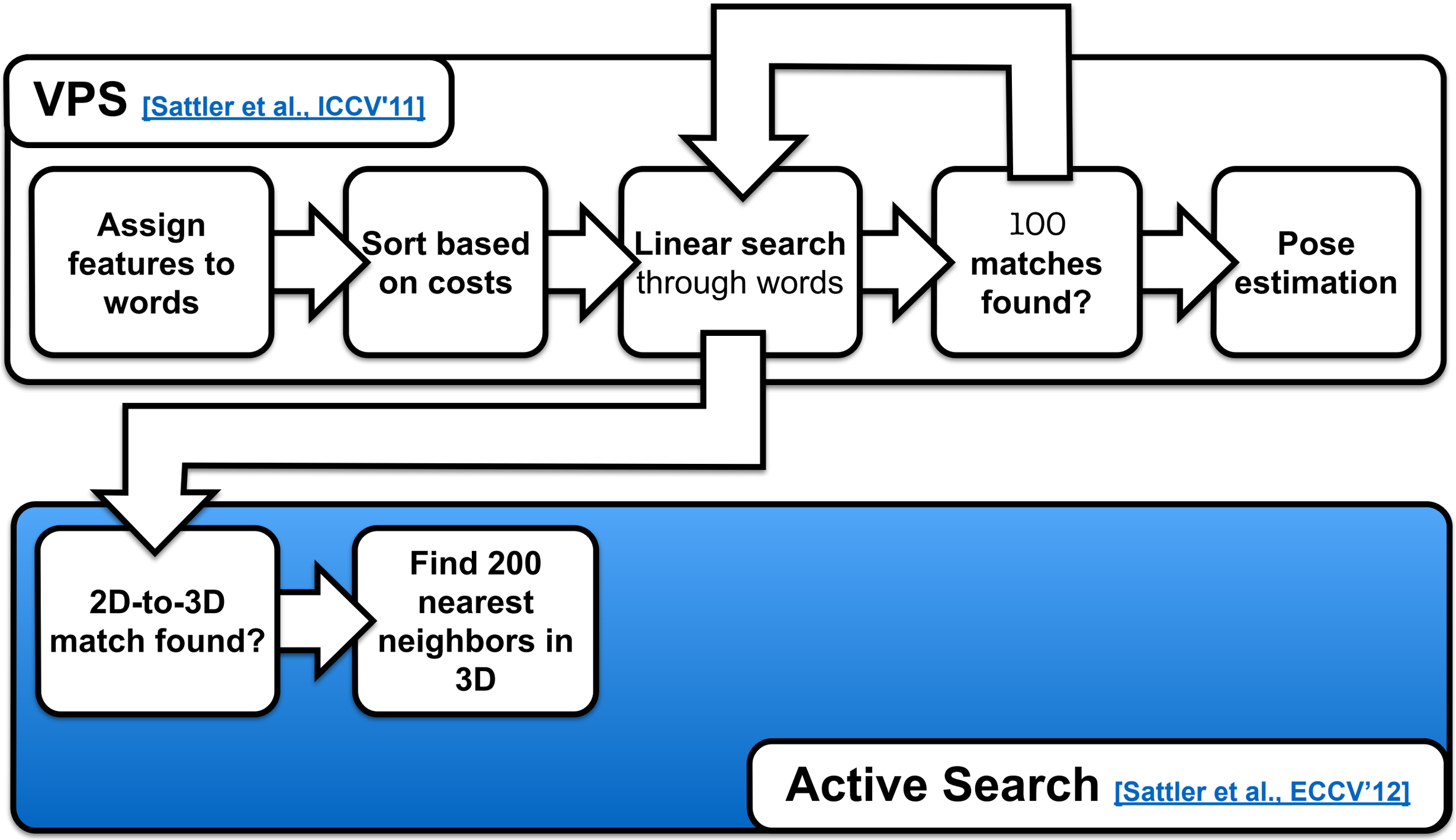
Integration Into VPS



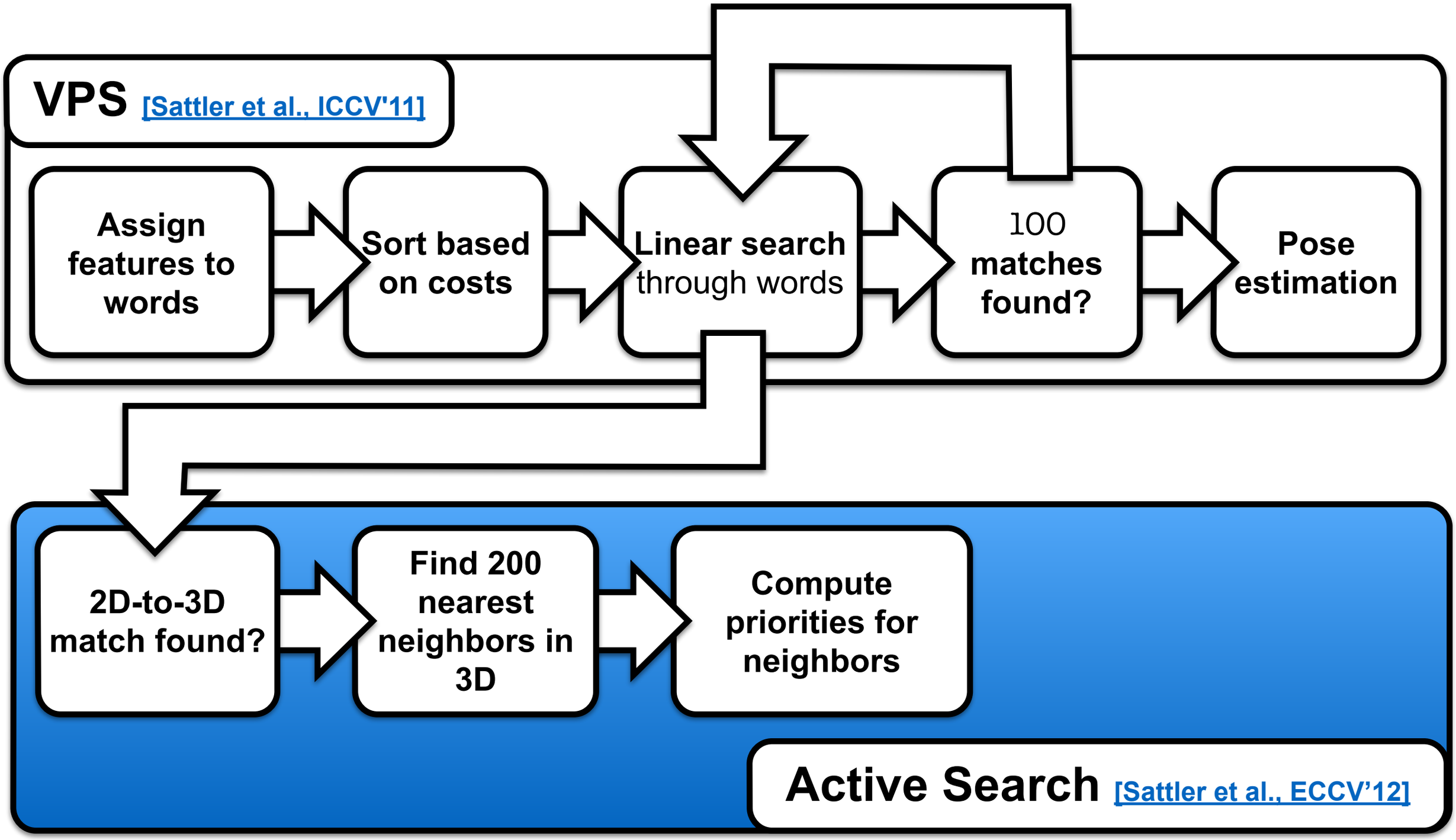
Integration Into VPS



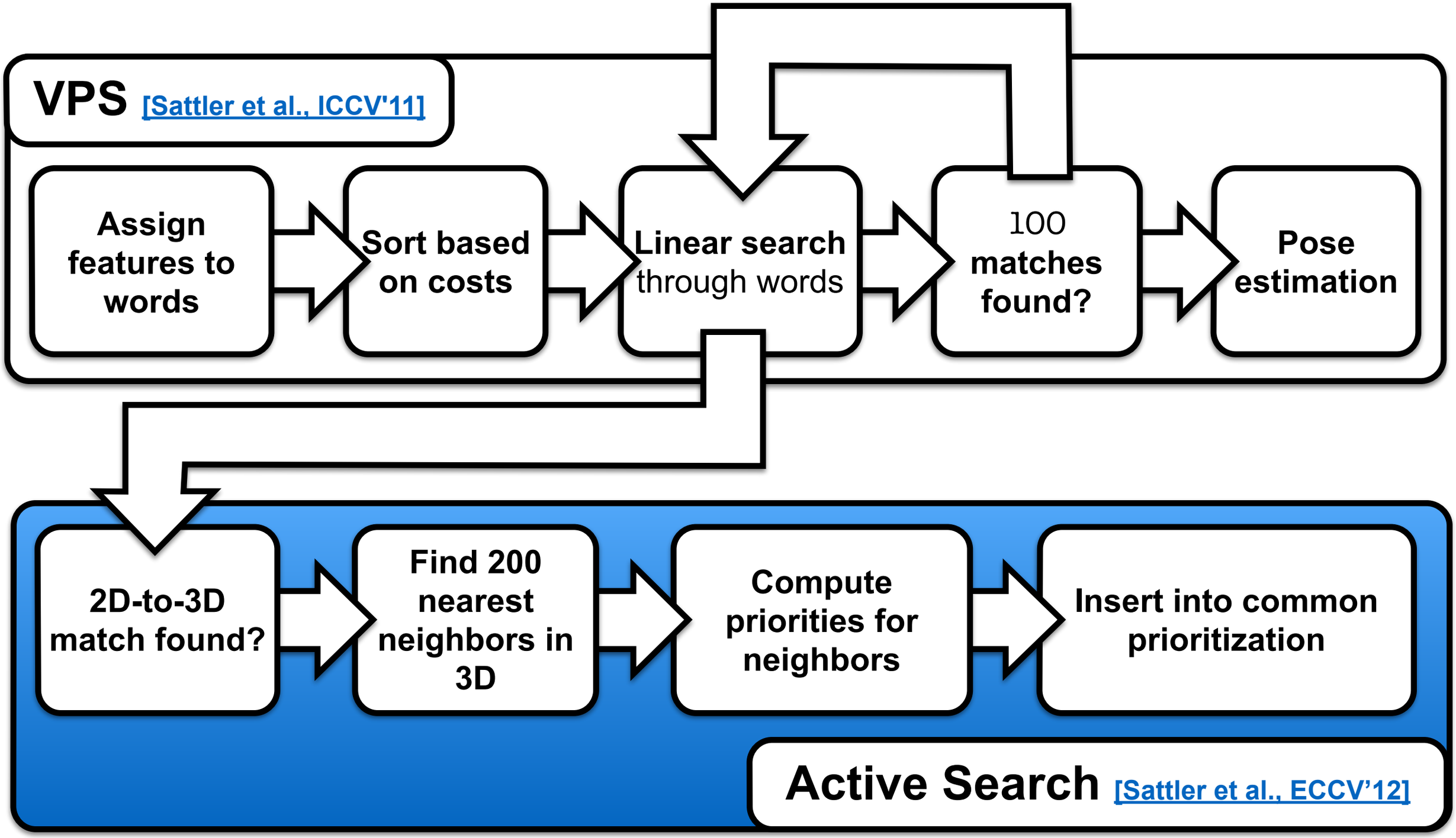
Integration Into VPS



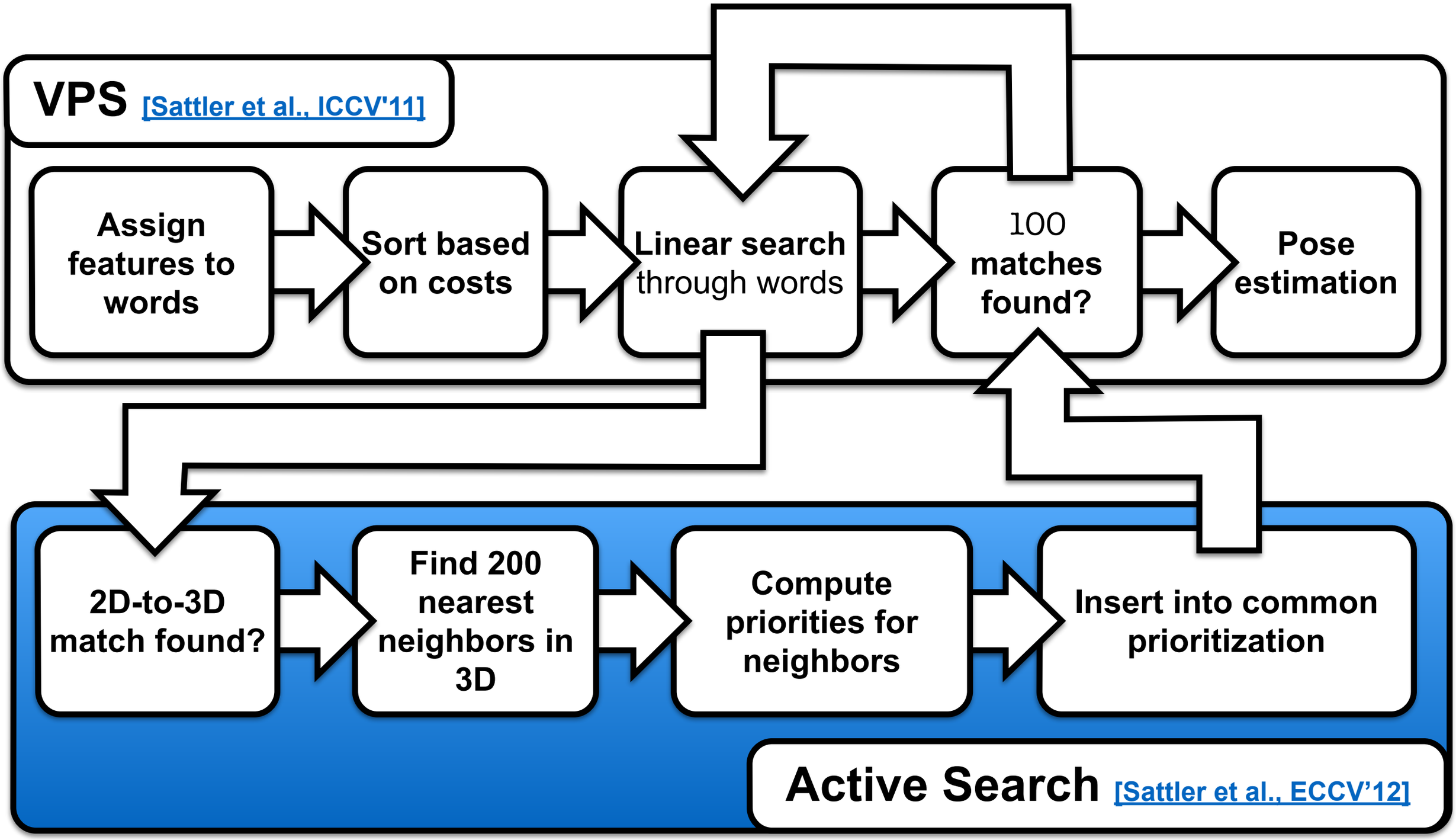
Integration Into VPS



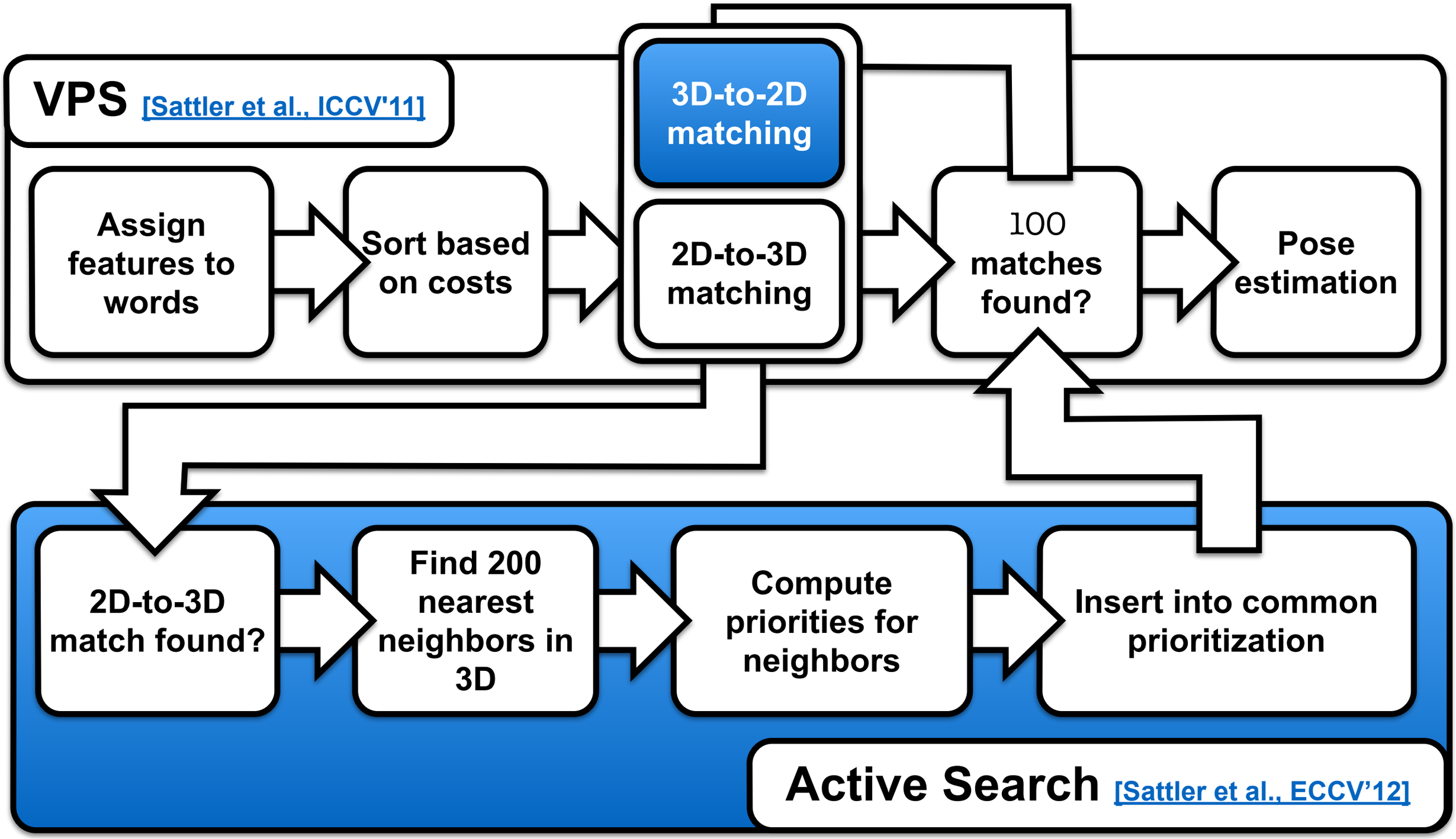
Integration Into VPS



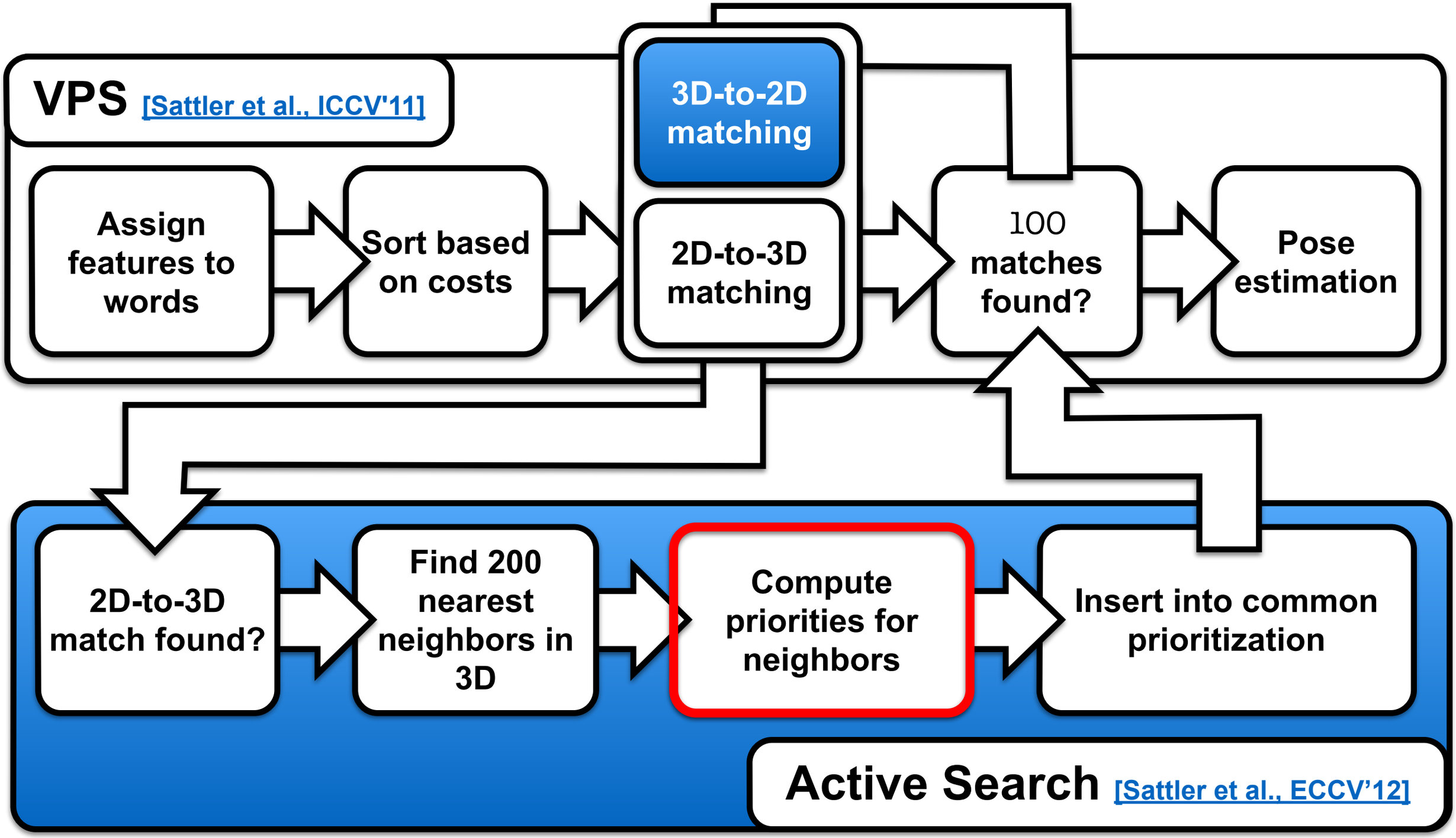
Integration Into VPS



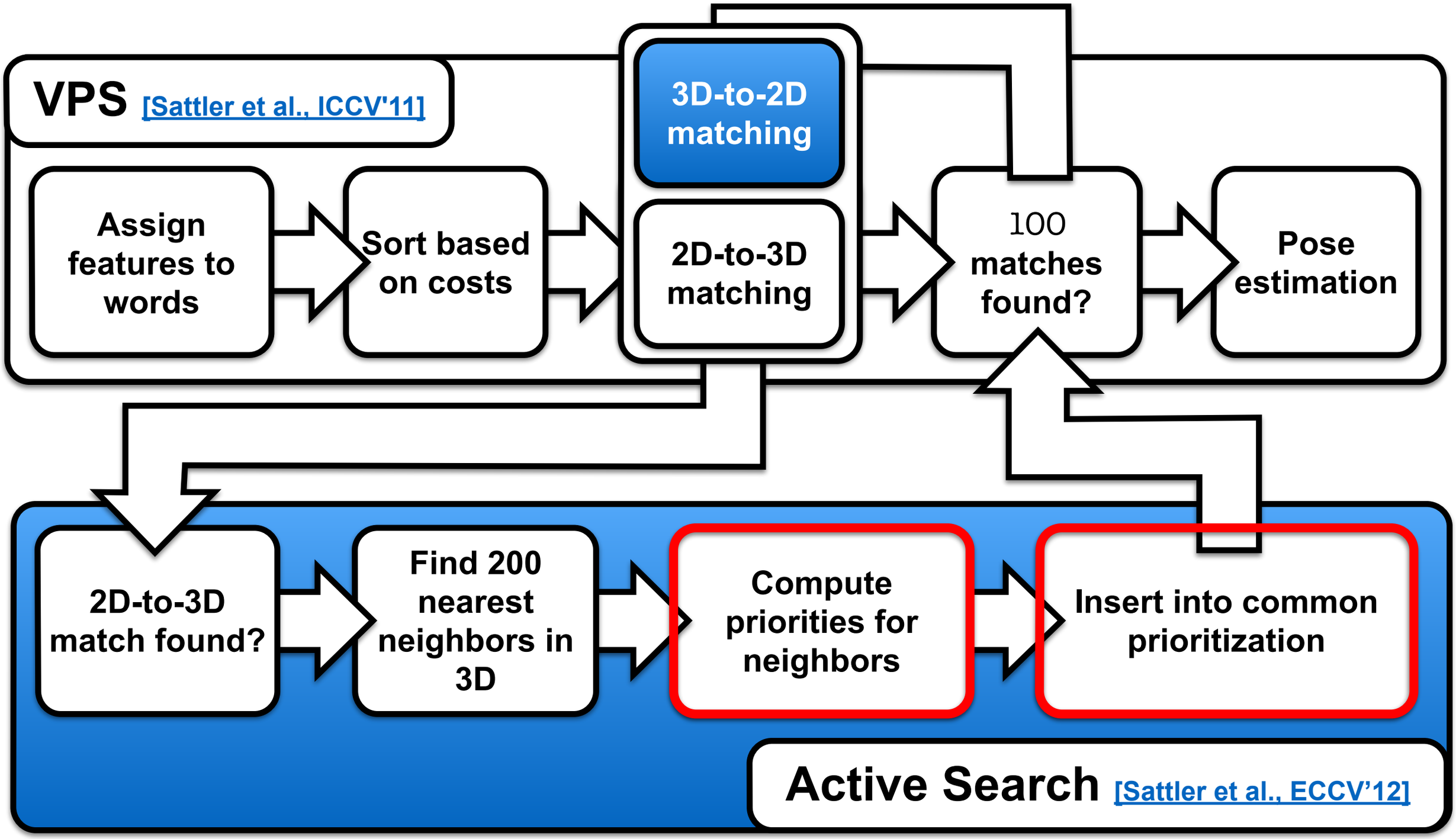
Integration Into VPS



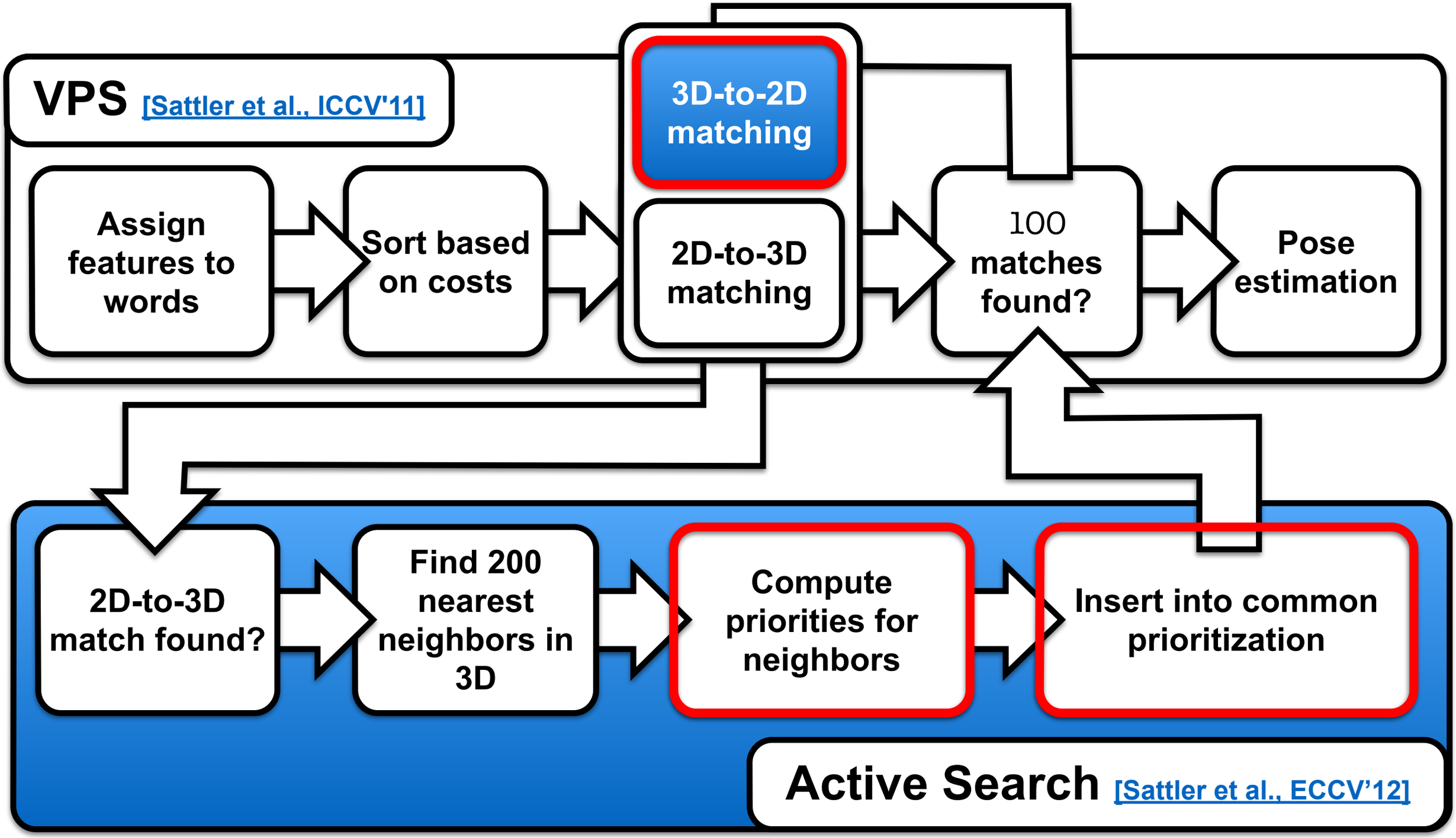
Integration Into VPS



Integration Into VPS

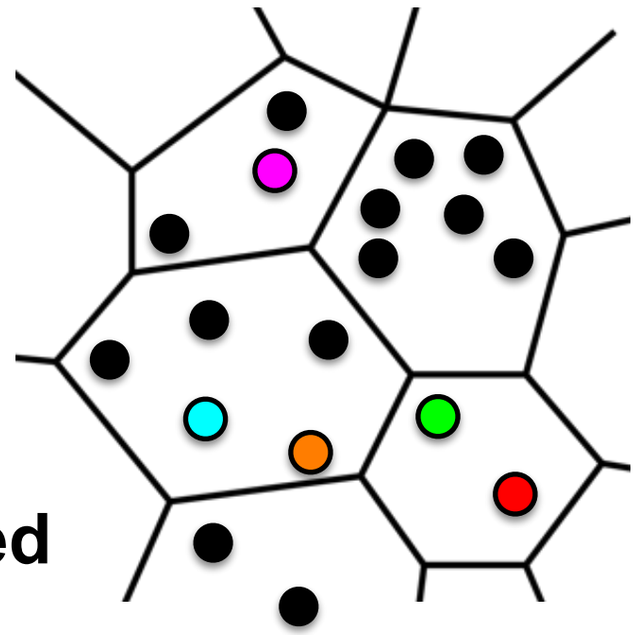


Integration Into VPS



Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

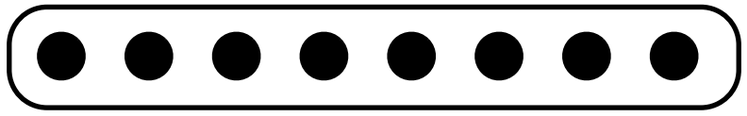


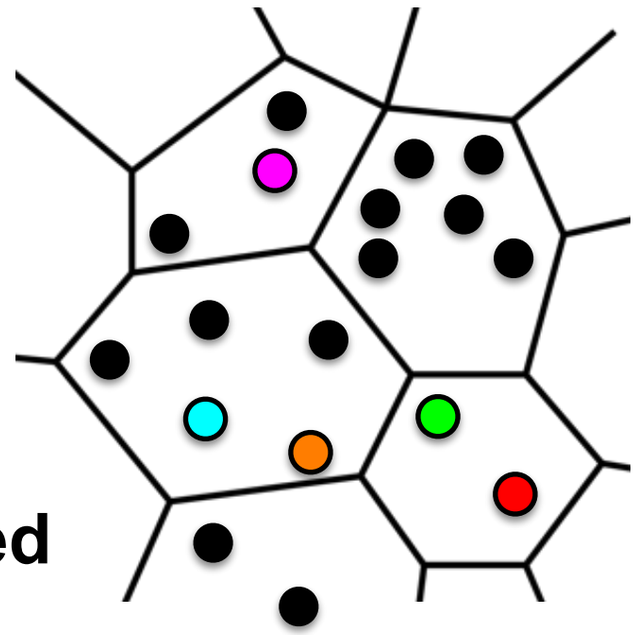
Large Vocabulary required for VPS (100k words)

Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

Vocabulary Tree



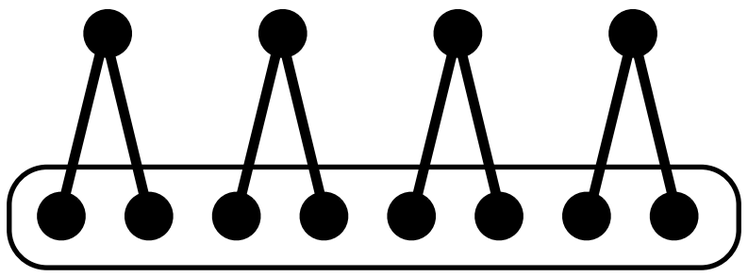


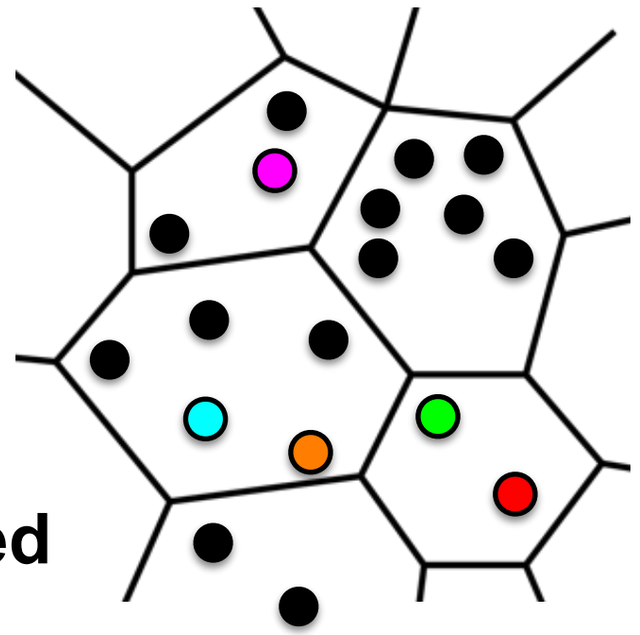
Large Vocabulary required for VPS (100k words)

Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

Vocabulary Tree



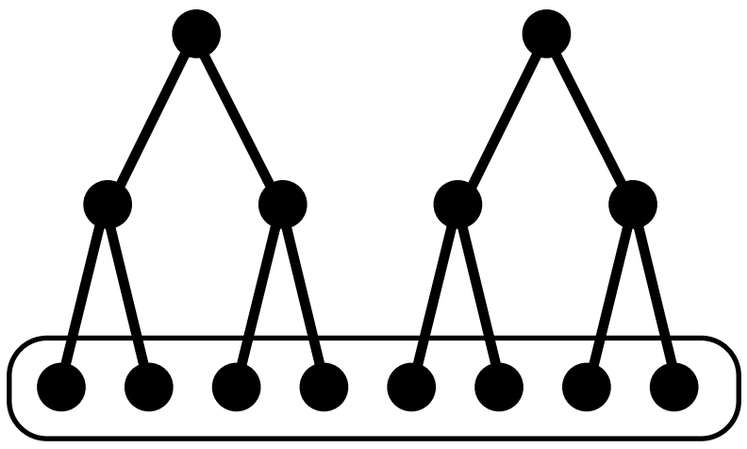


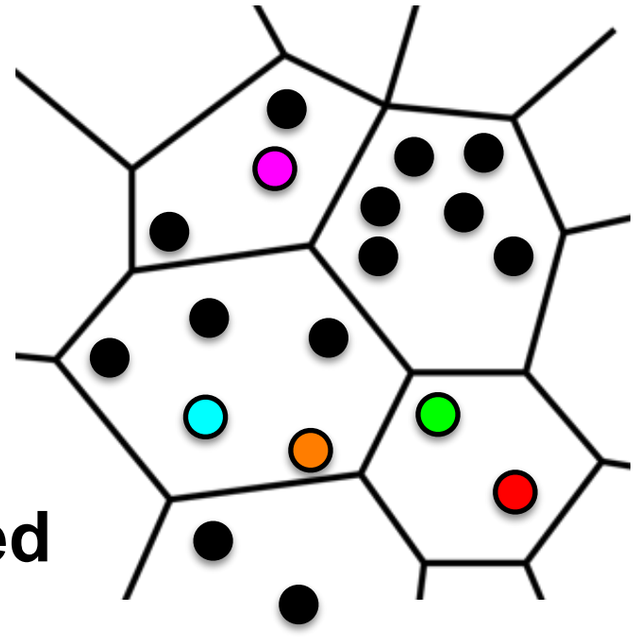
Large Vocabulary required for VPS (100k words)

Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

Vocabulary Tree

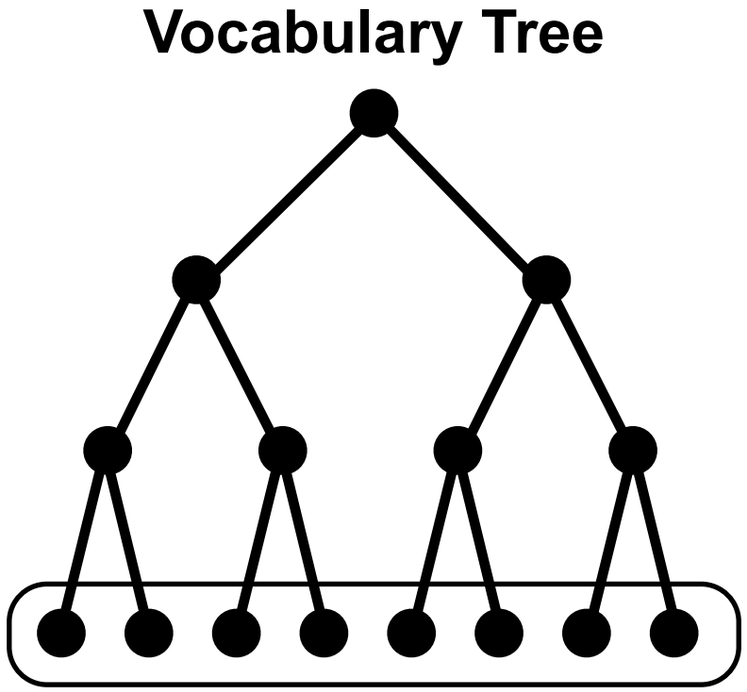


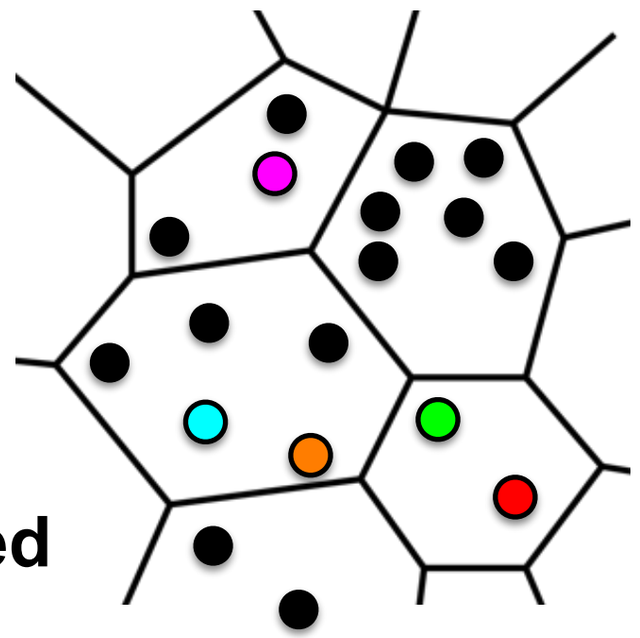


Large Vocabulary required for VPS (100k words)

Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures



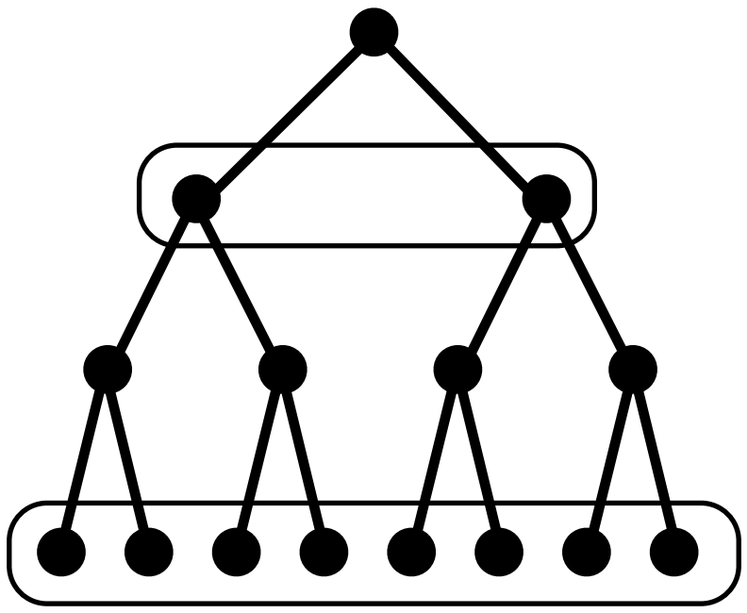


Large Vocabulary required for VPS (100k words)

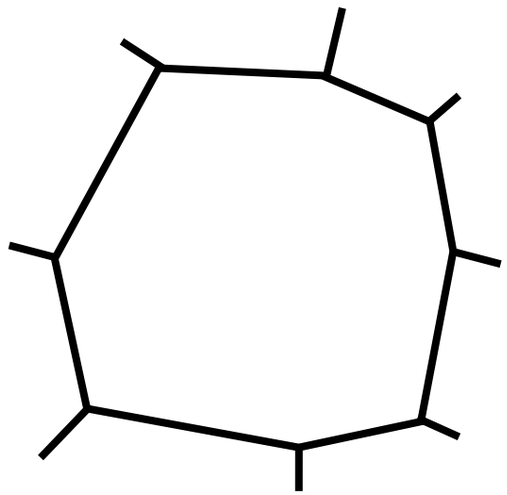
Integration 3D-to-2D Matching

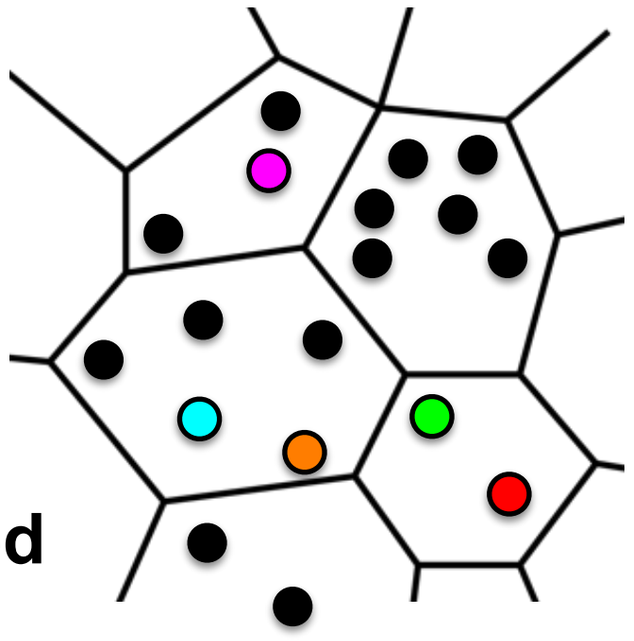
- Reduce quantization artifacts
- Reuse existing data structures

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

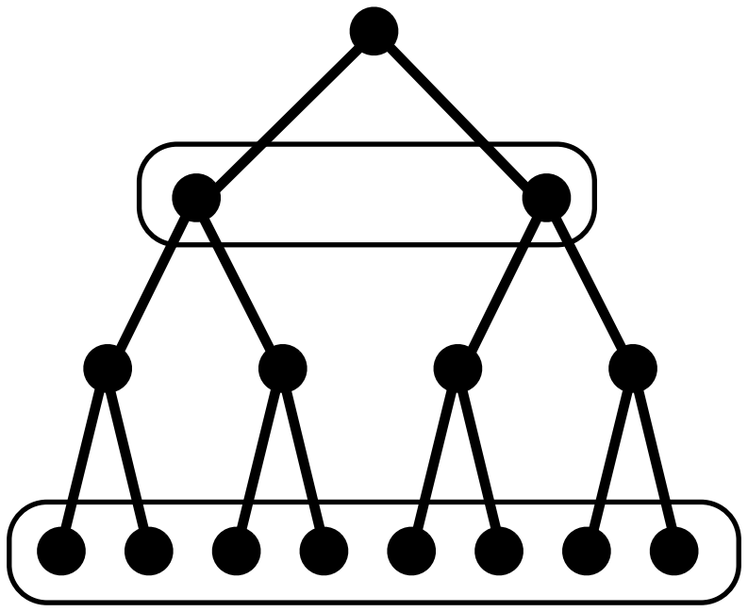
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

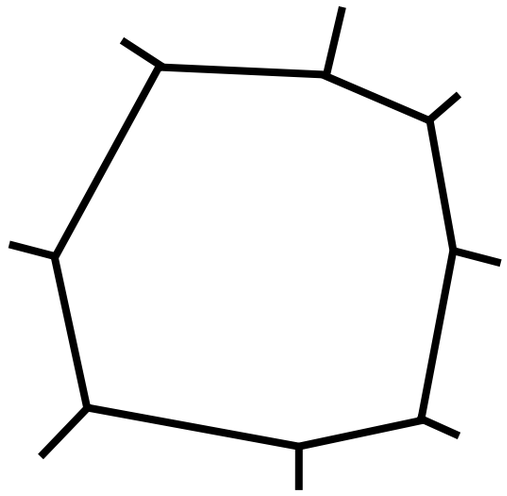


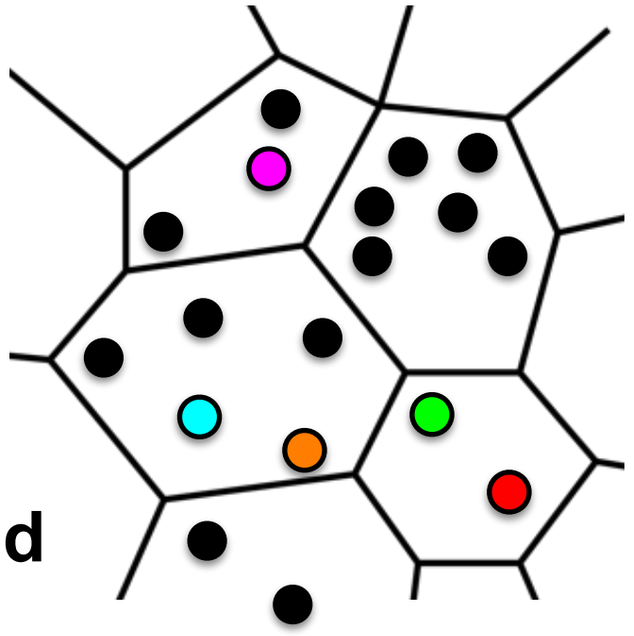
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

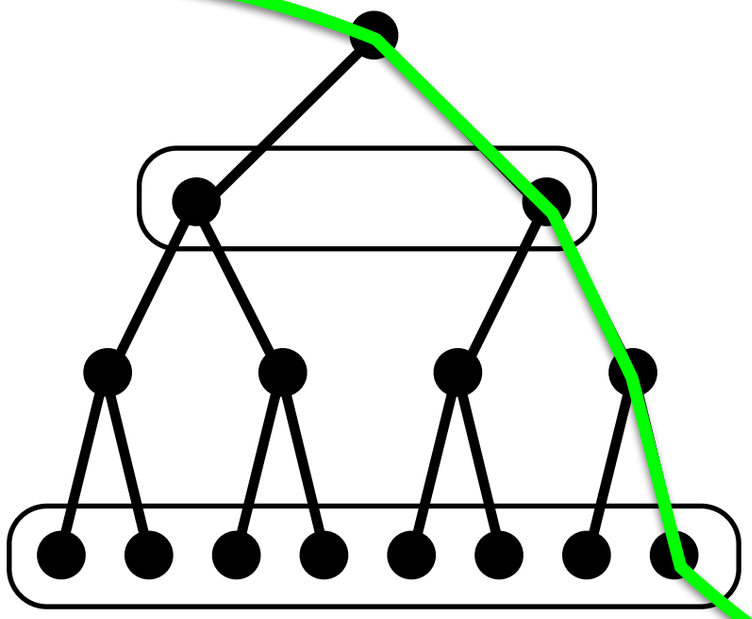
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

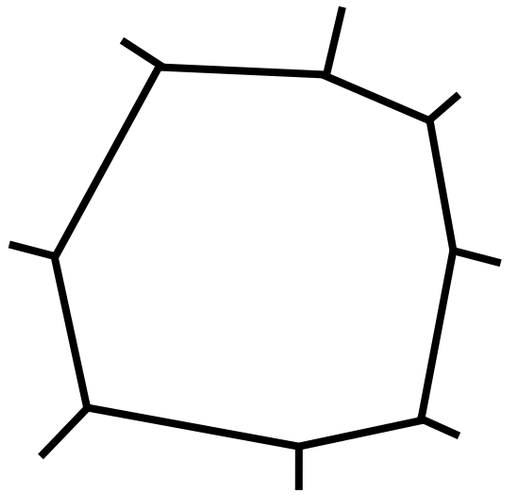


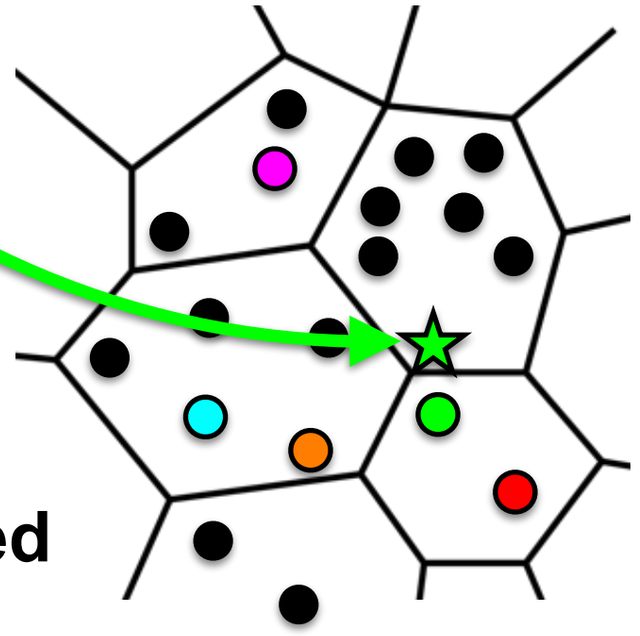
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





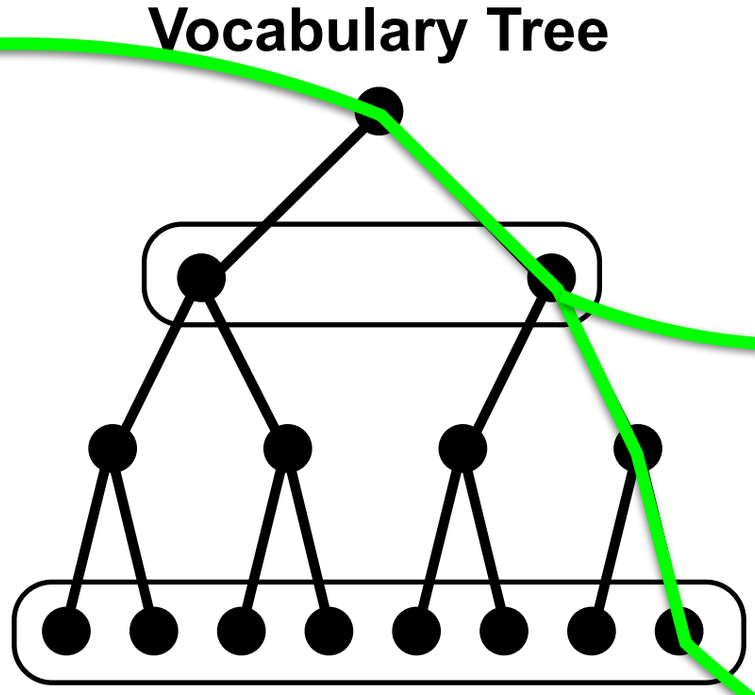
Large Vocabulary required for VPS (100k words)

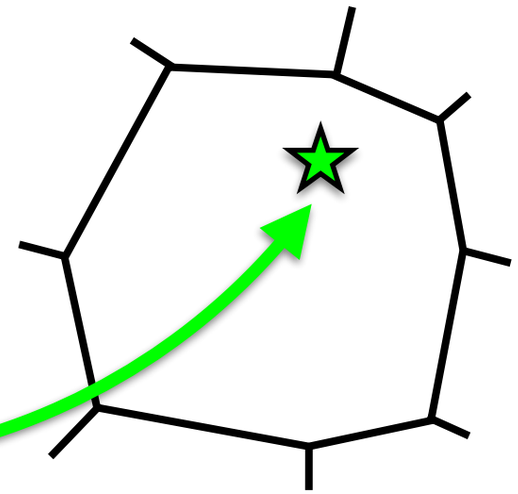
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

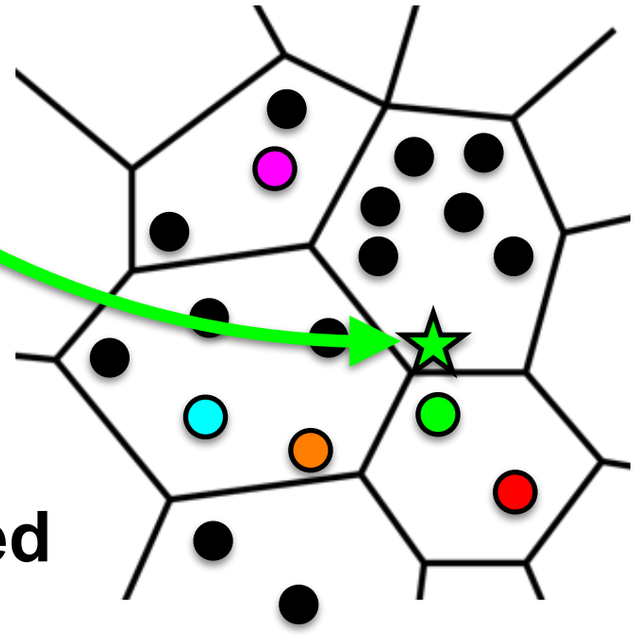


Query Image





Small Vocabulary for 3D-to-2D search (100-1k words)



Large Vocabulary required for VPS (100k words)

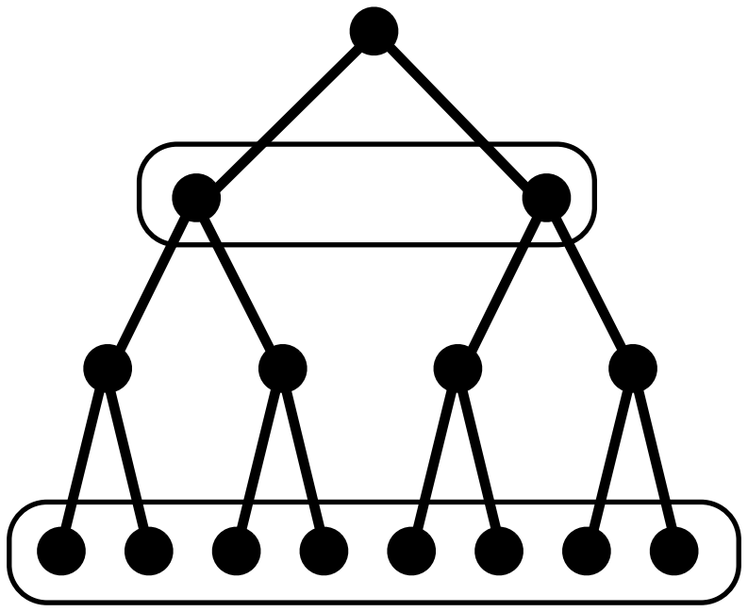
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

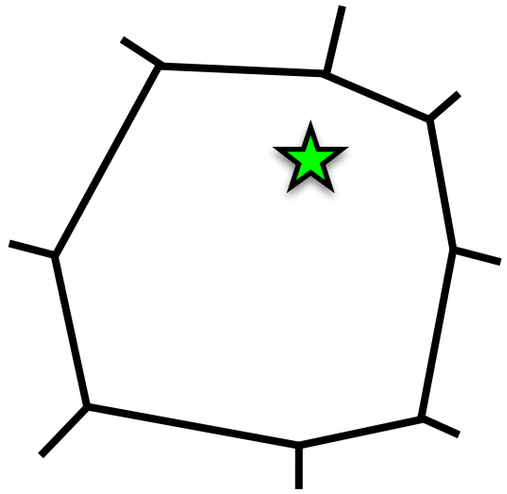


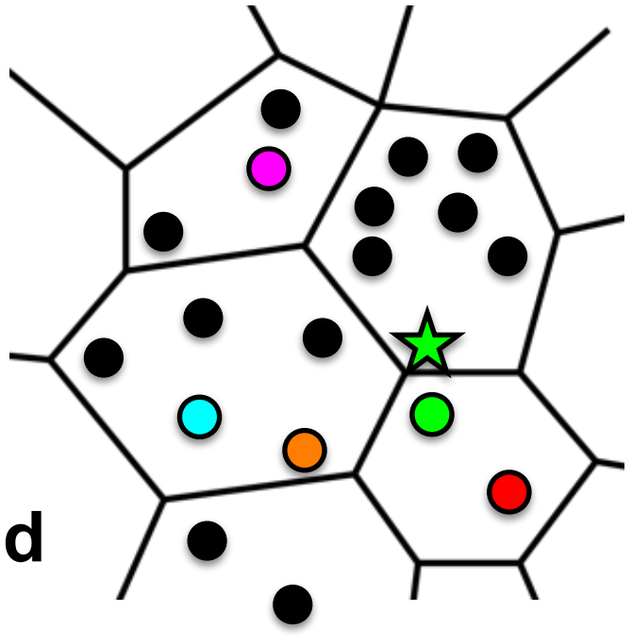
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

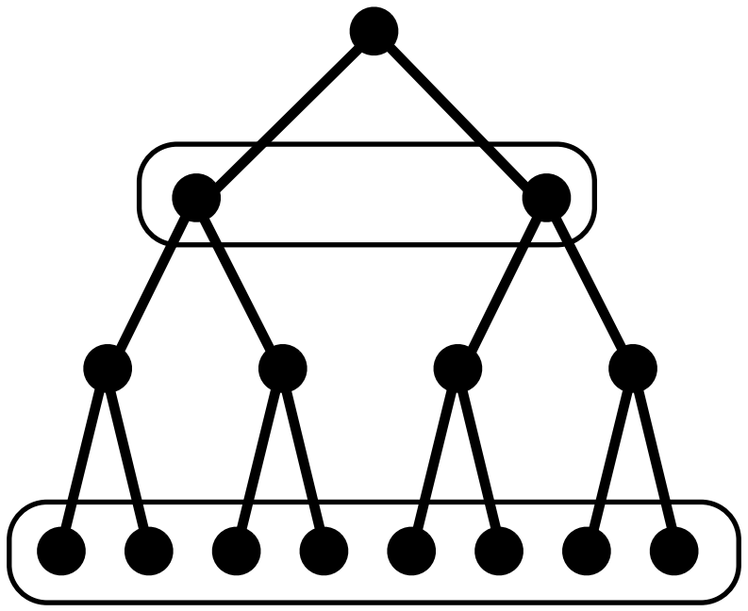
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

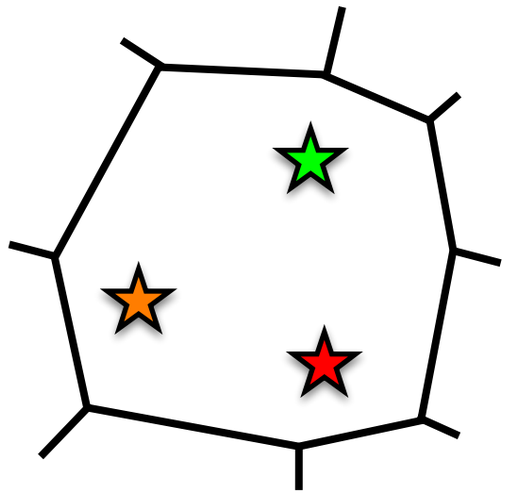


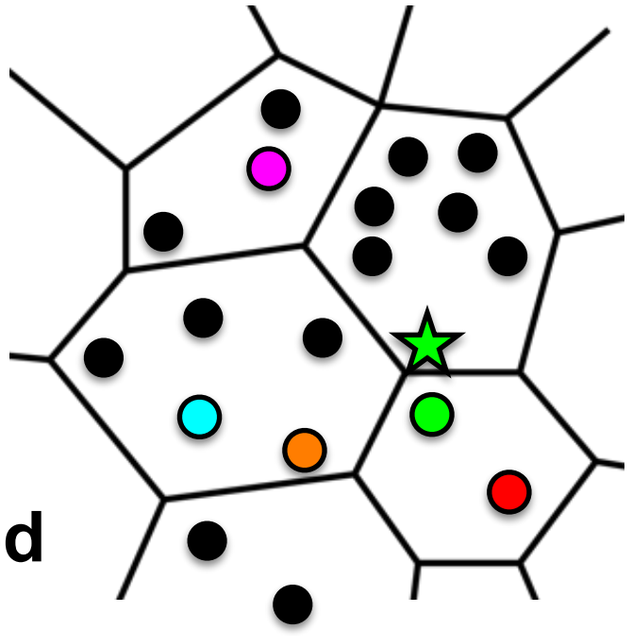
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

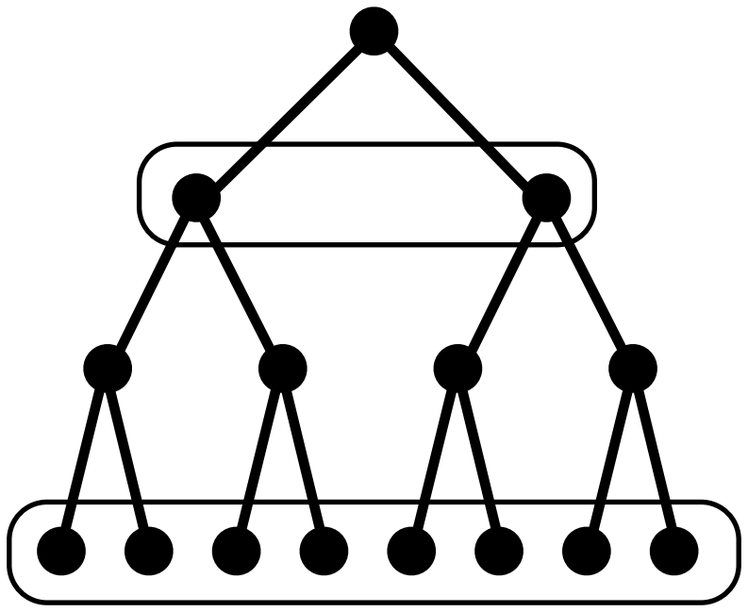
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

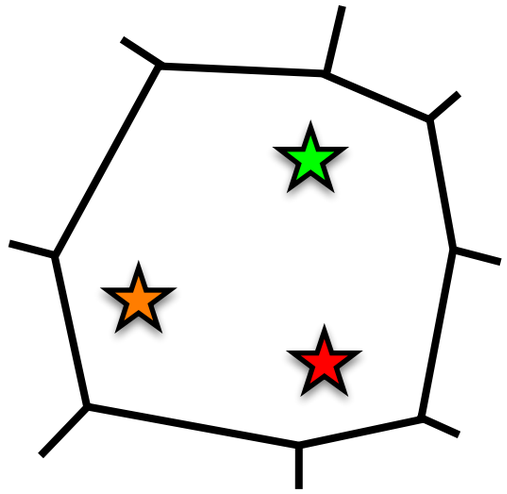


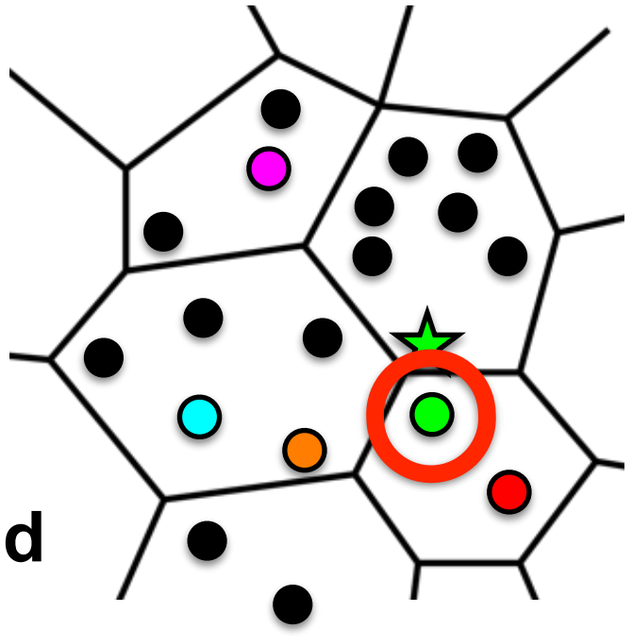
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

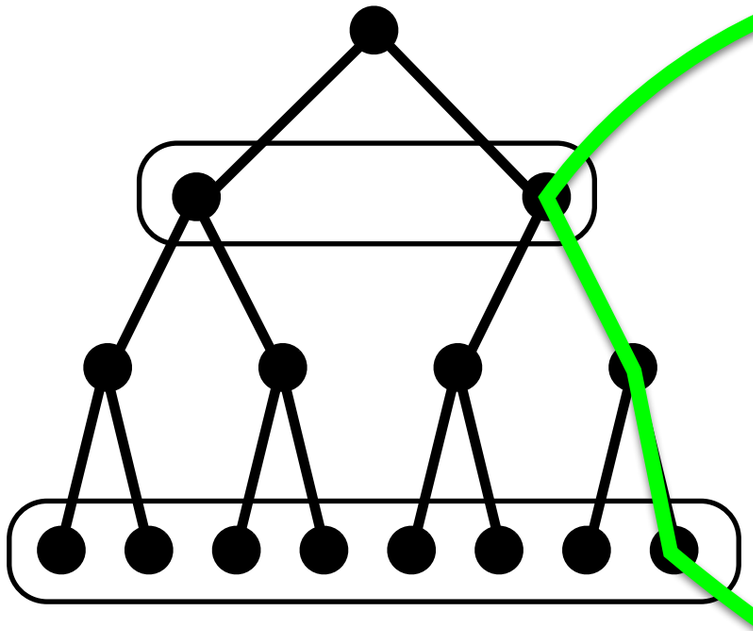
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

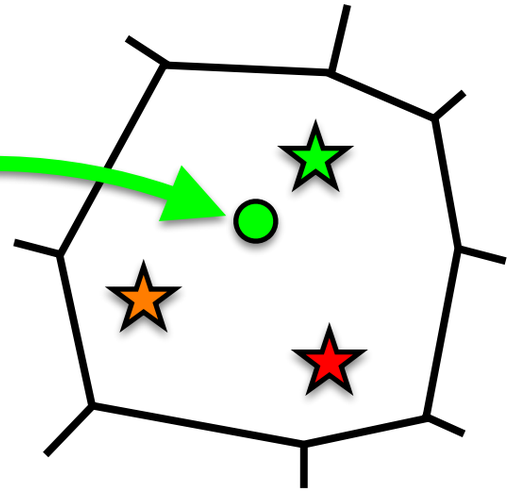


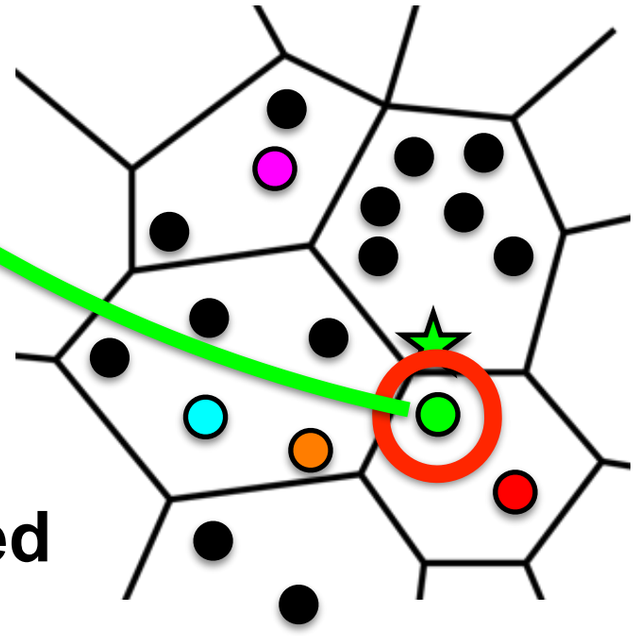
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

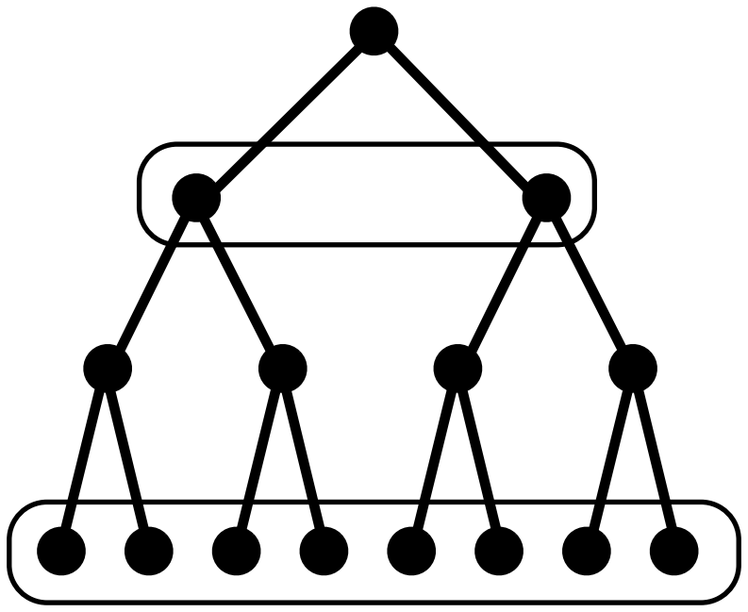
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

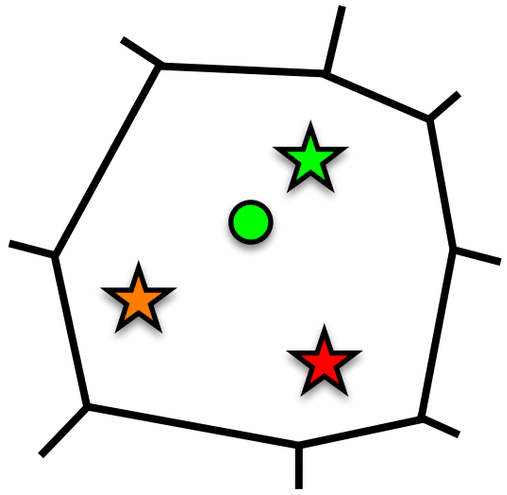


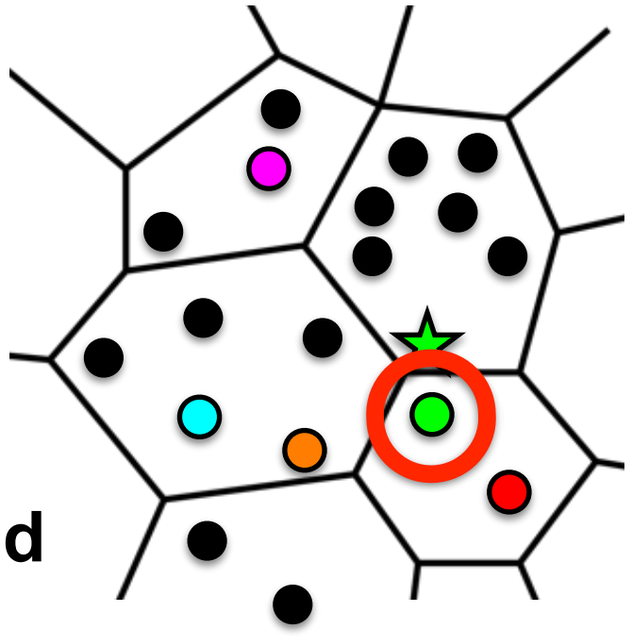
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)





Large Vocabulary required for VPS (100k words)

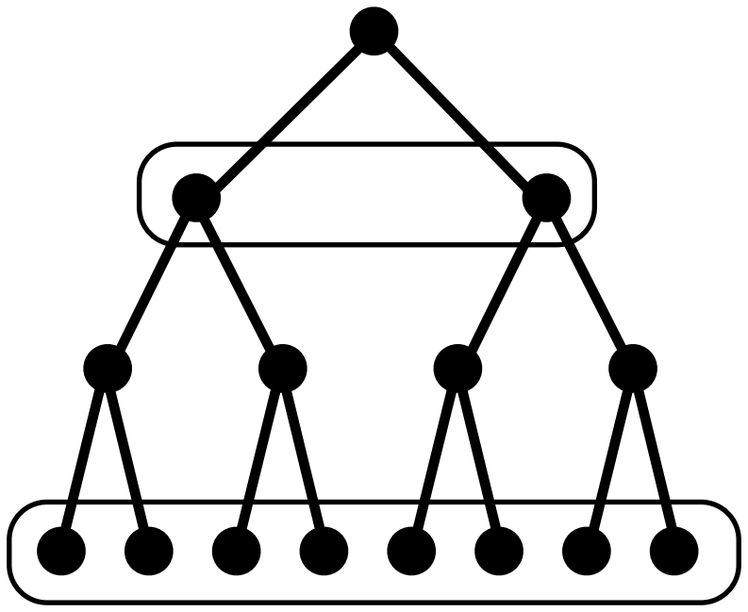
Integration 3D-to-2D Matching

- Reduce quantization artifacts
- Reuse existing data structures

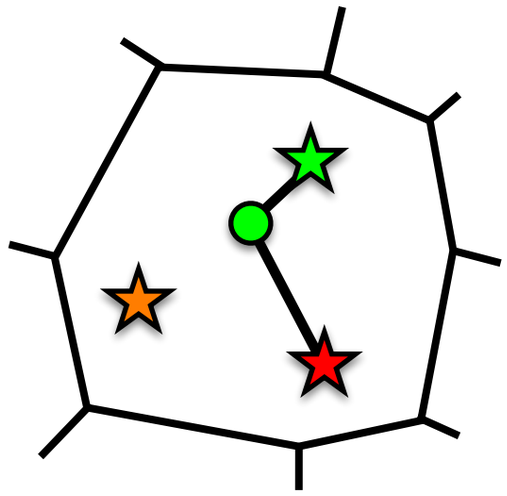


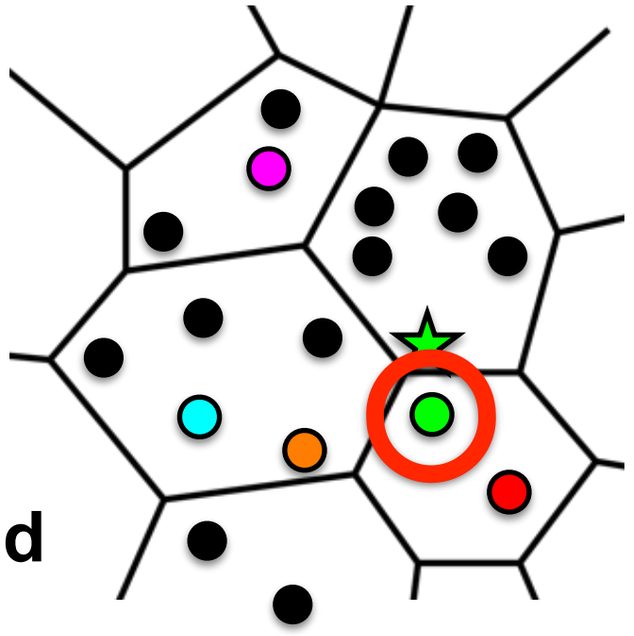
Query Image

Vocabulary Tree



Small Vocabulary for 3D-to-2D search (100-1k words)

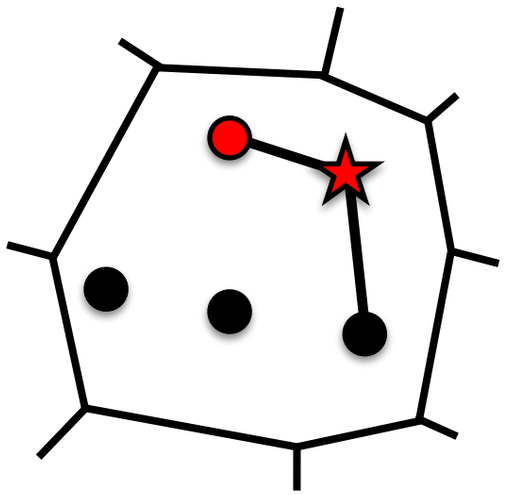




Large Vocabulary required for VPS (100k words)

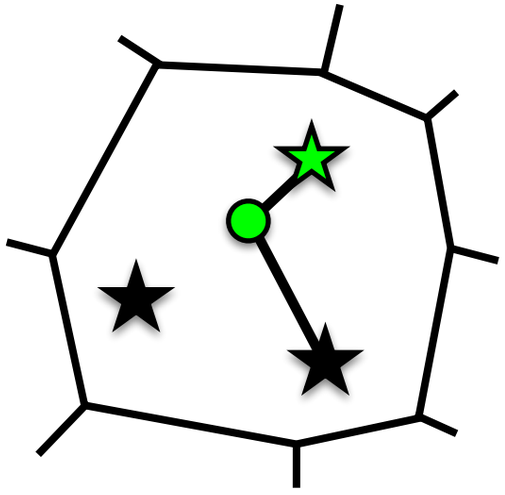
Common Prioritization Scheme

2D-to-3D Matching



- Find neighbors inside visual word
- Same definition of search costs

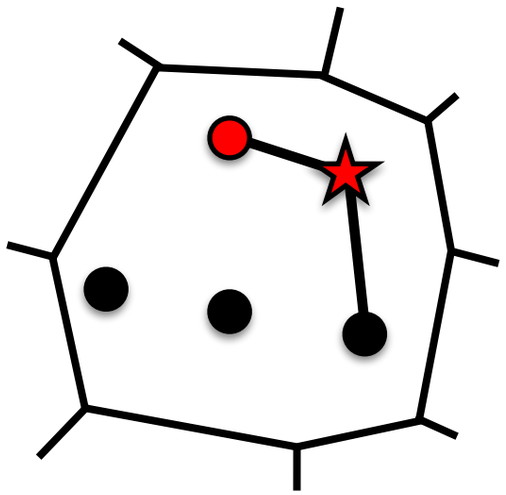
3D-to-2D Matching



[Sattler et al., ECCV'12]

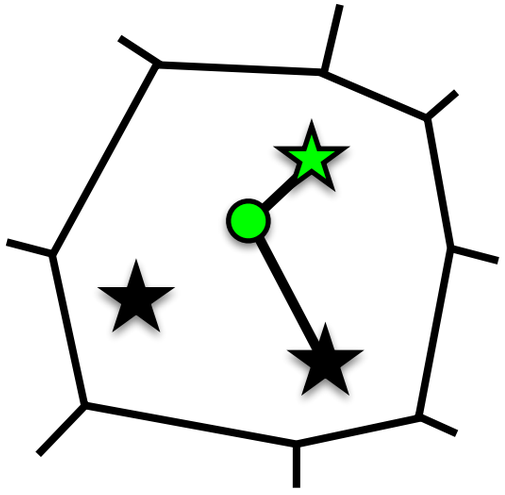
Common Prioritization Scheme

2D-to-3D Matching

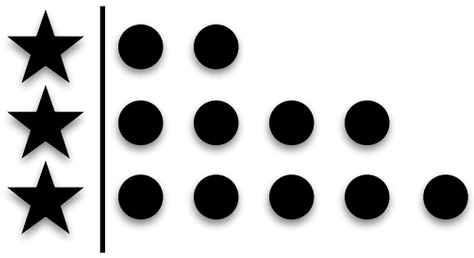


- Find neighbors inside visual word
- Same definition of search costs

3D-to-2D Matching



Priorities



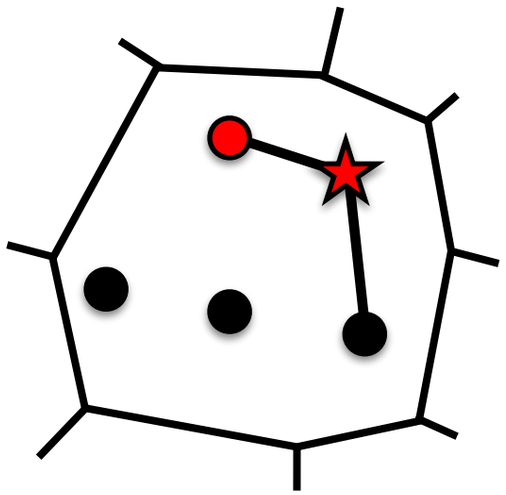
Priorities



[Sattler et al., ECCV'12]

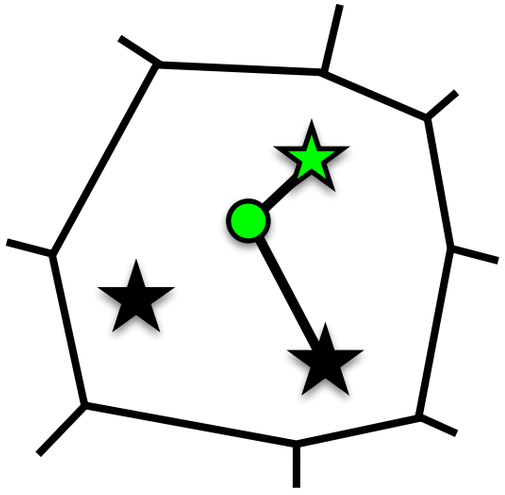
Common Prioritization Scheme

2D-to-3D Matching

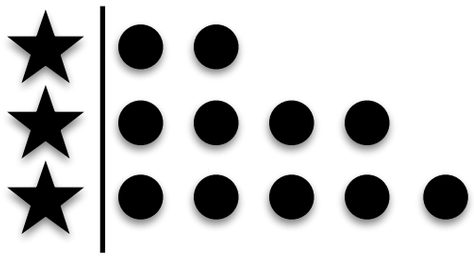


- Find neighbors inside visual word
- Same definition of search costs

3D-to-2D Matching



Priorities



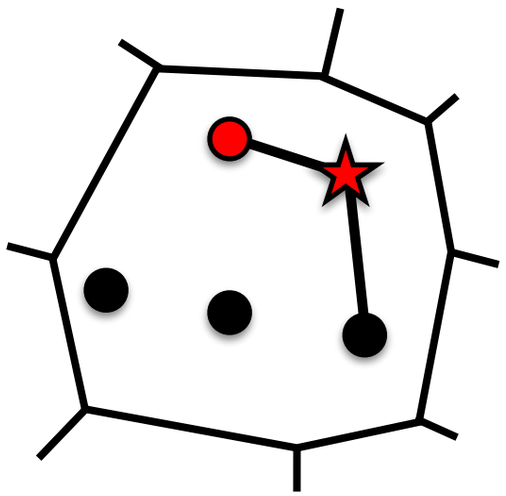
Priorities



[Sattler et al., ECCV'12]

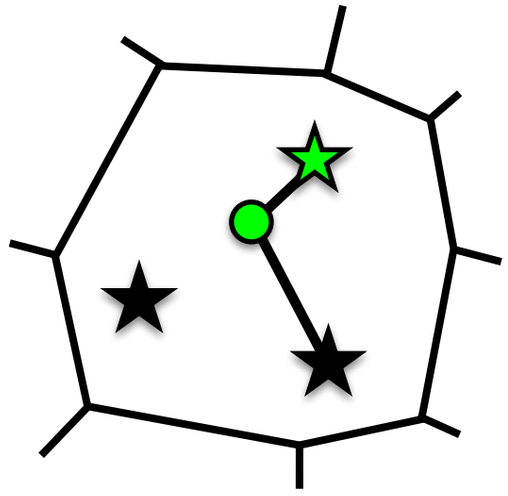
Common Prioritization Scheme

2D-to-3D Matching

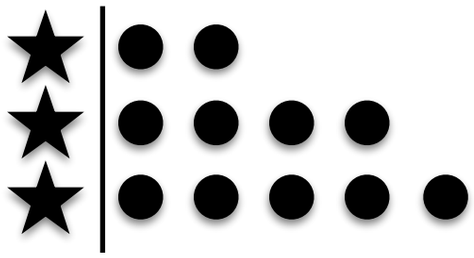


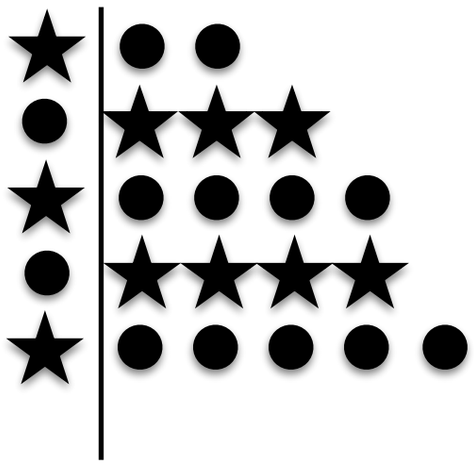
- Find neighbors inside visual word
- Same definition of search costs

3D-to-2D Matching



Priorities





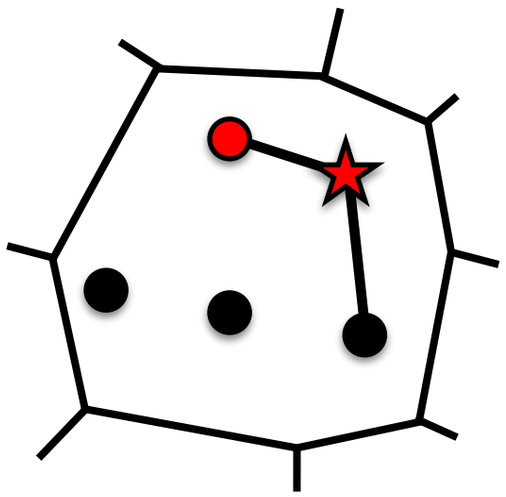
Priorities



[Sattler et al., ECCV'12]

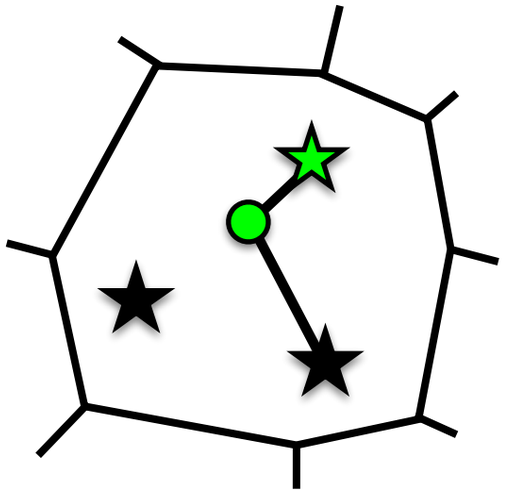
Common Prioritization Scheme

2D-to-3D Matching

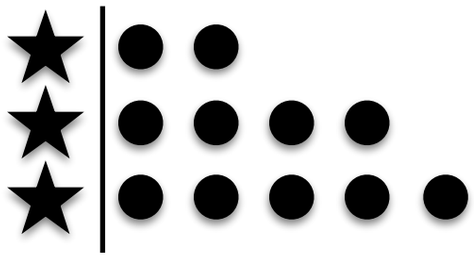


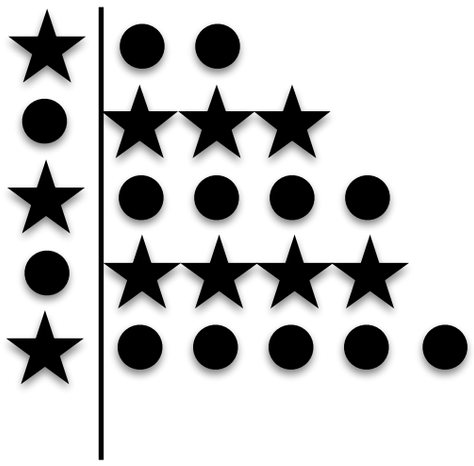
- Find neighbors inside visual word
- Same definition of search costs

3D-to-2D Matching



Priorities





Priorities



Common prioritization: Prefer cheaper search direction

[Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

Assign features to words → **Sort based on costs** → **3D-to-2D matching** / **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → **Find 200 nearest neighbors in 3D** → **Compute priorities for neighbors** → **Insert into common prioritization**

Active Search [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

Assign features to words → **Sort based on costs** → **3D-to-2D matching** / **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → **Find 200 nearest neighbors in 3D** → **Compute priorities for neighbors** → **Insert into common prioritization**

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11] $\mathcal{O}(1)\cdot$

Assign features to words → **Sort based on costs** → **3D-to-2D matching** / **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → **Find 200 nearest neighbors in 3D** → **Compute priorities for neighbors** → **Insert into common prioritization**

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$ $\mathcal{O}(F)$

Assign features to words → **Sort based on costs** → **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → **Find 200 nearest neighbors in 3D** → **Compute priorities for neighbors** → **Insert into common prioritization**

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$ $\mathcal{O}(1)$

Assign features to words → **Sort based on costs** → **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → **Find 200 nearest neighbors in 3D** → **Compute priorities for neighbors** → **Insert into common prioritization**

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$

$\mathcal{O}(1)$

Assign features to words

Sort based on costs

2D-to-3D matching

100 **matches found?**

Pose estimation

2D-to-3D match found?

$\mathcal{O}(\log P)$

Compute priorities for neighbors

Insert into common prioritization

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$ $\mathcal{O}(1)$

Assign features to words → **Sort based on costs** → **2D-to-3D matching** → **100 matches found?** → **Pose estimation**

2D-to-3D match found? → $\mathcal{O}(\log P)$ → $\mathcal{O}(1)$ → **Insert into common prioritization**

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$

$\mathcal{O}(1)$

Assign features to words

Sort based on costs

2D-to-3D matching

100 **matches found?**

Pose estimation

2D-to-3D match found?

$\mathcal{O}(\log P)$

$\mathcal{O}(1)$

$\mathcal{O}(1)$

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

Assign features to words → **Sort based on costs** → $\mathcal{O}(1)\cdot$ **2D-to-3D matching** → $\mathcal{O}(1)$ **100 matches found?** → **Pose estimation**

$\mathcal{O}(\log P)$

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$

$\mathcal{O}(1)$

Total effort added by Active Search: $\mathcal{O}(\log P)$

$\mathcal{O}(\log P)$

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

Computational Complexity

P = number 3D points

F = number 2D features

W = number visual words

VPS [Sattler et al., ICCV'11]

$\mathcal{O}(1)\cdot$

$\mathcal{O}(1)$

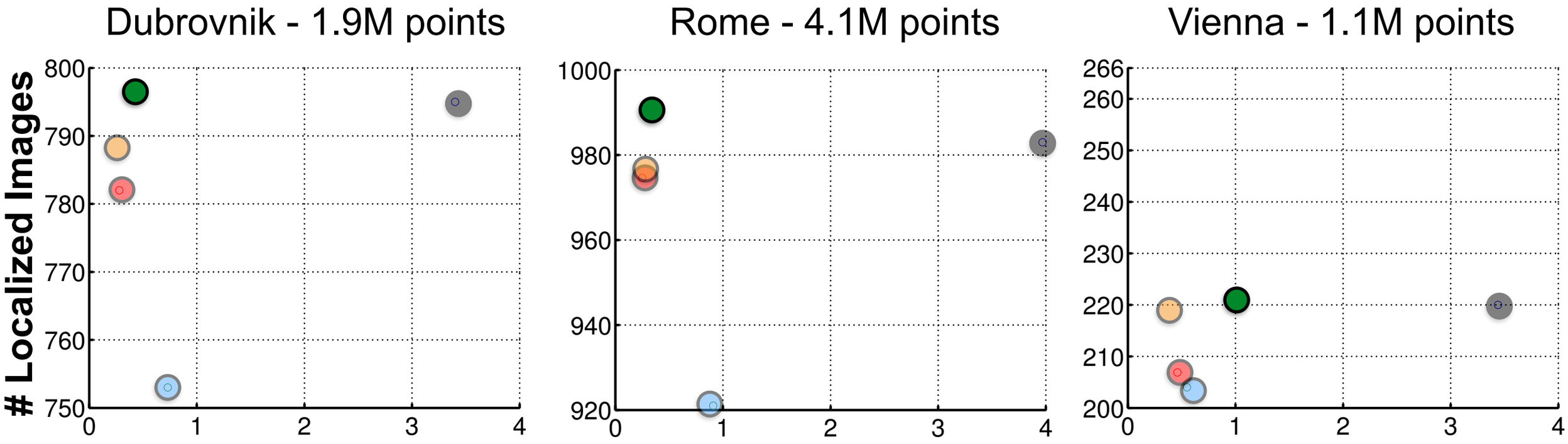
Total effort added by Active Search: $\mathcal{O}(\log P)$

Effort added by Soft Assignments: $\mathcal{O}(P/W)$ **per feature**

$\mathcal{O}(\log P)$

$\mathcal{O}(1)\cdot$ **Active Search** [Sattler et al., ECCV'12]

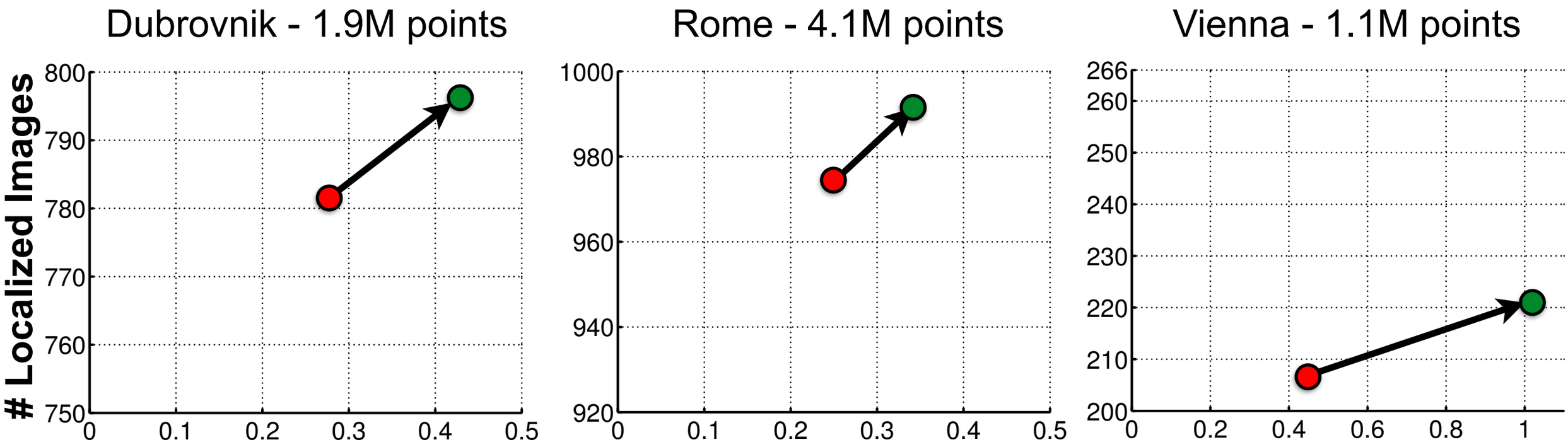
Results



✓ As effective as kd-tree or better

✗ Less effective than VPS due to additional computations

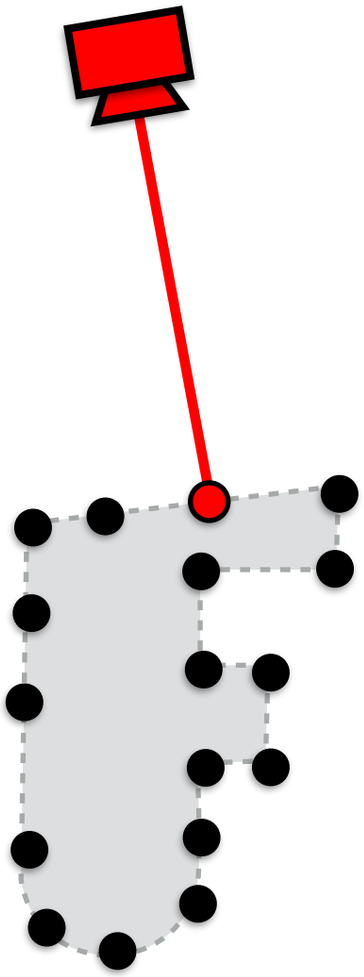
Results



✓ As effective as kd-tree or better

✗ Less effective than VPS due to additional computations

Visibility Filtering

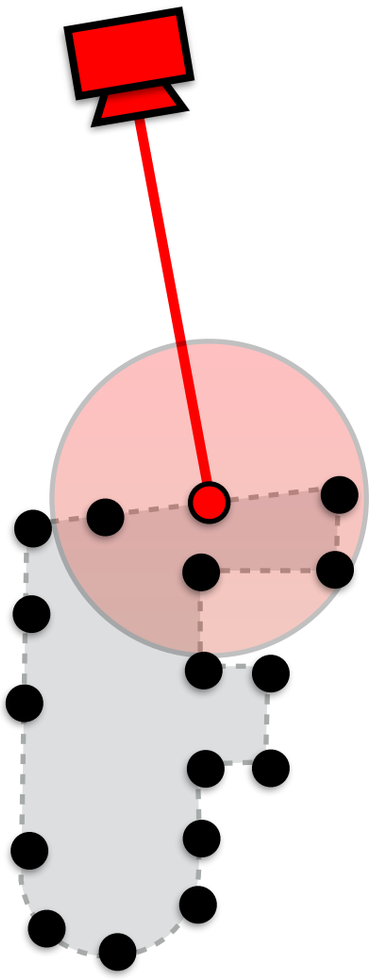


Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

[Sattler et al., ECCV'12]

Visibility Filtering

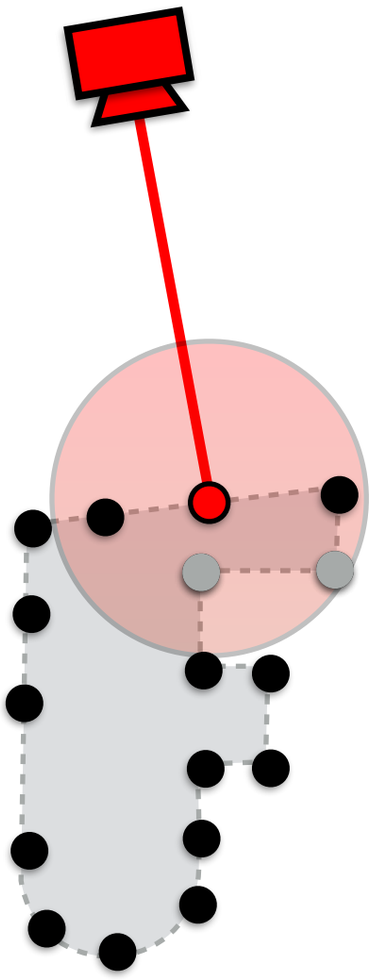


Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

[Sattler et al., ECCV'12]

Visibility Filtering

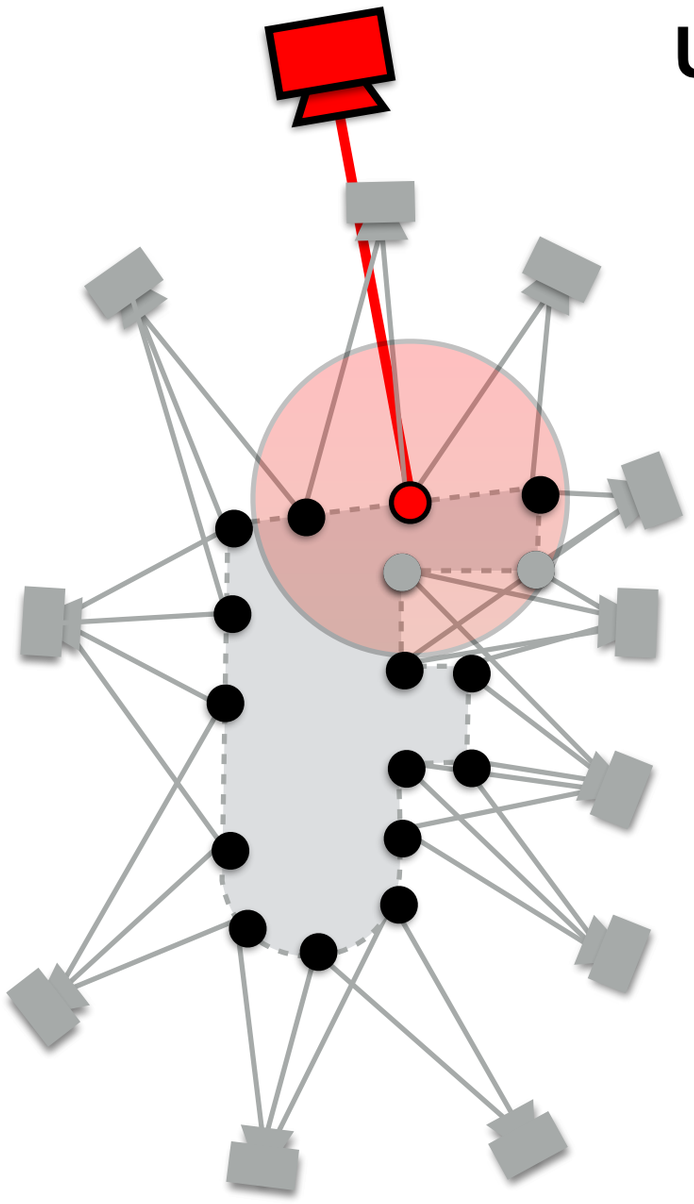


Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

[Sattler et al., ECCV'12]

Visibility Filtering

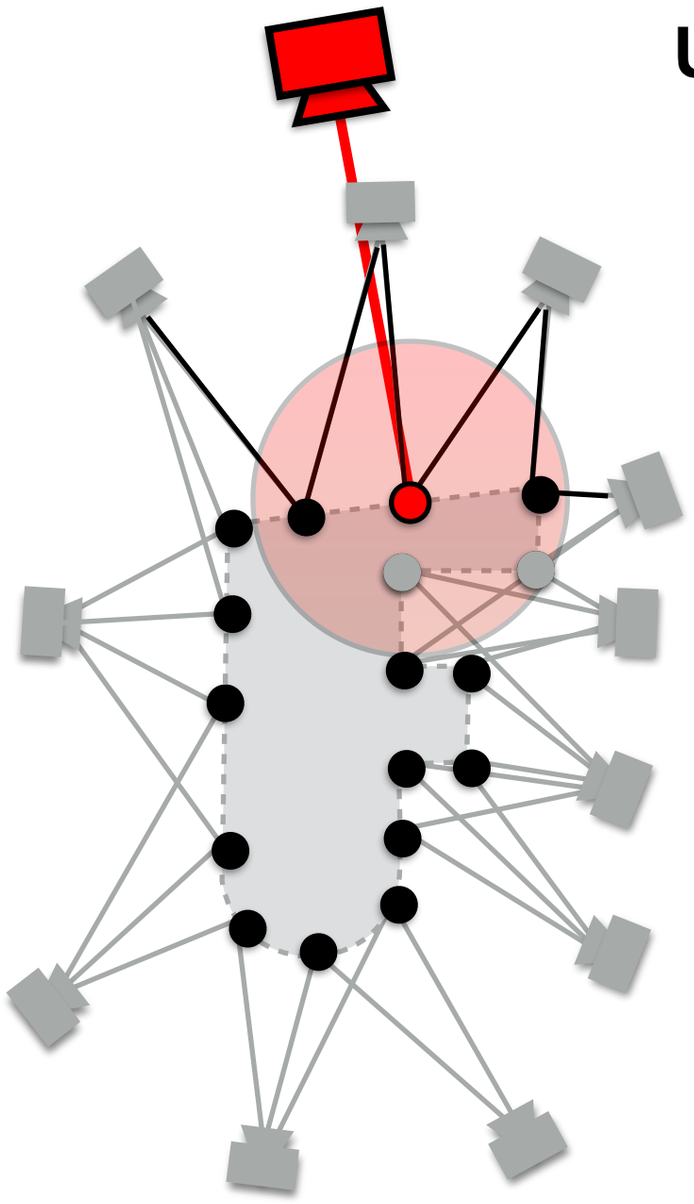


Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

[Sattler et al., ECCV'12]

Visibility Filtering

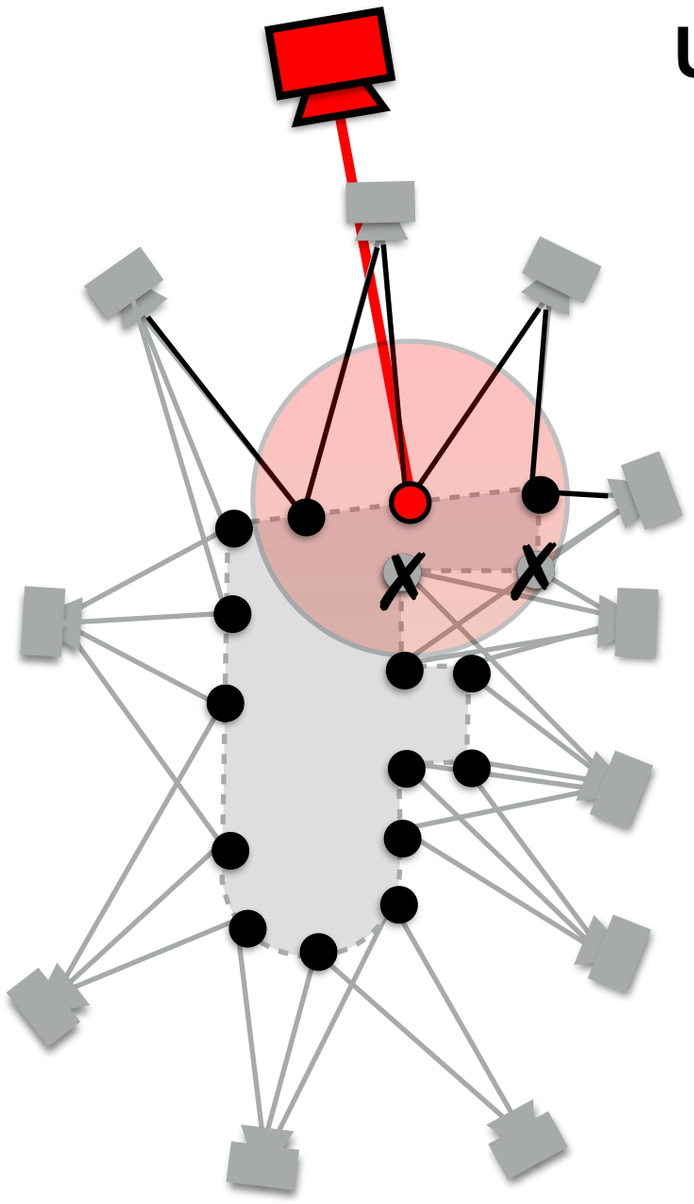


Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

[Sattler et al., ECCV'12]

Visibility Filtering



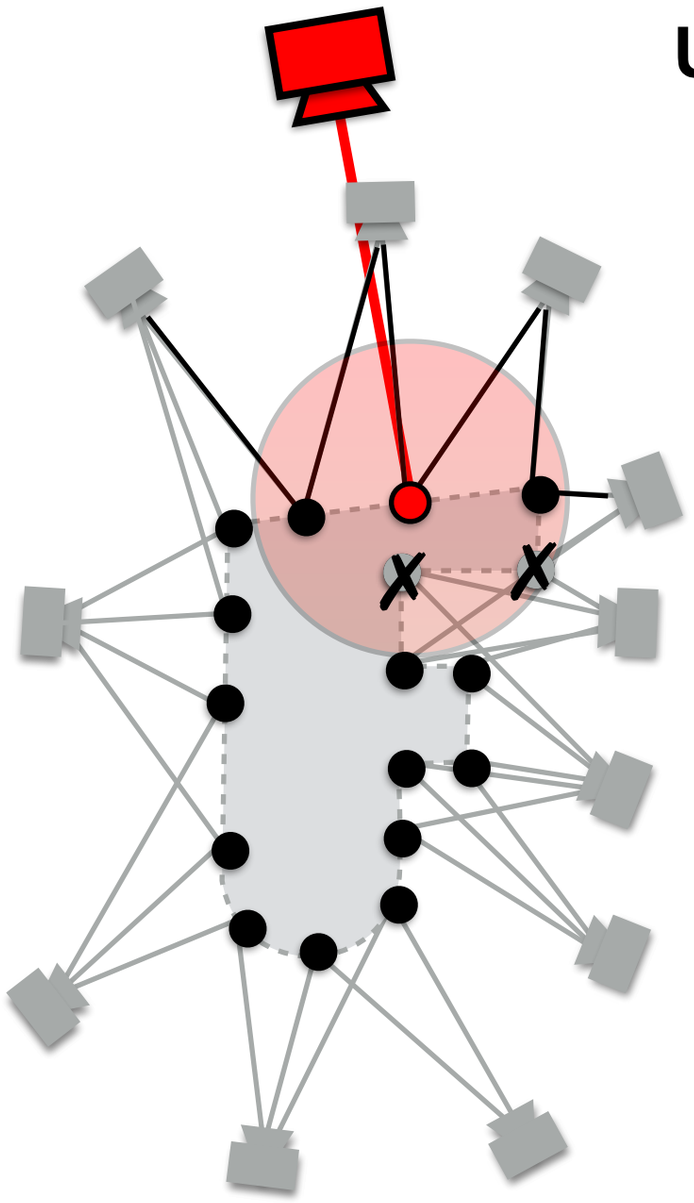
Use Visibility Graph to …

… filter out 3D-to-2D matching candidates

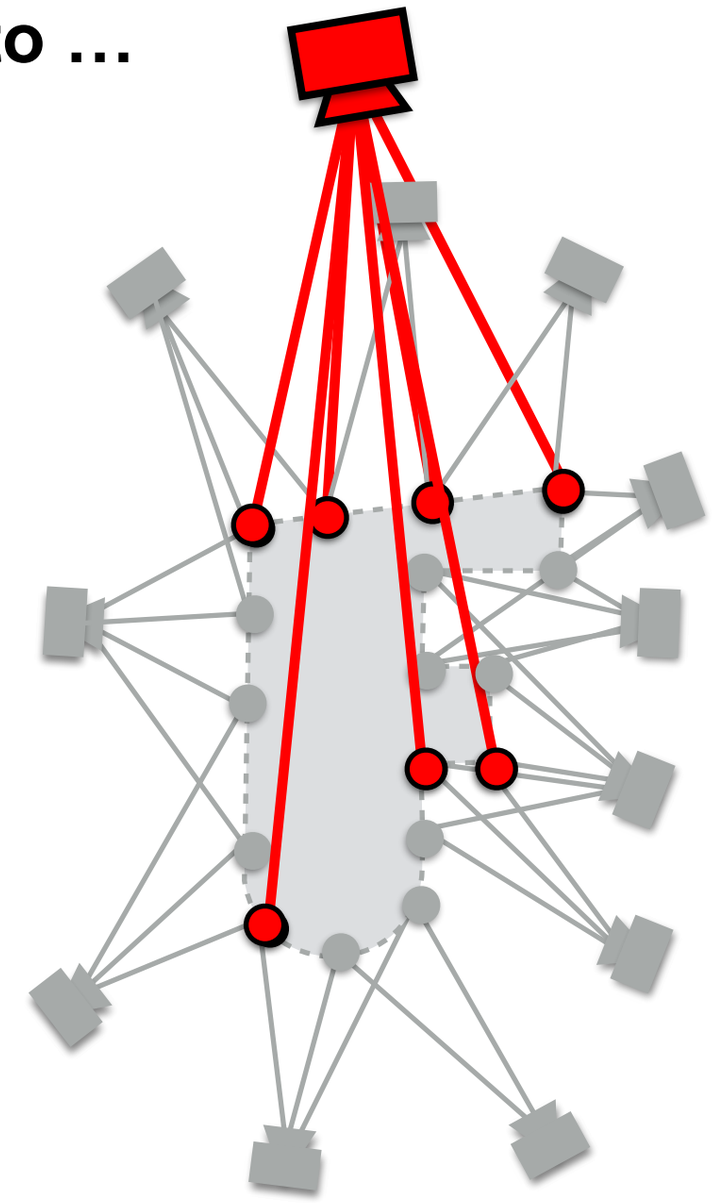
[Sattler et al., ECCV'12]

Visibility Filtering

Use Visibility Graph to …



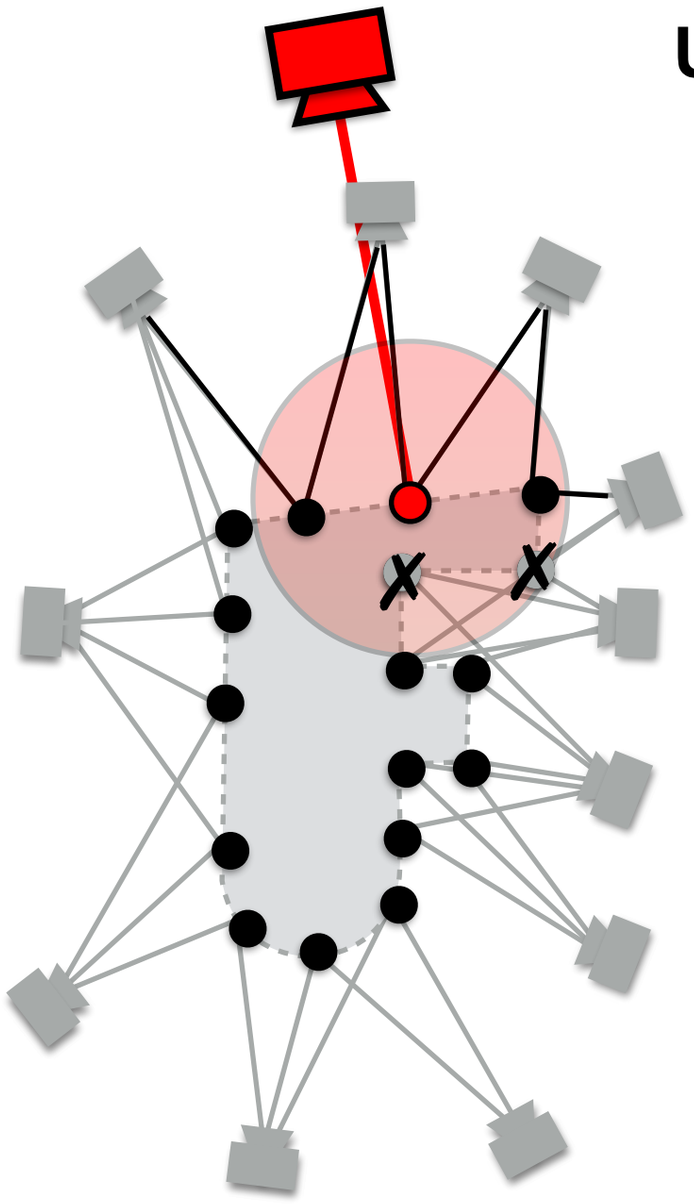
… filter out 3D-to-2D matching candidates



… remove wrong matches before RANSAC

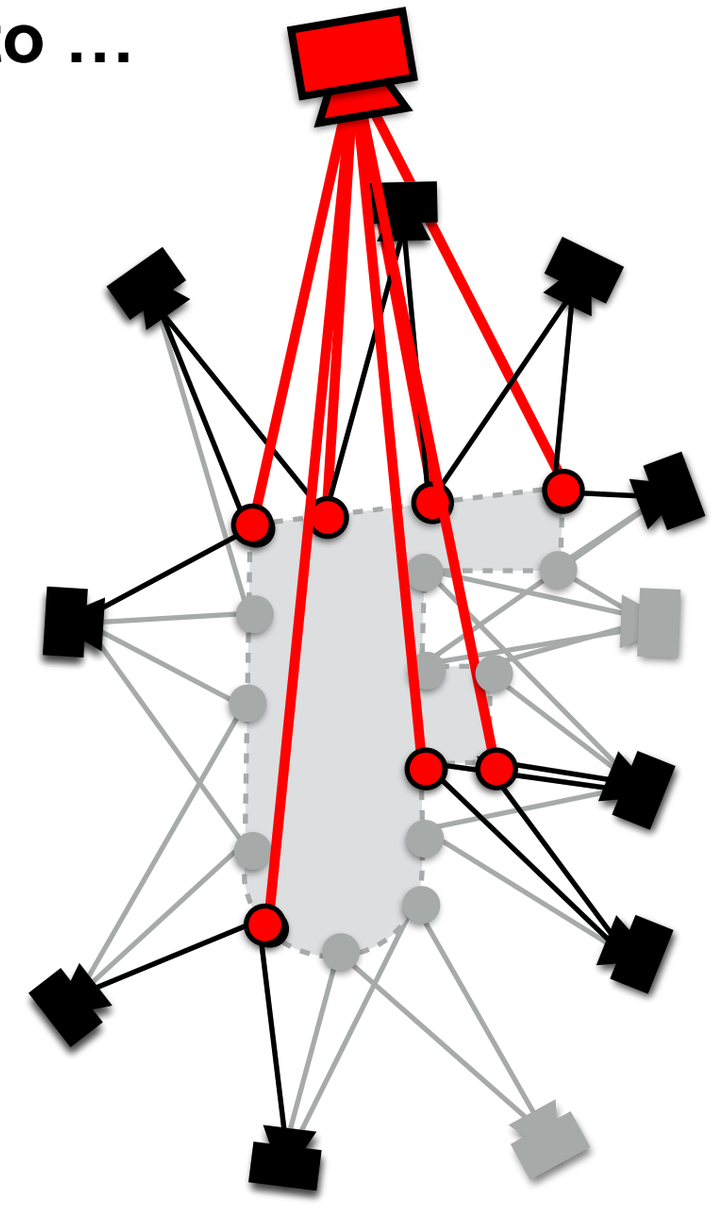
[Sattler et al., ECCV'12]

Visibility Filtering



… filter out 3D-to-2D matching candidates

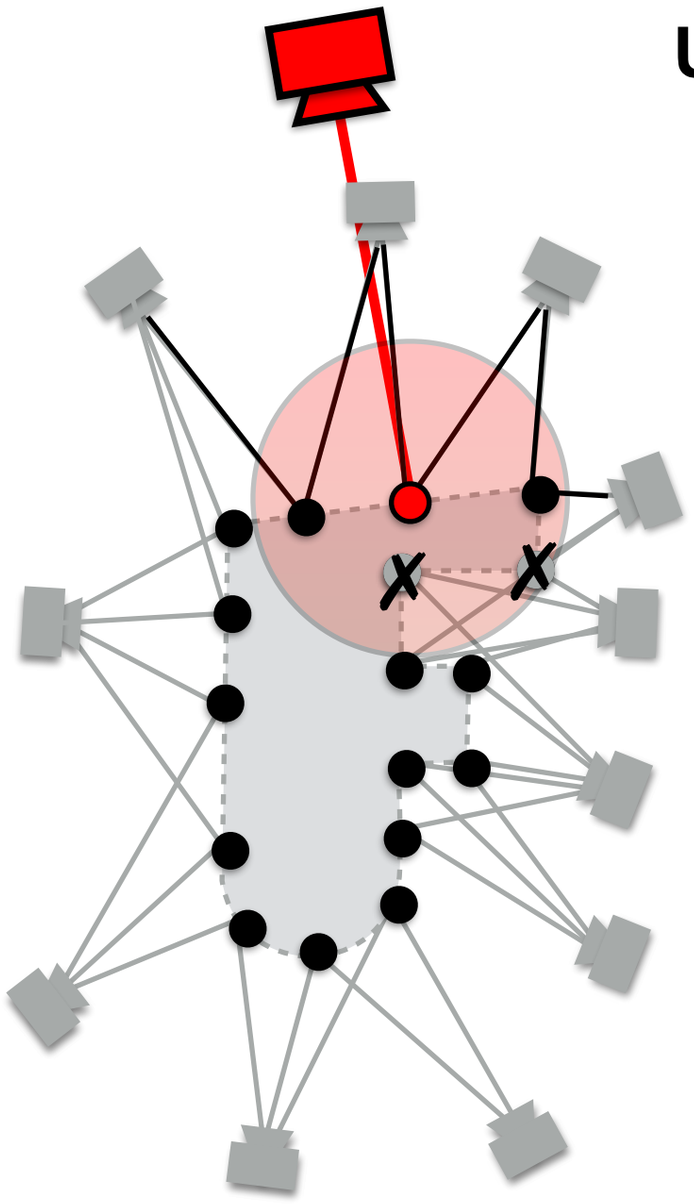
Use Visibility Graph to …



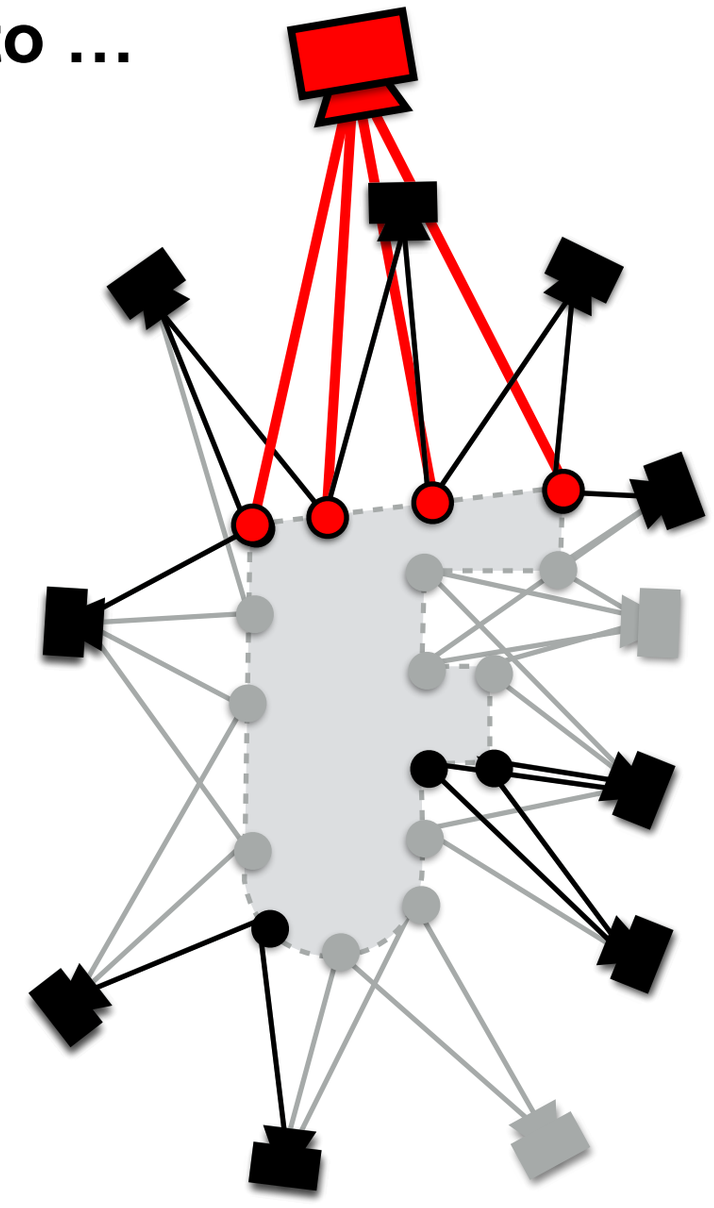
… remove wrong matches before RANSAC

[Sattler et al., ECCV'12]

Visibility Filtering



Use Visibility Graph to …

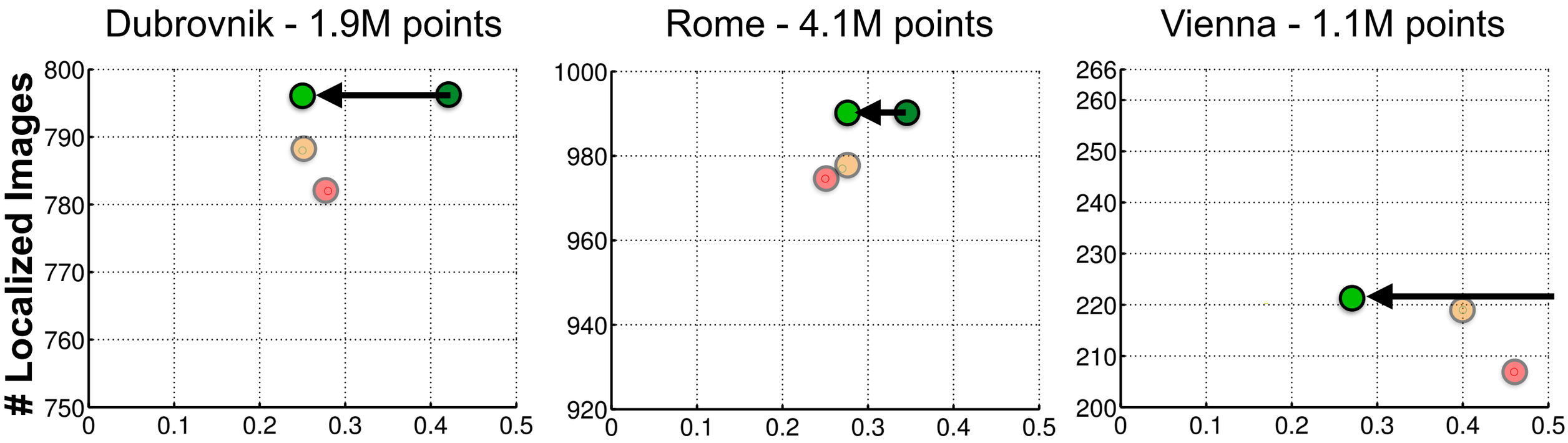


… filter out 3D-to-2D matching candidates

… remove wrong matches before RANSAC

[Sattler et al., ECCV'12]

Results

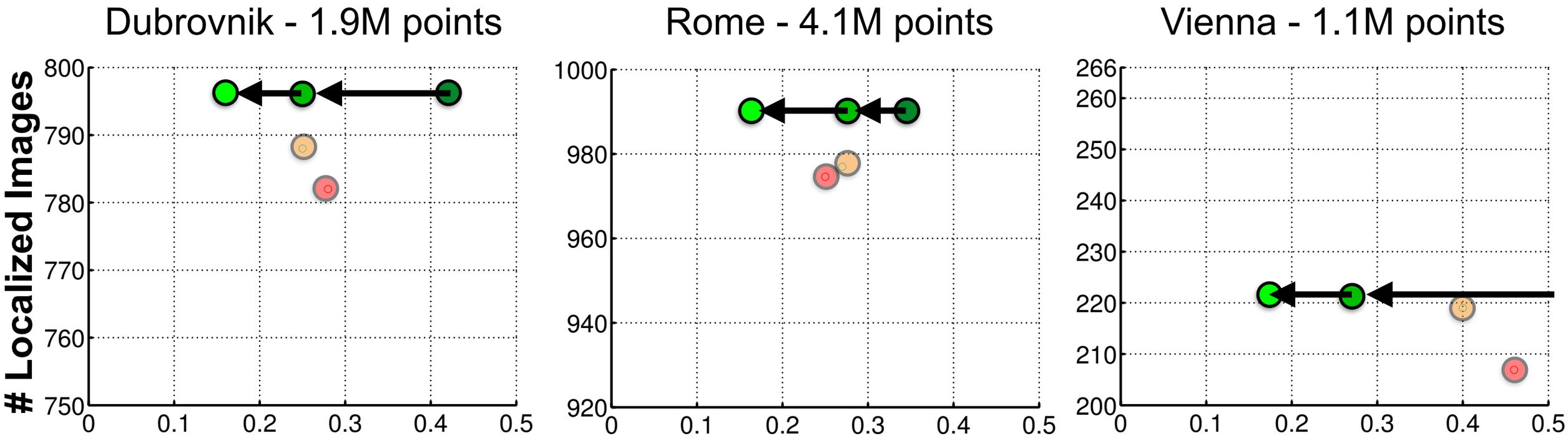


Mean localization time per image [s] (excluding feature extraction)

- **Active Search + Filtering**
- Active Search
- kd-tree
- P2F - compact model
- VPS
- PGPM

code will be available “soon”

Results

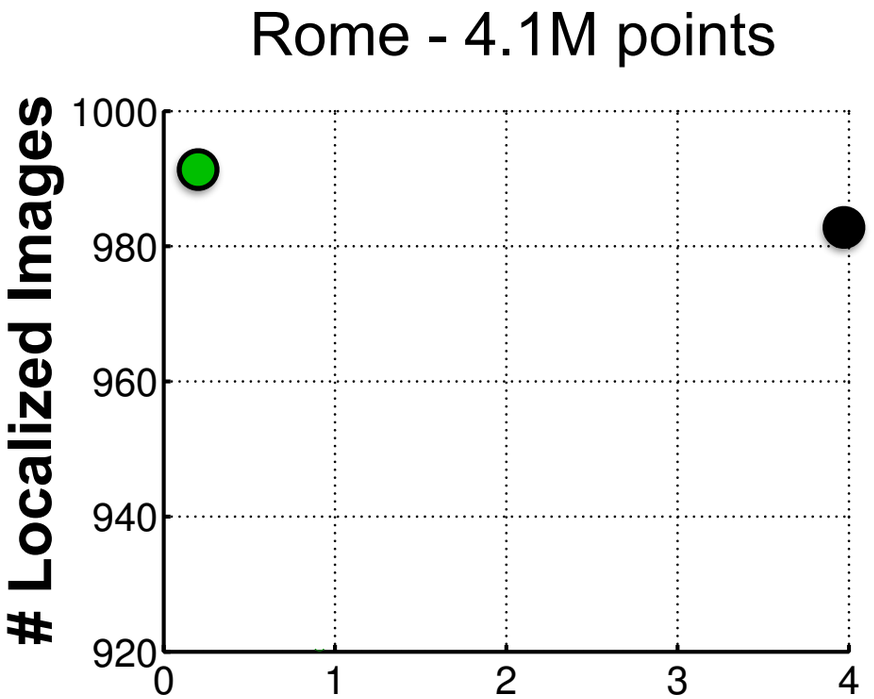


Mean localization time per image [s] (excluding feature extraction)

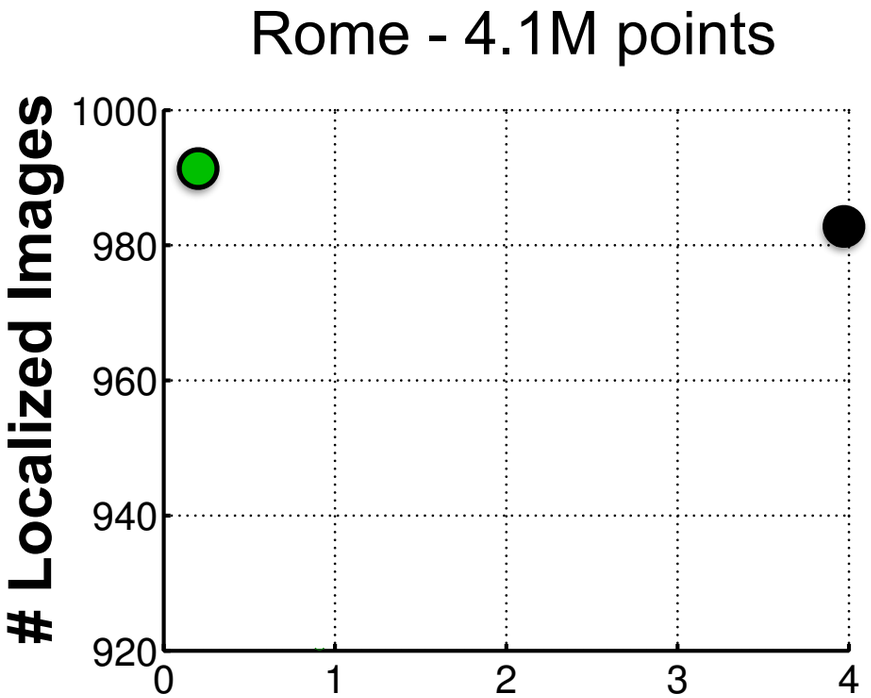
- **Active Search + Filtering + Cache Optimization** [Sattler, Thesis'14]
- **Active Search + Filtering**
- Active Search
- kd-tree
- P2F - compact model
- VPS
- PGPM

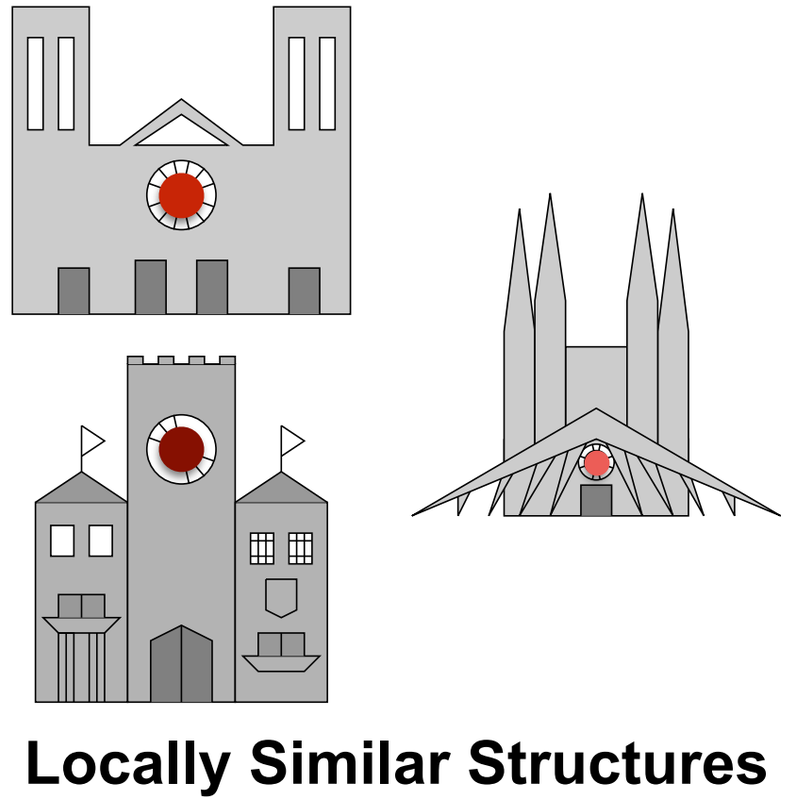
code will be available "soon"

Active Search vs. Pure 2D-to-3D Matching



Active Search vs. Pure 2D-to-3D Matching

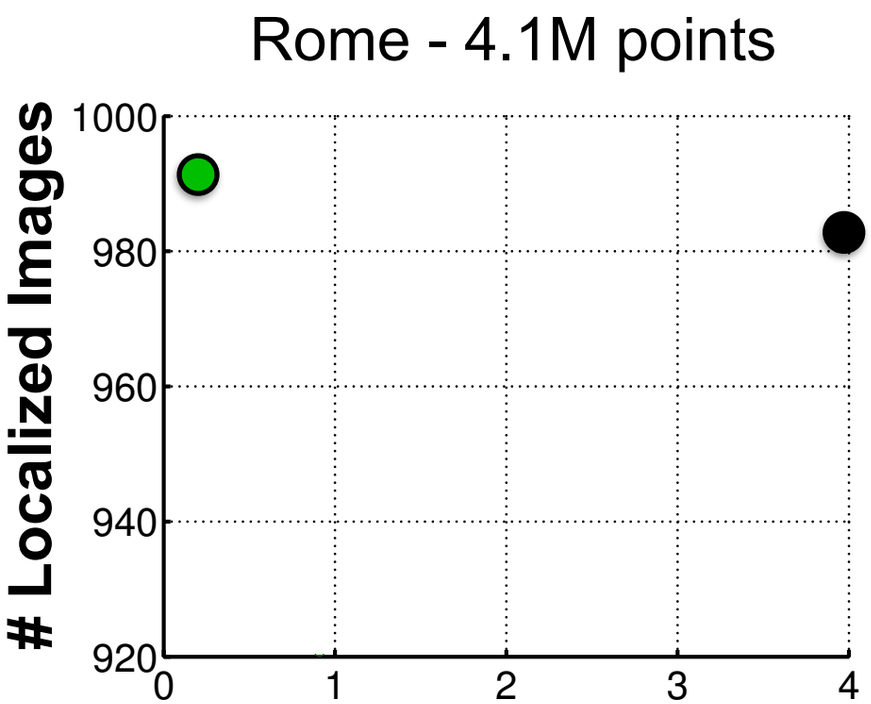




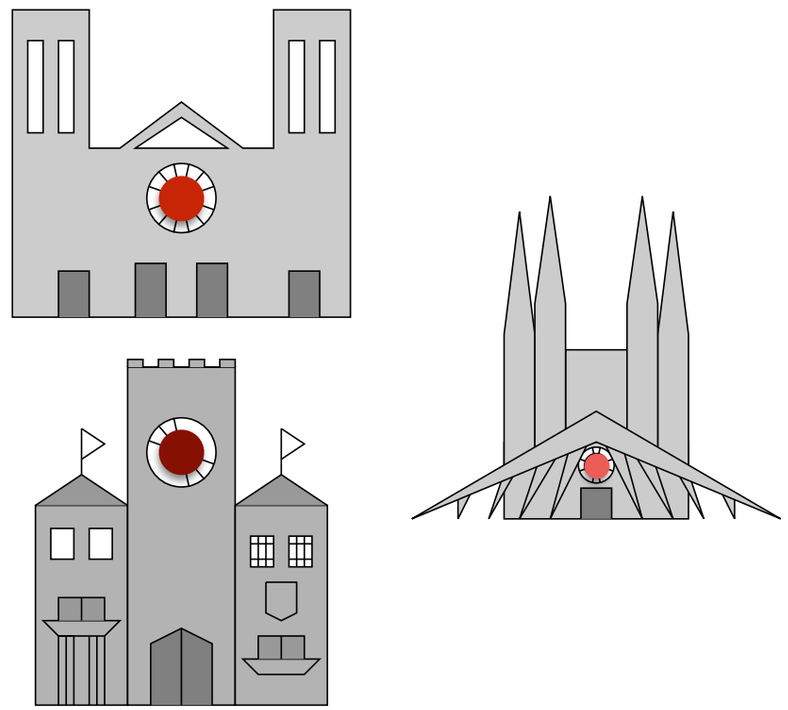
Locally Similar Structures

- Larger models = higher chance to have (nearly) identical local structures

Active Search vs. Pure 2D-to-3D Matching



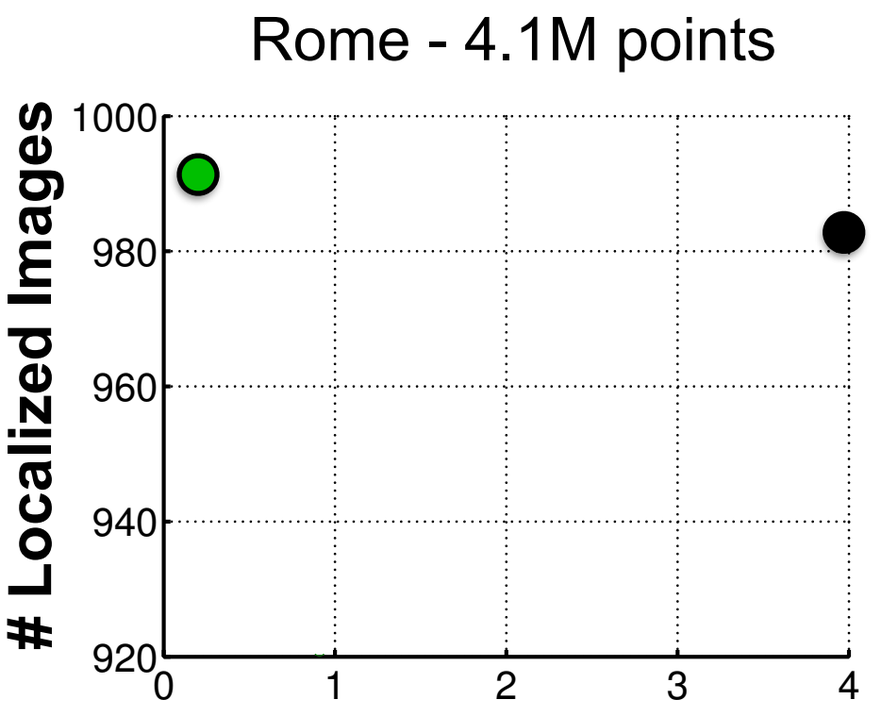


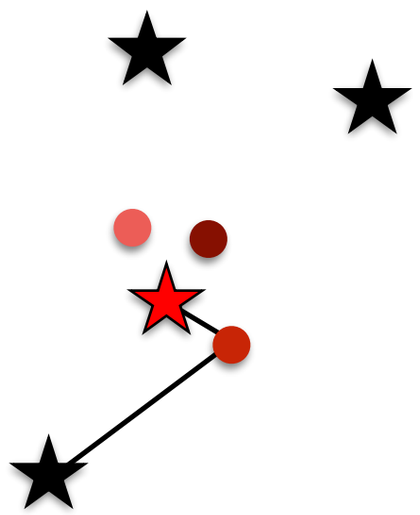


Descriptor Space **Locally Similar Structures**

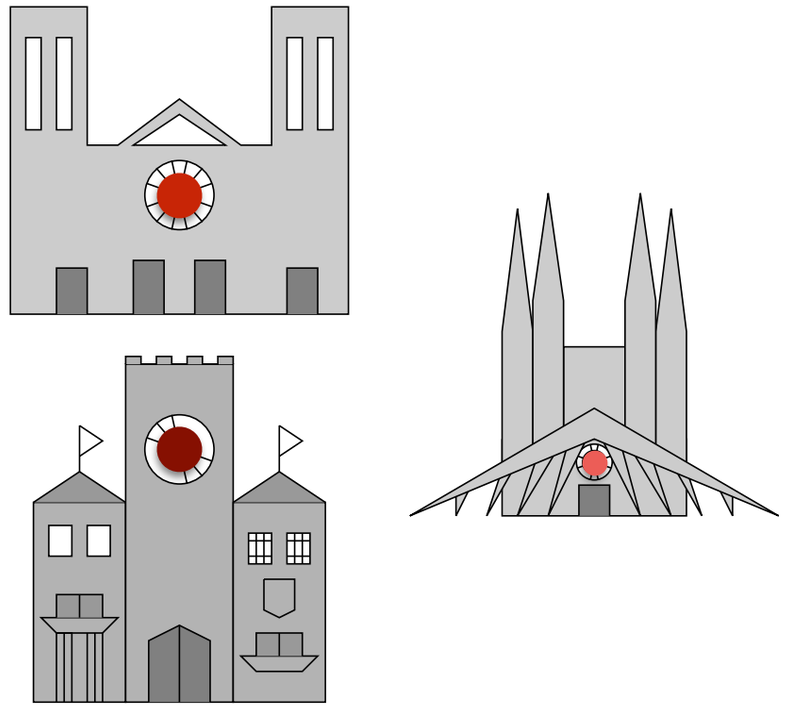
- Larger models = higher chance to have (nearly) identical local structures
- Ratio test rejects more and more correct 2D-to-3D matches

Active Search vs. Pure 2D-to-3D Matching





Descriptor Space



Locally Similar Structures

- Larger models = higher chance to have (nearly) identical local structures
- Ratio test rejects more and more correct 2D-to-3D matches
- Active Search recovers rejected matches using 3D-to-2D matching

Active Search vs. Pure 2D-to-3D Matching

Method	% Localized Images	Mean Localization Time [s]
kd-tree [Li et al., ECCV'12]	~87	“few seconds”
VPS (opt)	85.47	0.89
Active Search (opt)	**95.34**	**0.48**

Landmarks 1k dataset [Li et al., ECCV'12]

- Most popular 1k landmarks from Flickr
- 38M points reconstructed from 204k images
- 10k query images

- Active Search significantly better than kd-tree approach!

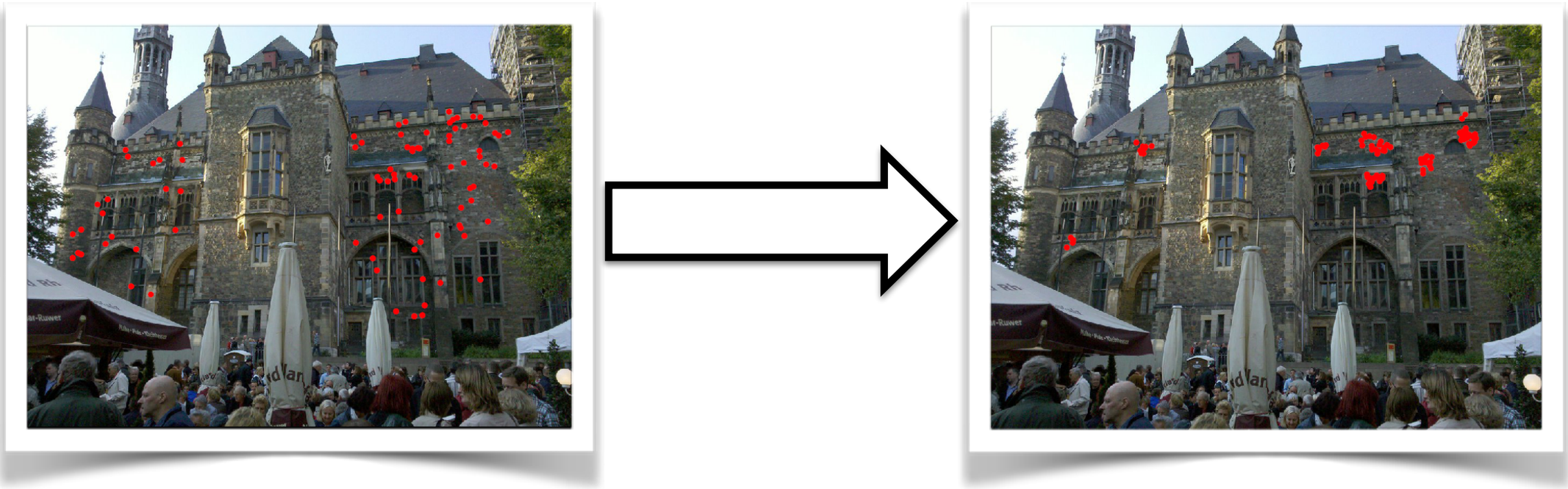
Results - Localization Accuracy

Method	Total # Images	Mean [m]	Median [m]	Quantiles [m]		
				25 %	75 %	90 %
P2F - compact model	753	**18.3**	9.3	7.5	13.4	
VPS*	783	21.8	2.1	0.8	6.8	
PGPM	788	34.8	3.1	0.88	11.83	
Active Search *	**797**	30.7	**1.3**	**0.5**	**5.0**	**19.2**

Measured on Dubrovnik dataset, all methods use P6P

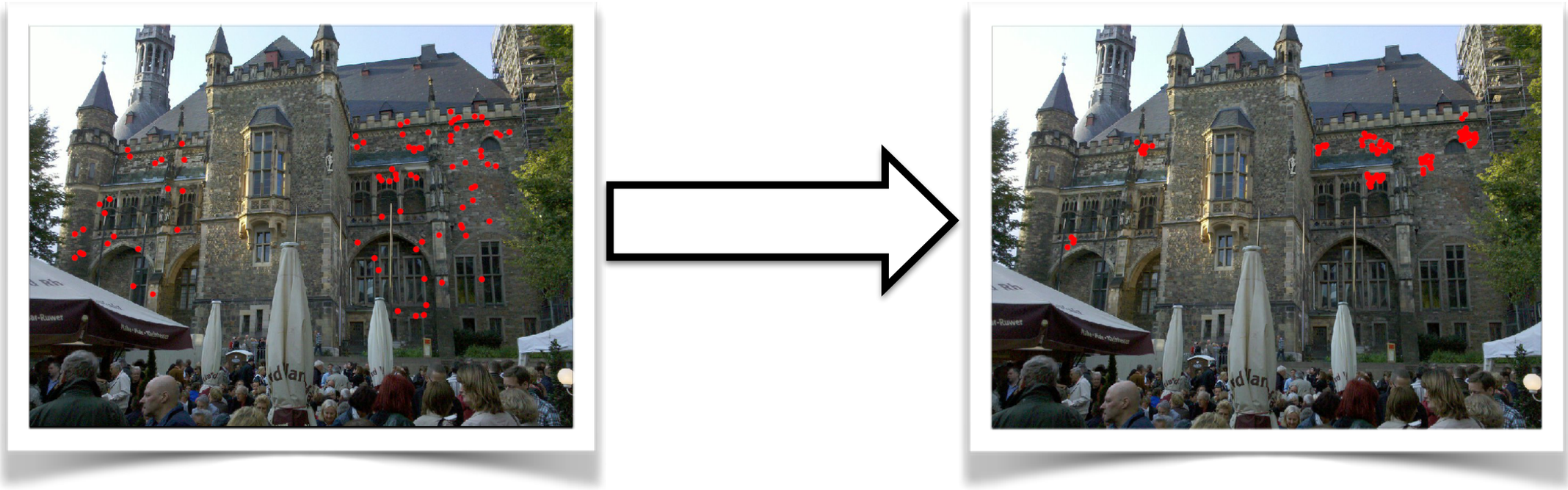
* Distribution over median localization error per camera [Sattler, Thesis'14]

Active Search - Localization Accuracy



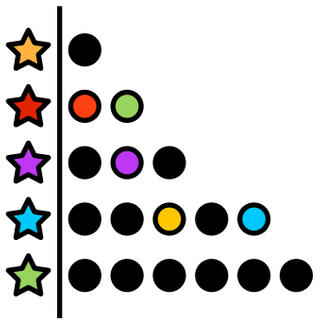
- For larger models, 3D-to-2D matching always preferred
  - ➡ Matches are found in small image regions
  - ➡ Unstable configurations for pose estimation = worse accuracy

Active Search - Localization Accuracy



- For larger models, 3D-to-2D matching always preferred
  - ➡ Matches are found in small image regions
  - ➡ Unstable configurations for pose estimation = worse accuracy
- “Easy” fix: Use compact model obtained by solving Set Cover problem to enforce larger point distances [Sattler, Thesis’14]

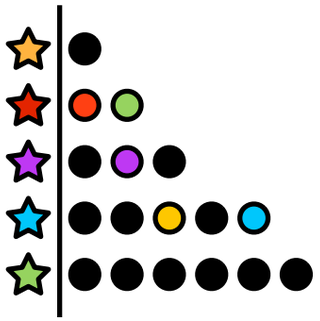
Key Insights



Prioritized Search

- 2D-to-3D matching more reliable than 3D-to-2D search
- Efficient search through **prioritization**
- Effectiveness reduced by **quantization**

Key Insights



Prioritized Search

- 2D-to-3D matching more reliable than 3D-to-2D search
- Efficient search through **prioritization**
- Effectiveness reduced by **quantization**

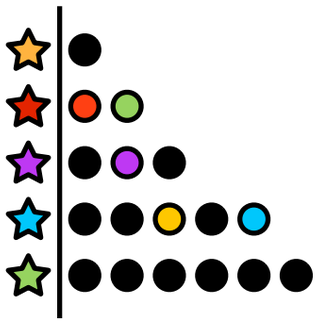


Active Search

- **Recover** missing matches via 3D-to-2D search

➡**State-of-the-art localization effectiveness**

Key Insights



Prioritized Search

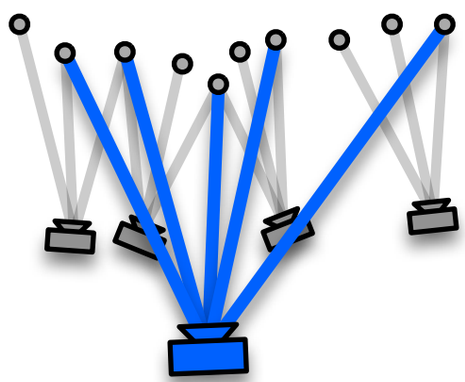
- 2D-to-3D matching more reliable than 3D-to-2D search
- Efficient search through **prioritization**
- Effectiveness reduced by **quantization**



Active Search

- **Recover** missing matches via 3D-to-2D search

➡**State-of-the-art localization effectiveness**



Visibility Filtering

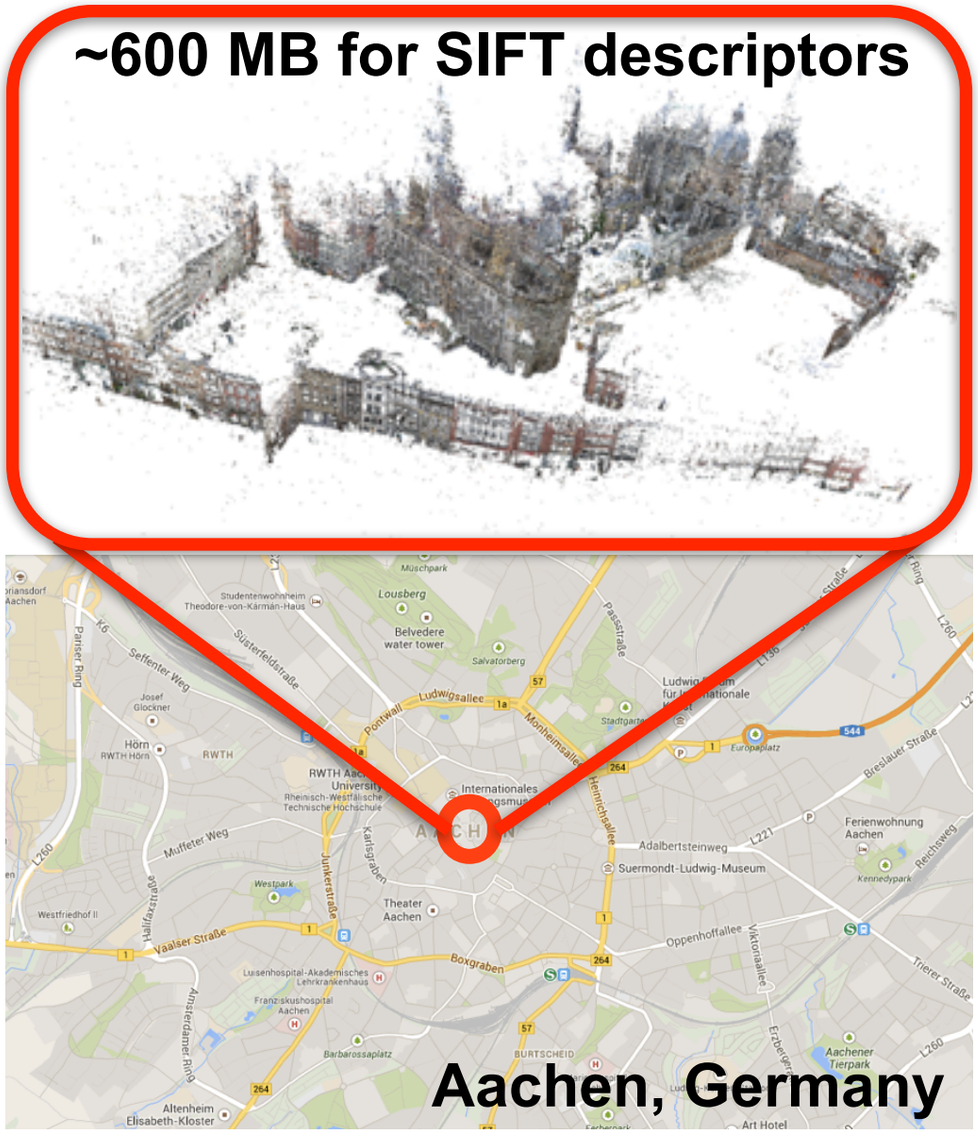
- Accelerate both 3D-to-2D matching & pose estimation

➡**State-of-the-art localization efficiency & effectiveness**

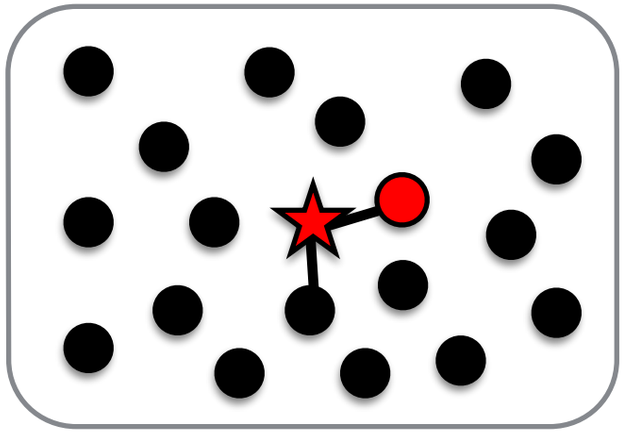
Open Questions

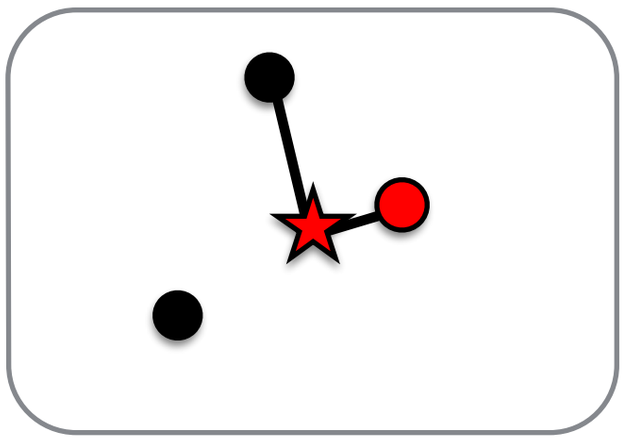
- So far only used a single image with no additional information
- Unrealistic, we usually know much more:
  - GPS & IMU data
  - Video stream
  - Semantic information
- How to best exploit these additional information?

Scalability Problems

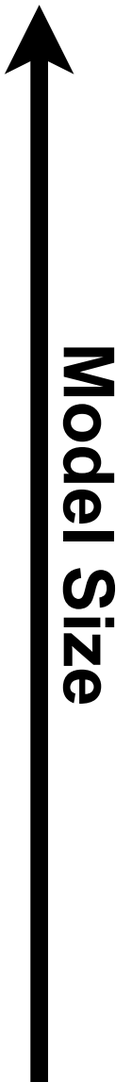


Memory Consumption





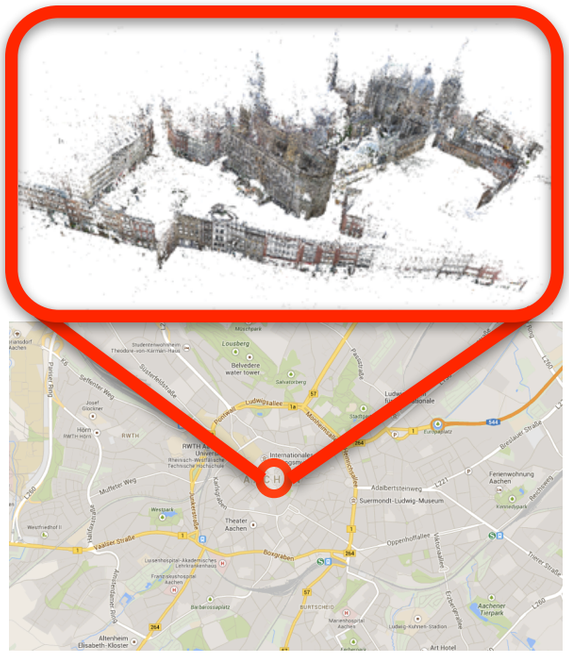
Ratio Test

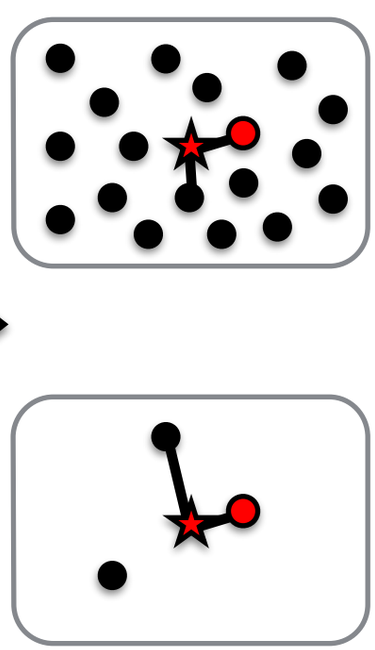


Overview

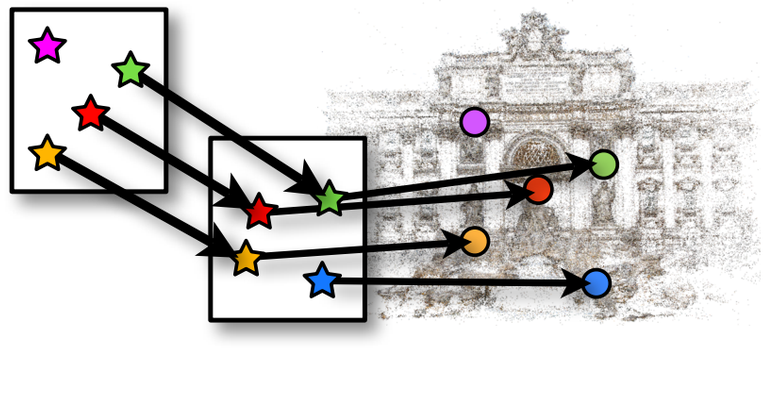
- Basics
- Localization using Prioritized Matching
- Scalable Image-Based Localization
- Open Challenges / Conclusion

Scalable Image-Based Localization







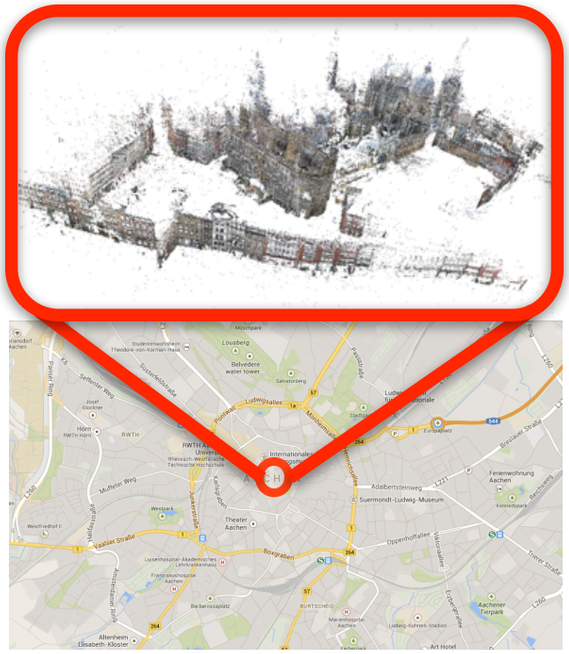


Memory Consumption Problem

Ratio Test Problem

Place Recognition for Localization

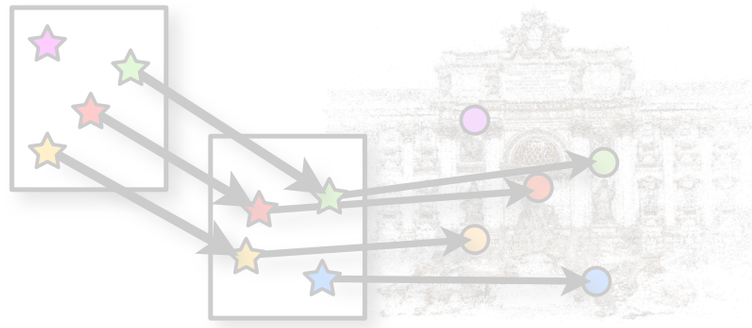
Scalable Image-Based Localization



Memory Consumption Problem



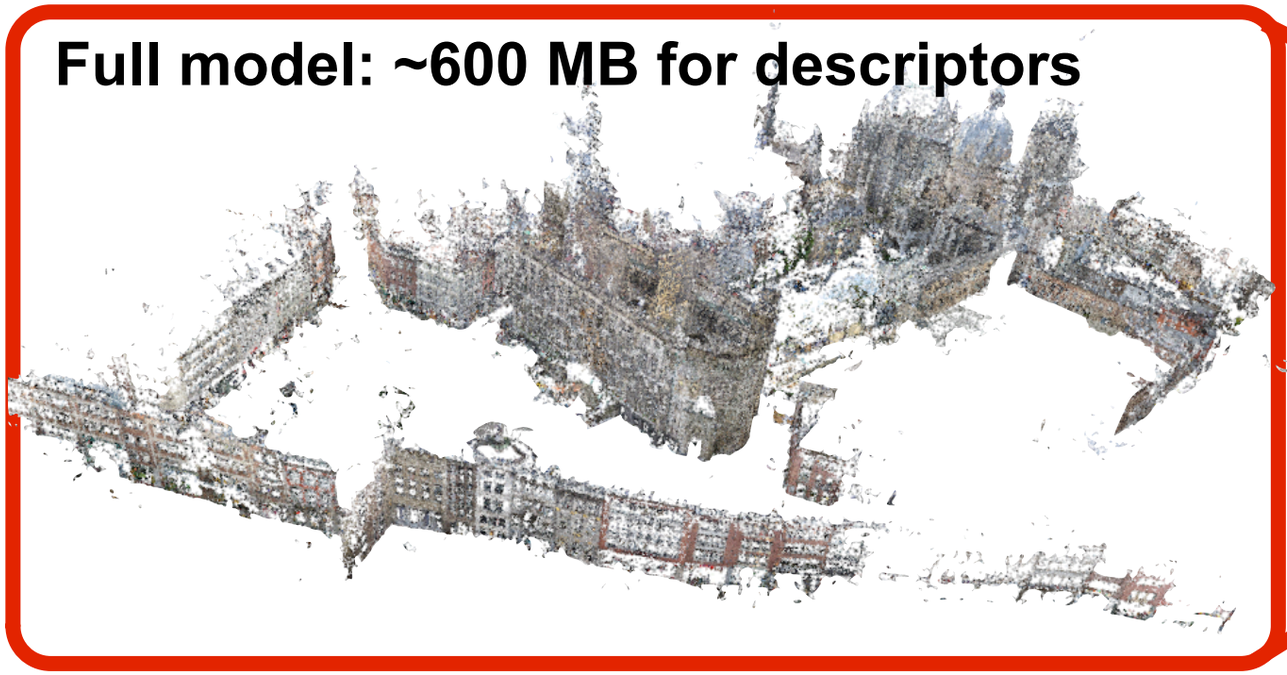
Ratio Test Problem

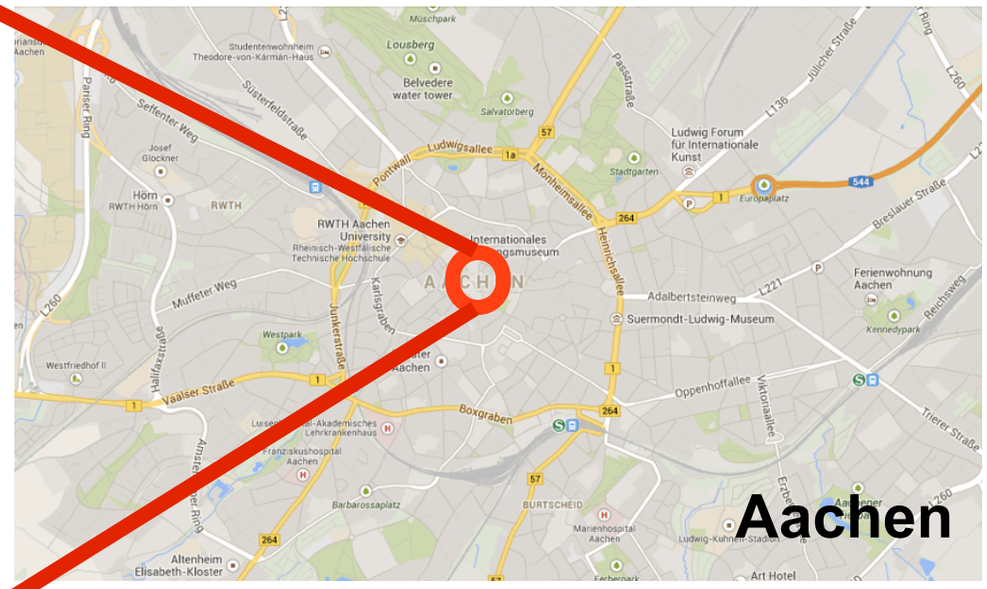


Place Recognition for Localization

The Memory Consumption Problem

Full model: ~600 MB for descriptors

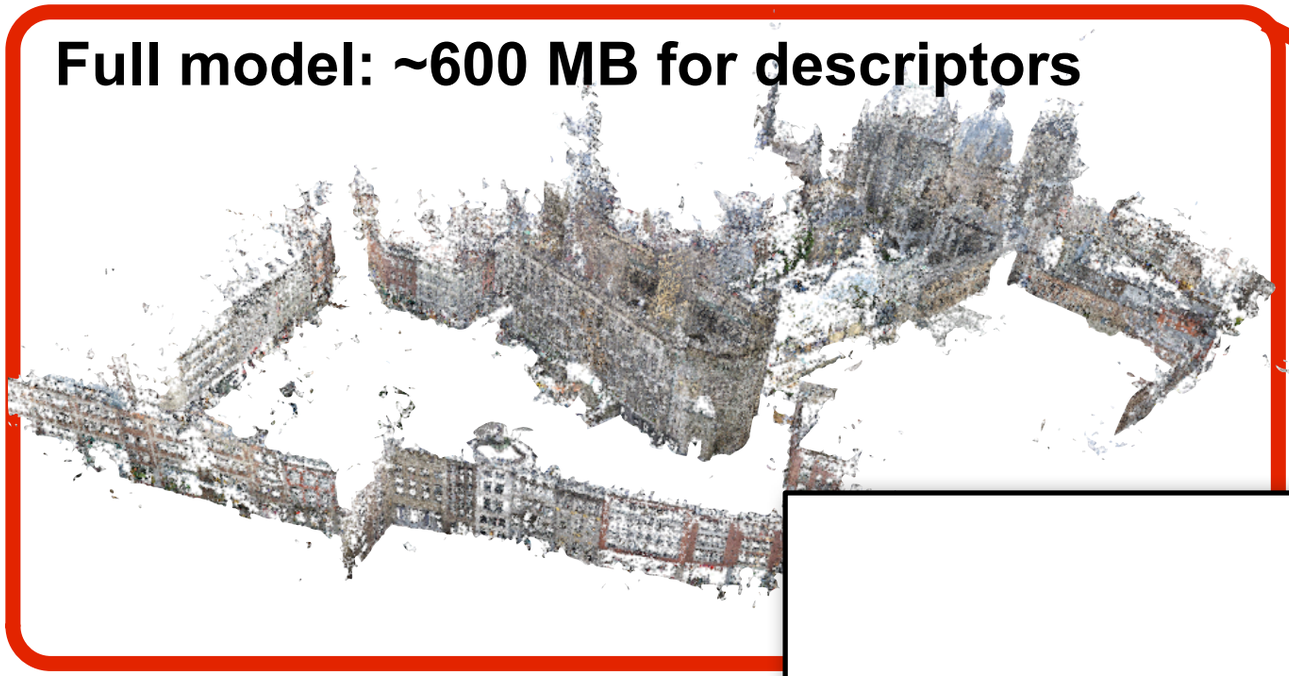


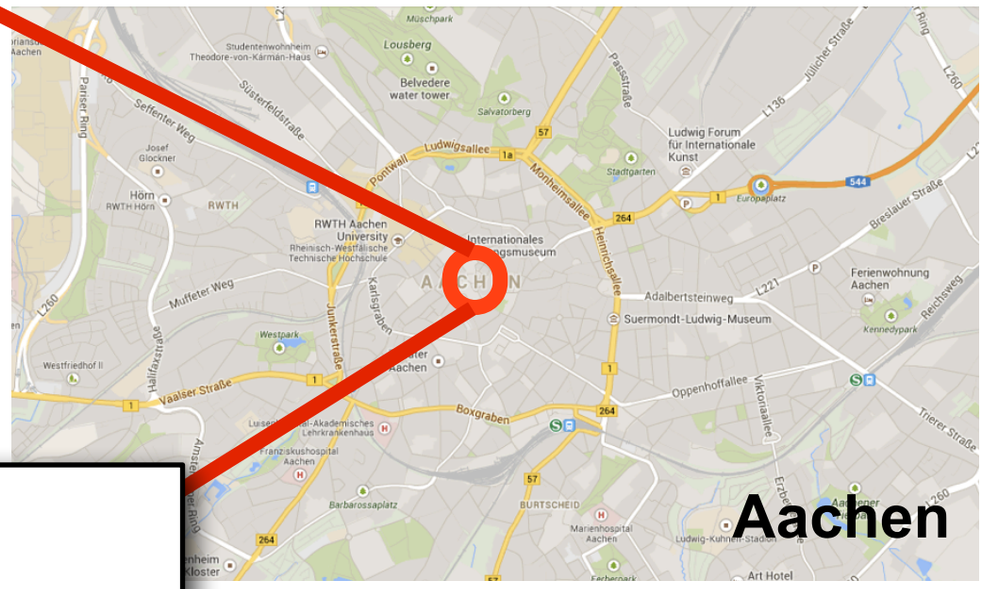


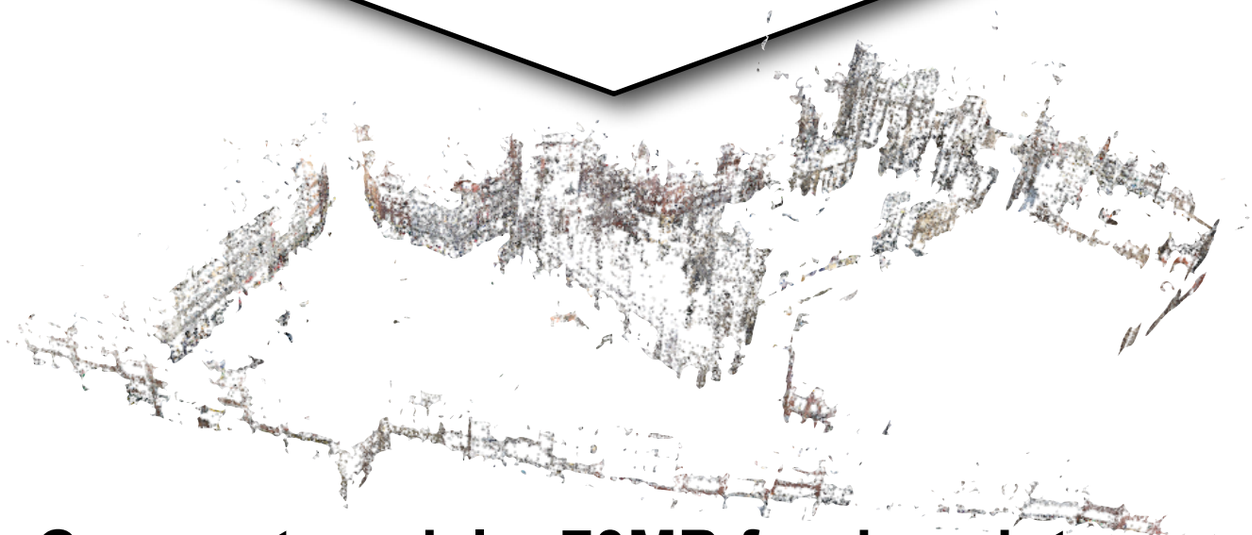
[Sattler et al.,LNCS'12]

The Memory Consumption Problem

Full model: ~600 MB for descriptors



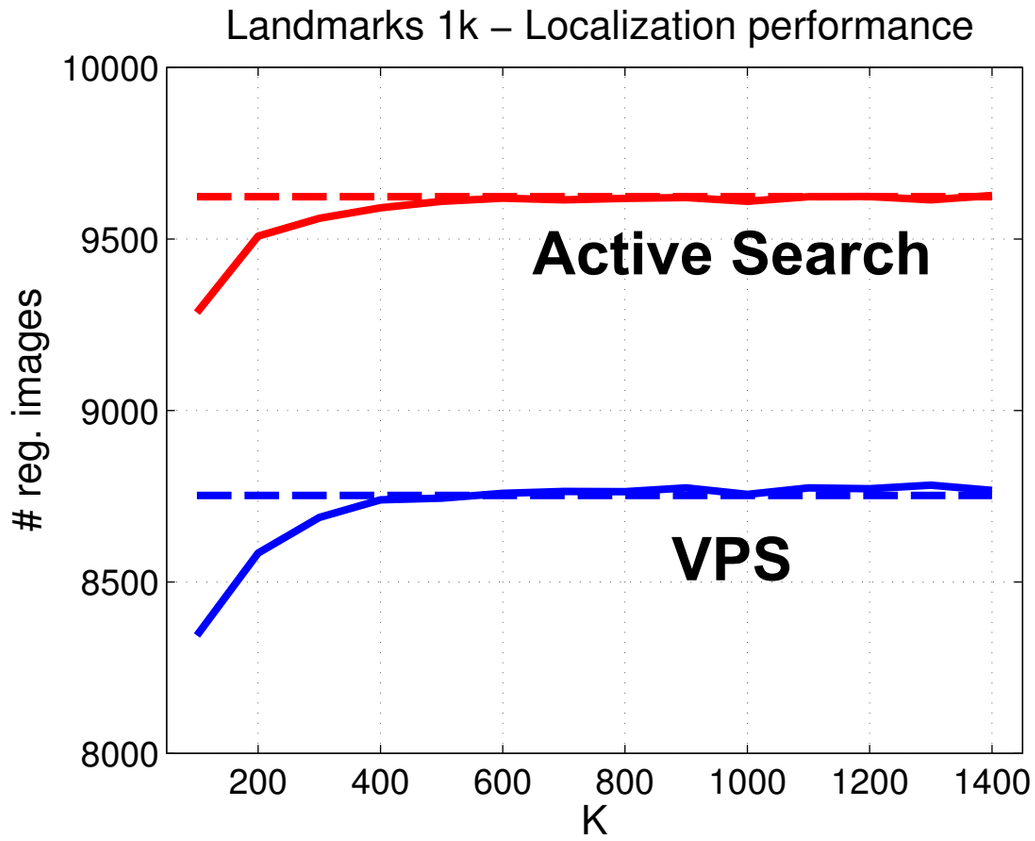




Compact model: ~70MB for descriptors

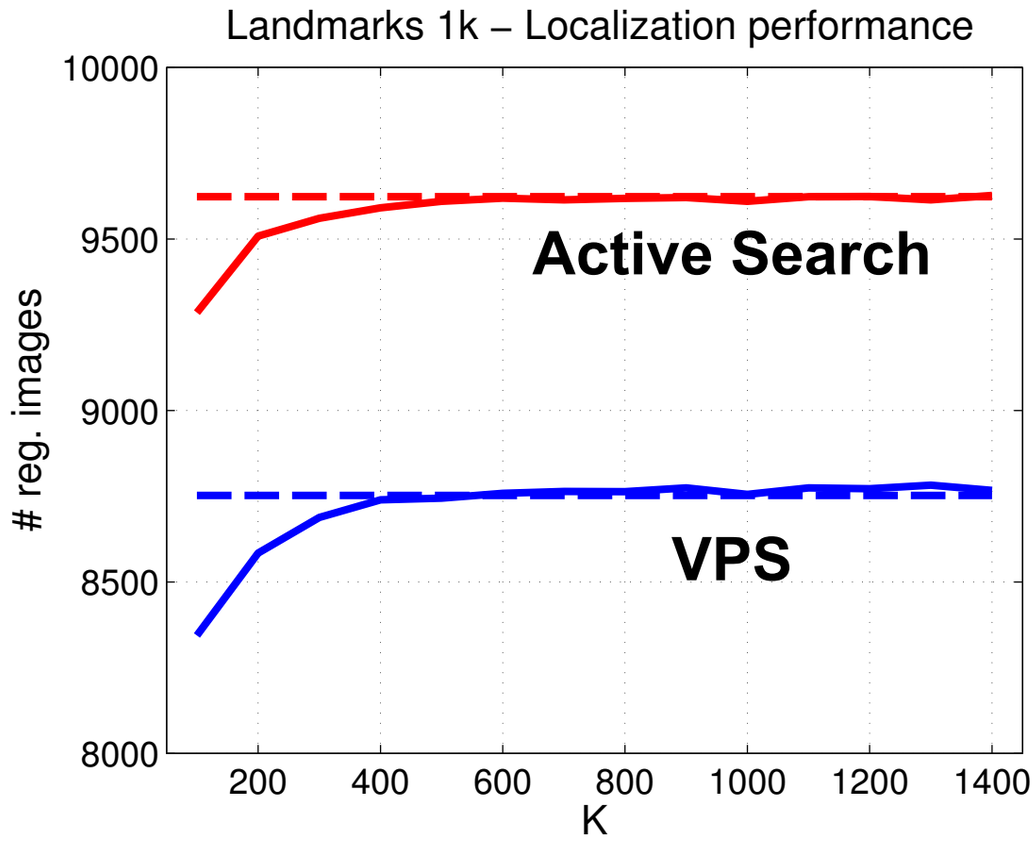
[Sattler et al.,LNCS'12]

Compact Models for Prioritized Search



[Sattler et al.,LNCS'12] [Sattler, Thesis'14]

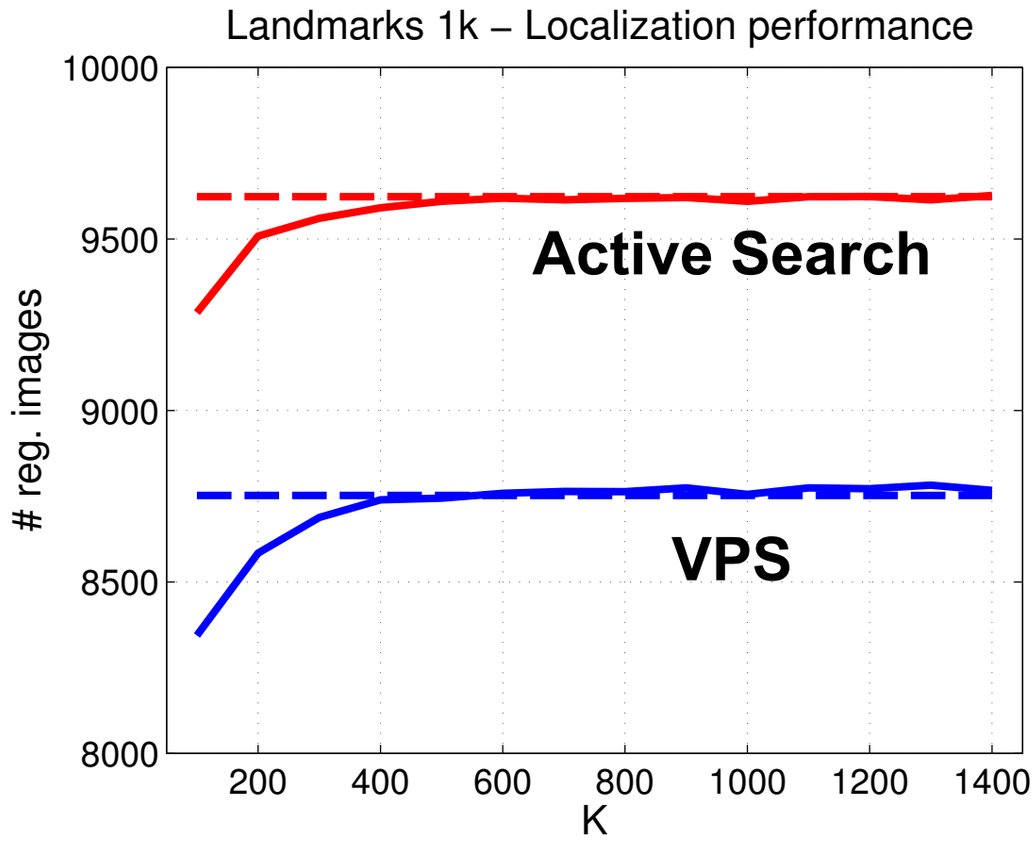
Compact Models for Prioritized Search



- Same effectiveness at faster search times for 30-40% of points

[Sattler et al.,LNCS'12] [Sattler, Thesis'14]

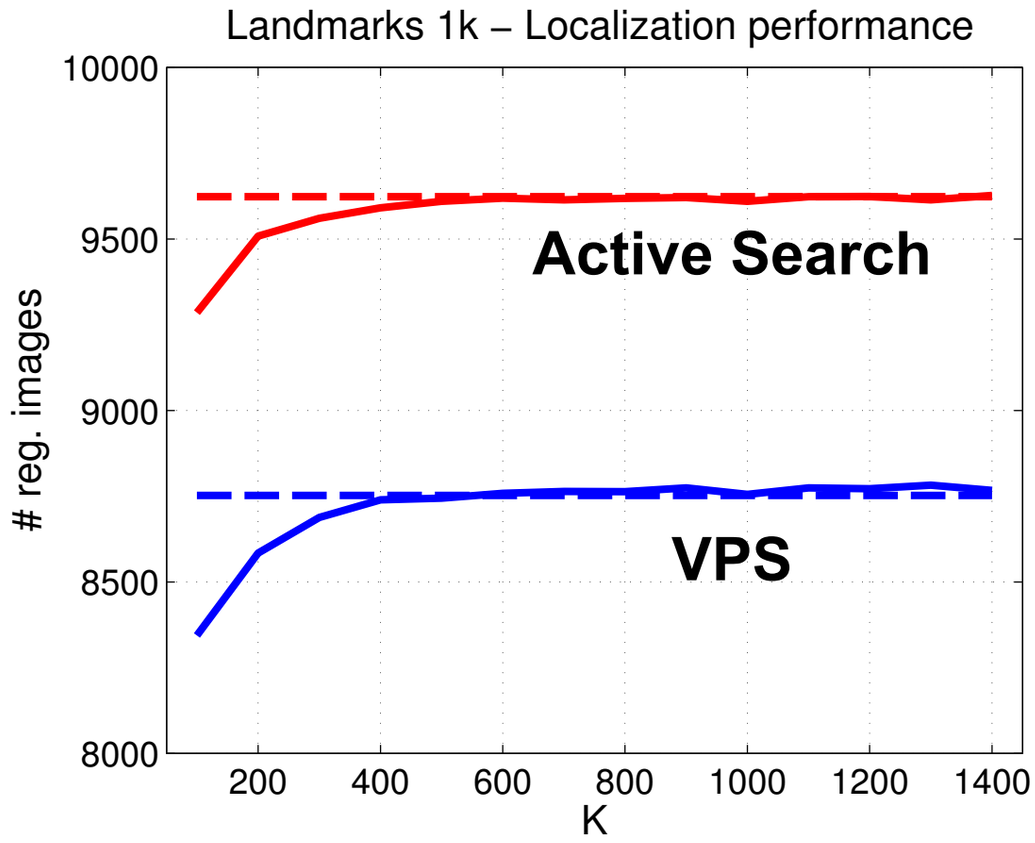
Compact Models for Prioritized Search



- Same effectiveness at faster search times for 30-40% of points
- Not enough to better scale to larger datasets

[Sattler et al.,LNCS'12] [Sattler, Thesis'14]

Compact Models for Prioritized Search



- Same effectiveness at faster search times for 30-40% of points
- Not enough to better scale to larger datasets
- Better compression scheme requires only ~3% [Cao & Snavely, CVPR'14] (Posters P1)

[Sattler et al.,LNCS'12] [Sattler, Thesis'14]

Using Image Retrieval?

- Memory increase mainly caused by descriptors
  - Needed to establish high-quality matches, avoid exponential increase in RANSAC's run-time

Using Image Retrieval?

- Memory increase mainly caused by descriptors
  - Needed to establish high-quality matches, avoid exponential increase in RANSAC's run-time
- Why not use place recognition / image retrieval as first step?

Using Image Retrieval?

- Memory increase mainly caused by descriptors
  - Needed to establish high-quality matches, avoid exponential increase in RANSAC's run-time
- Why not use place recognition / image retrieval as first step?

✓No need for full descriptors during retrieval

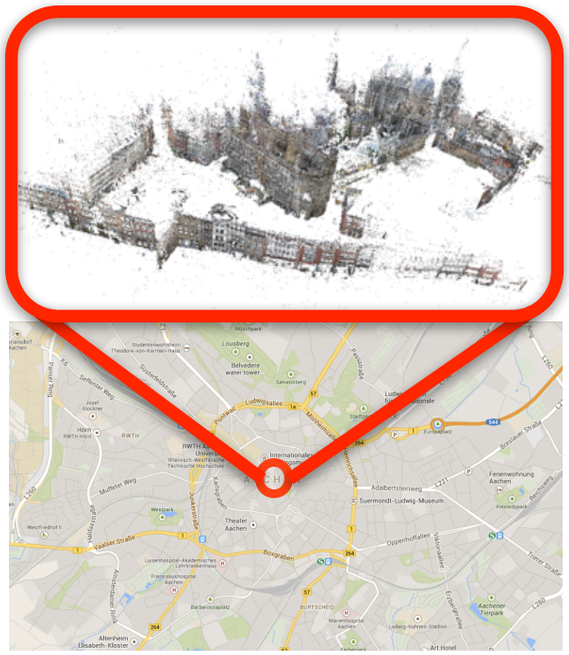
Using Image Retrieval?

- Memory increase mainly caused by descriptors
  - Needed to establish high-quality matches, avoid exponential increase in RANSAC's run-time
- Why not use place recognition / image retrieval as first step?

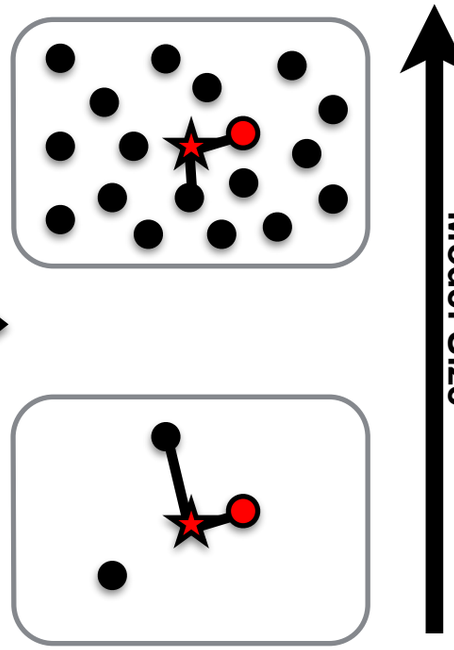
✓No need for full descriptors during retrieval

✓Load only few descriptors to establish 2D-to-2D-to-3D matches required for pose estimation

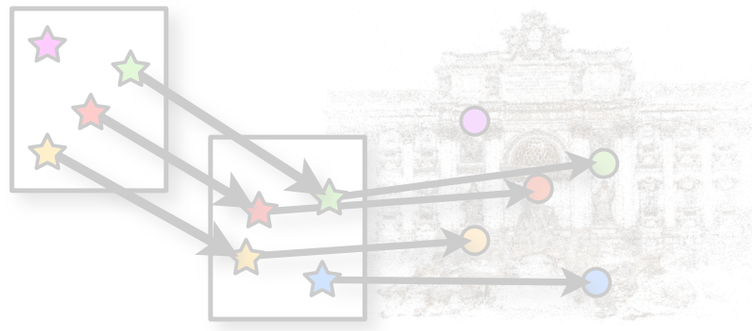
Scalable Image-Based Localization



Memory Consumption Problem

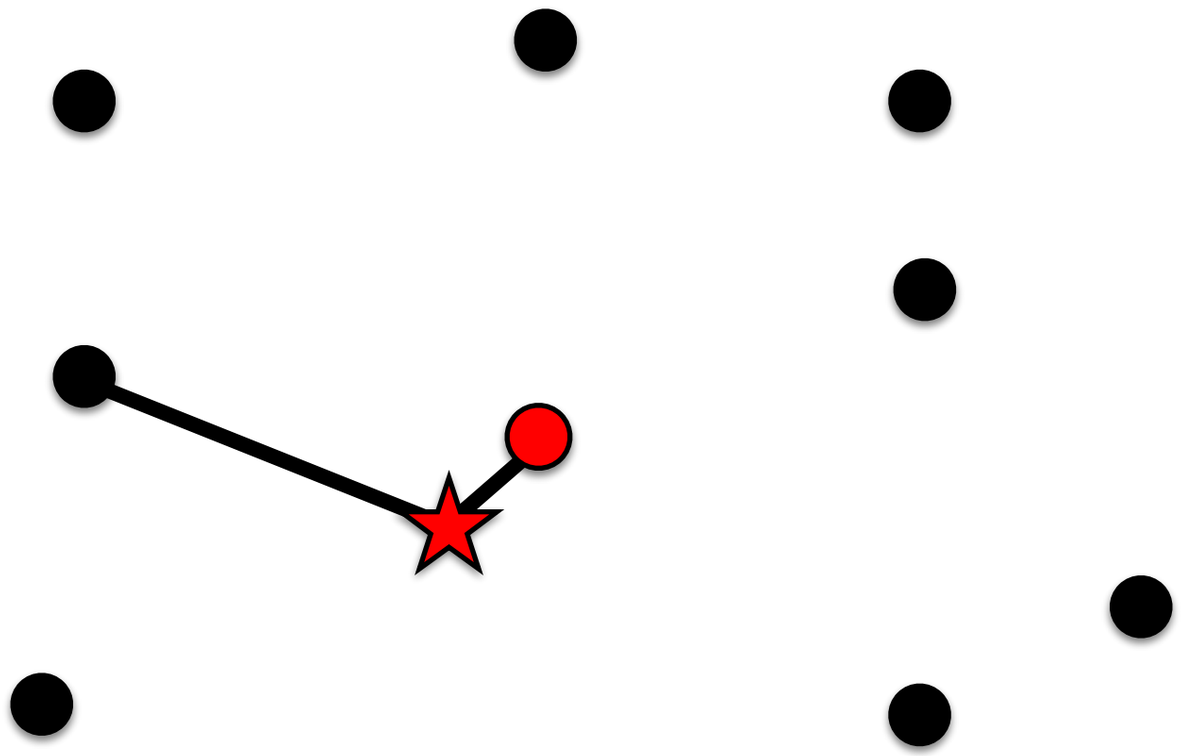


Ratio Test Problem



Place Recognition for Localization

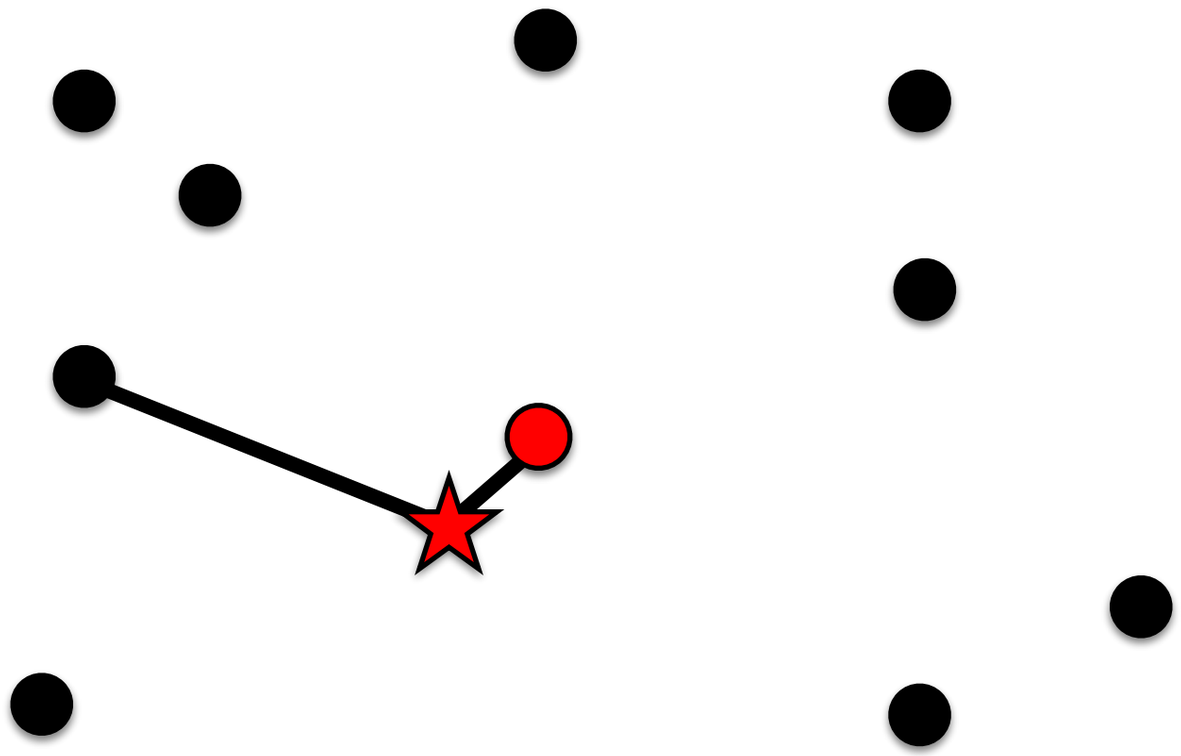
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

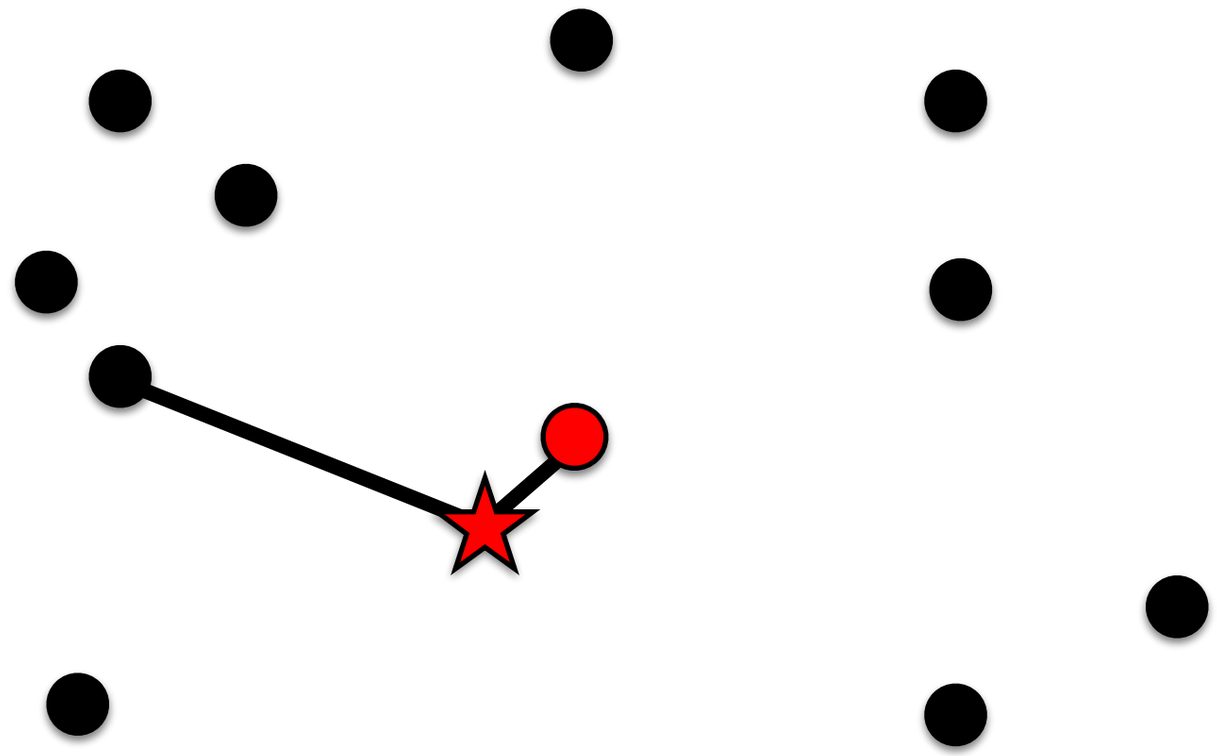
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

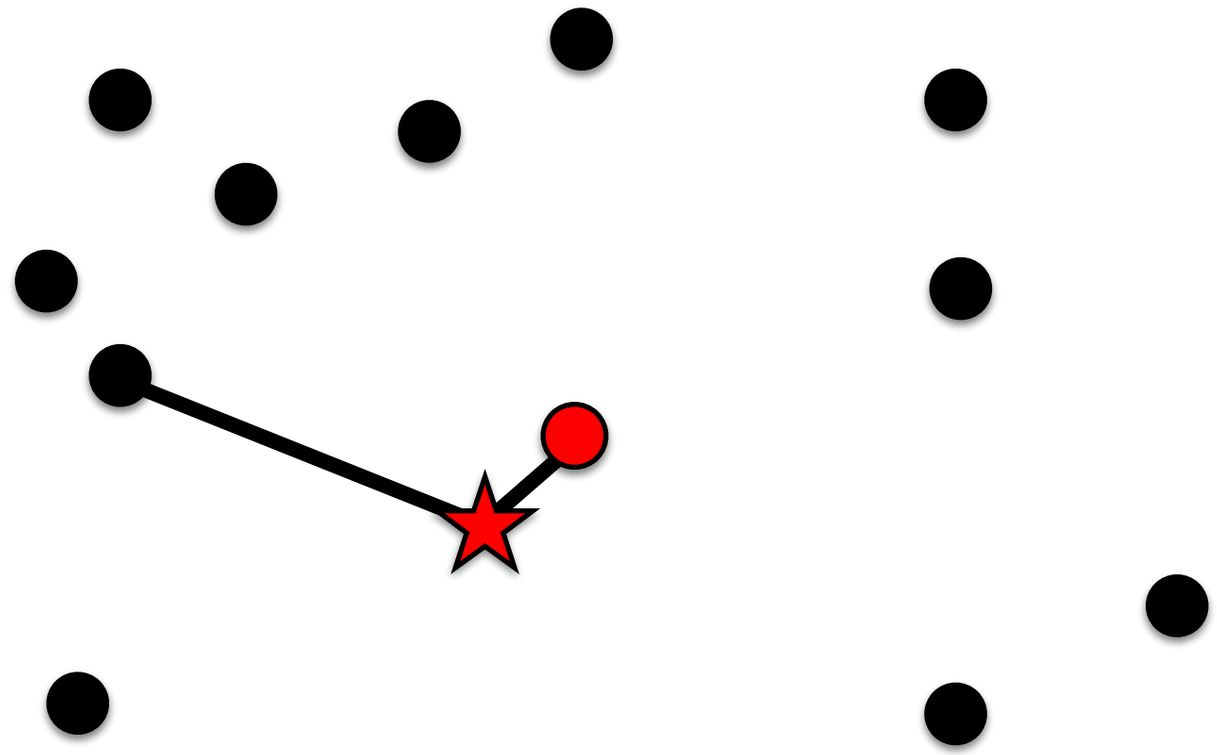
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

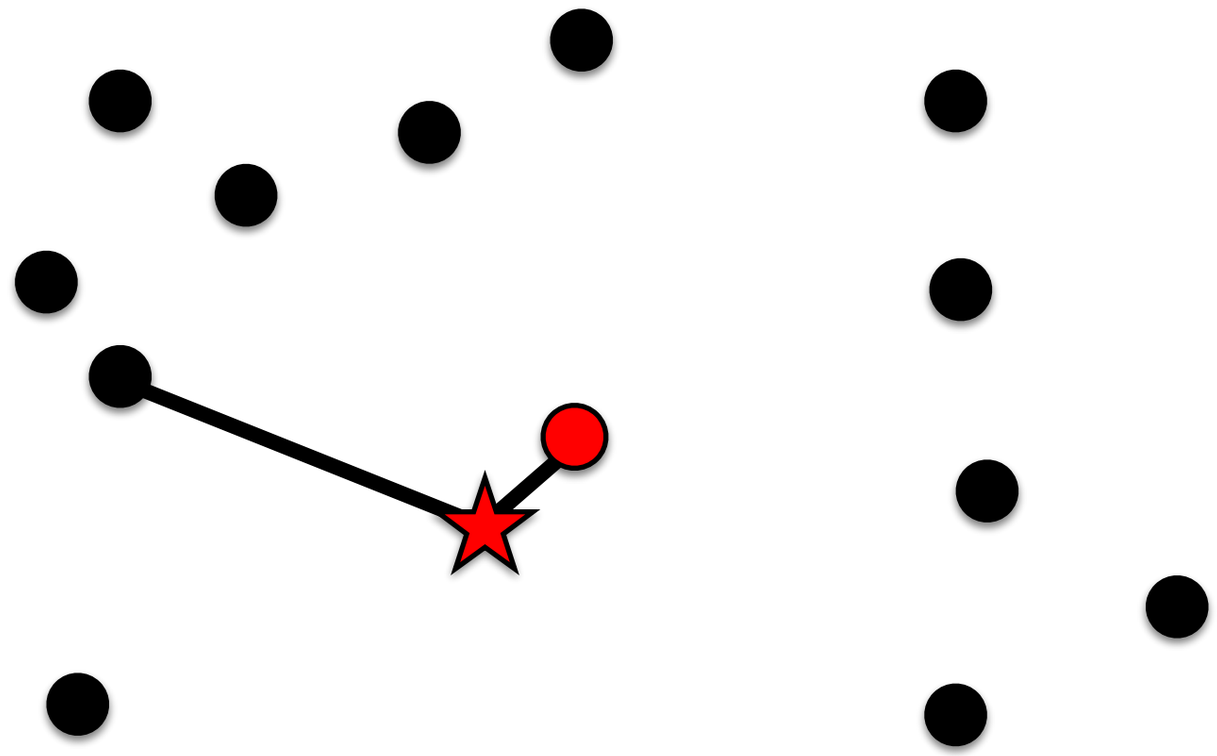
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

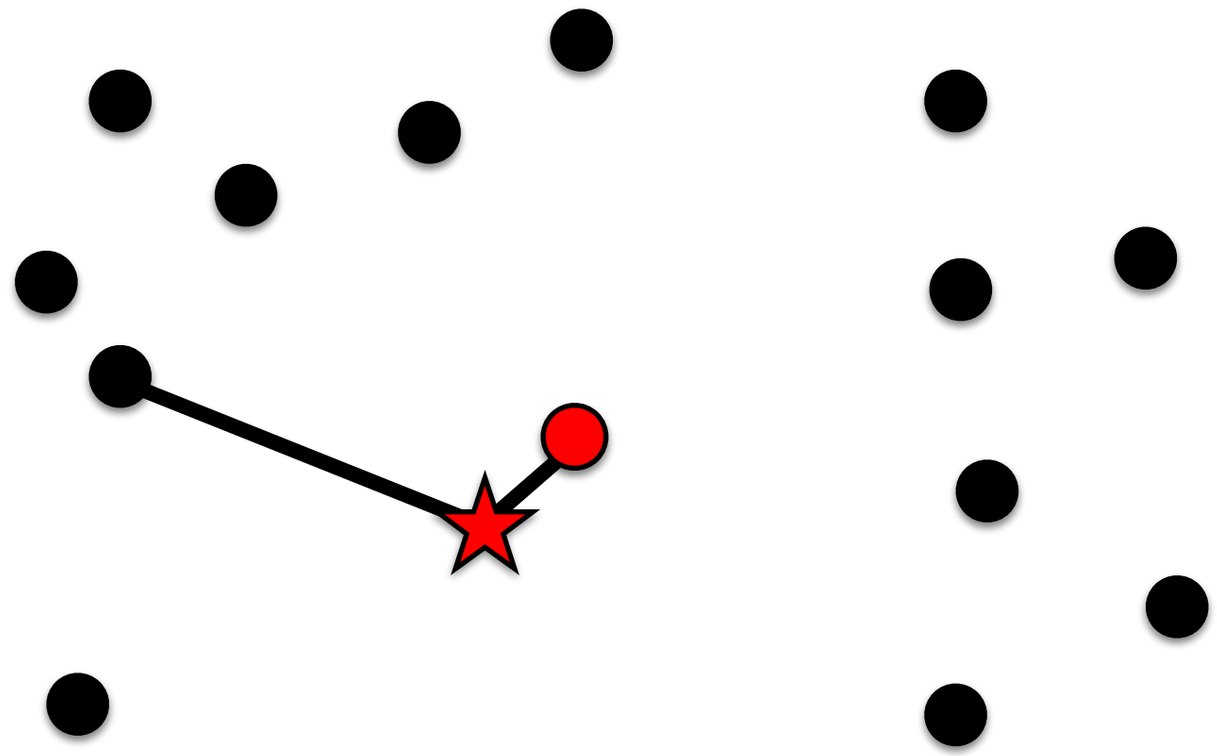
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

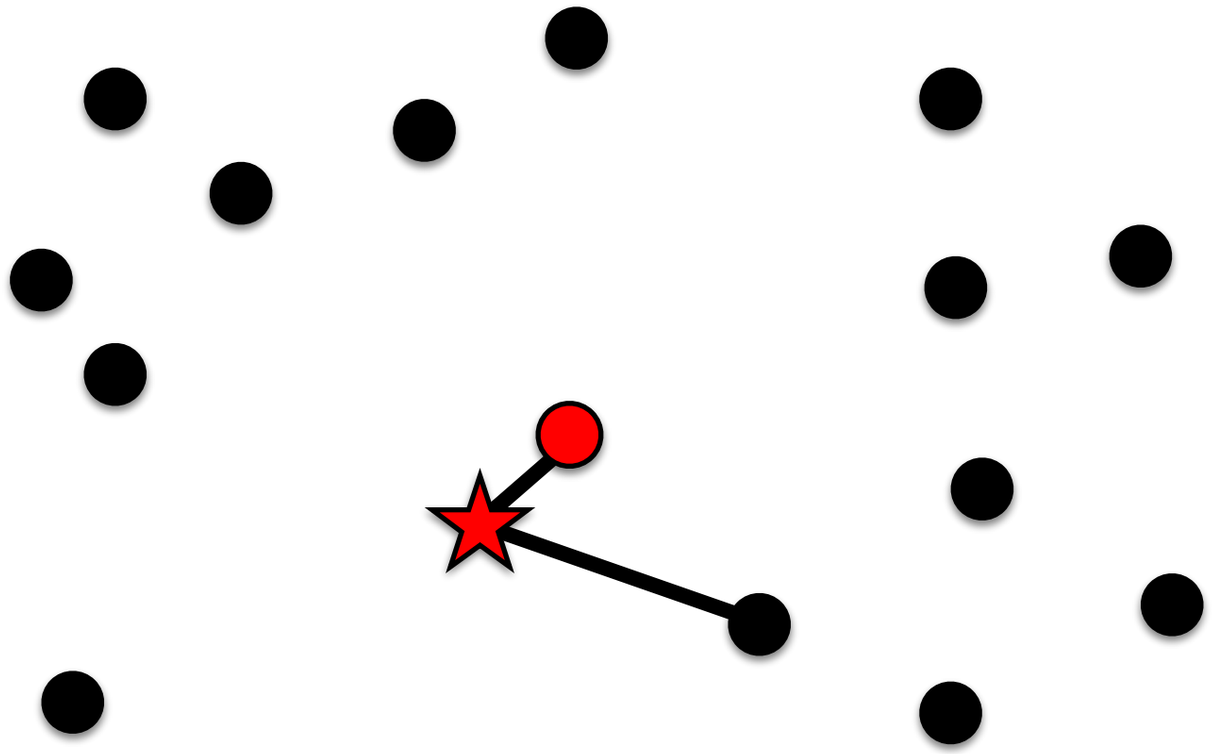
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

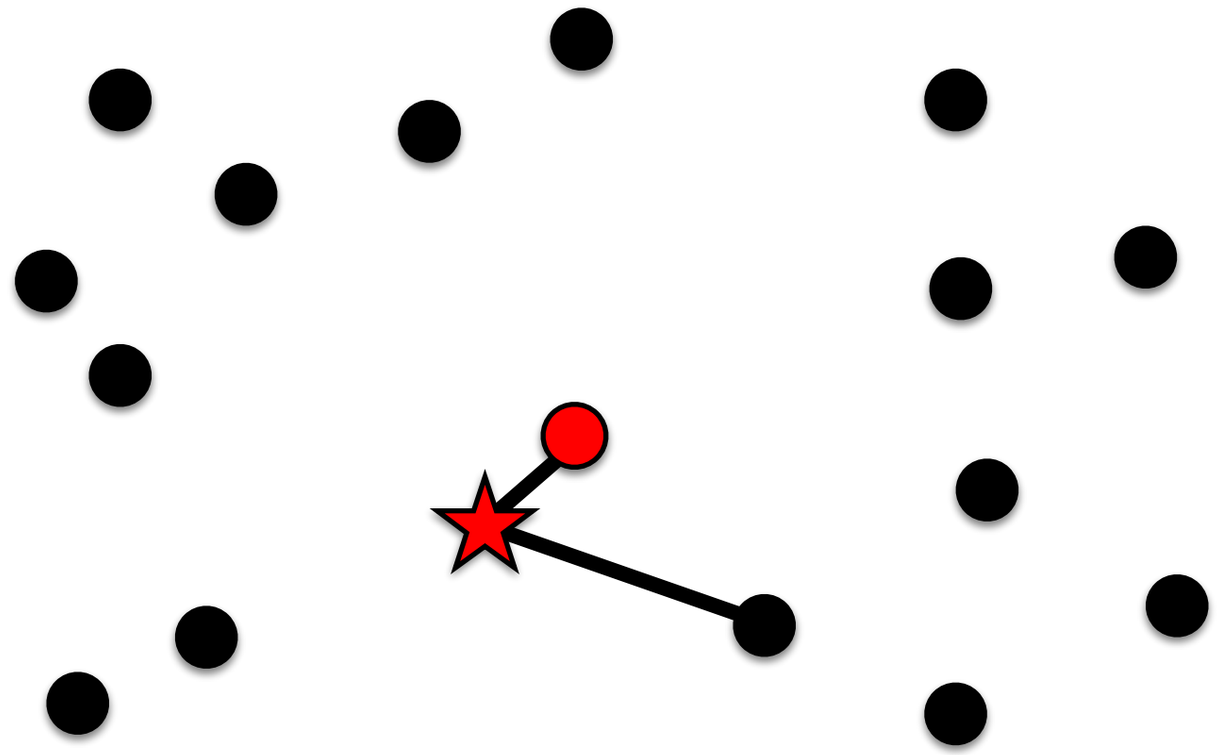
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

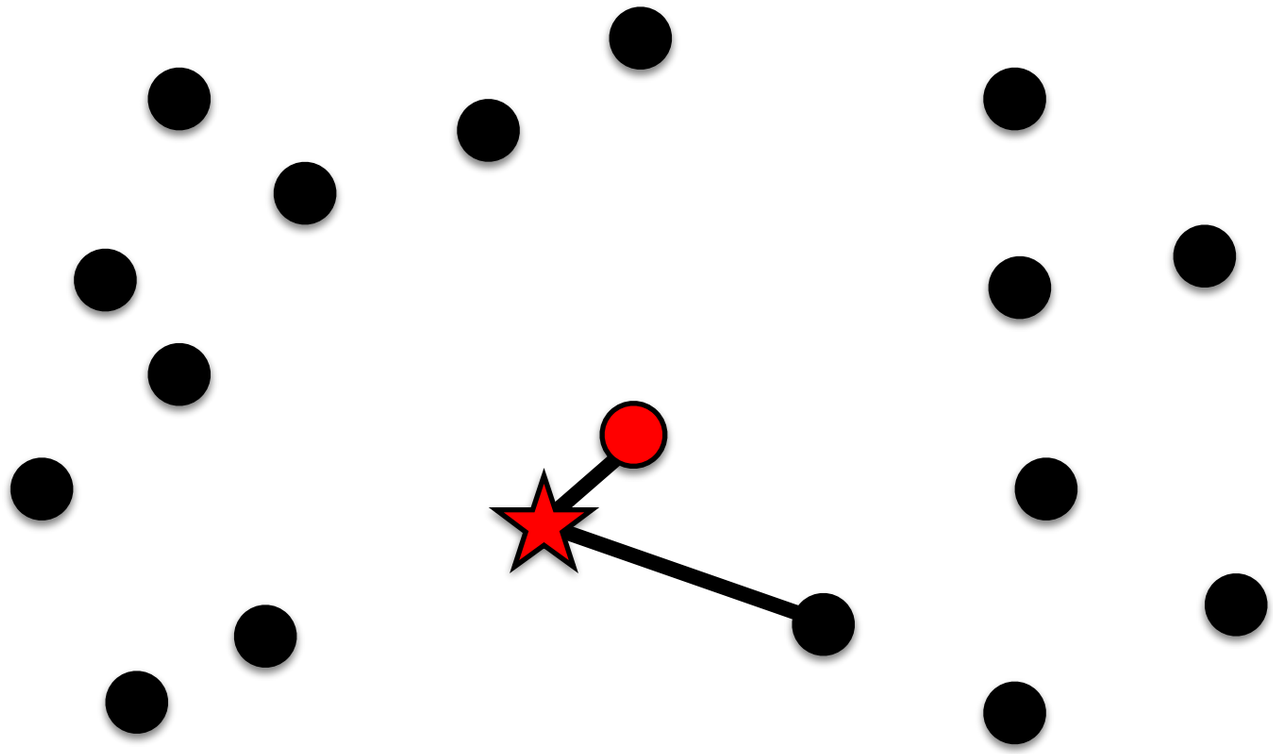
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

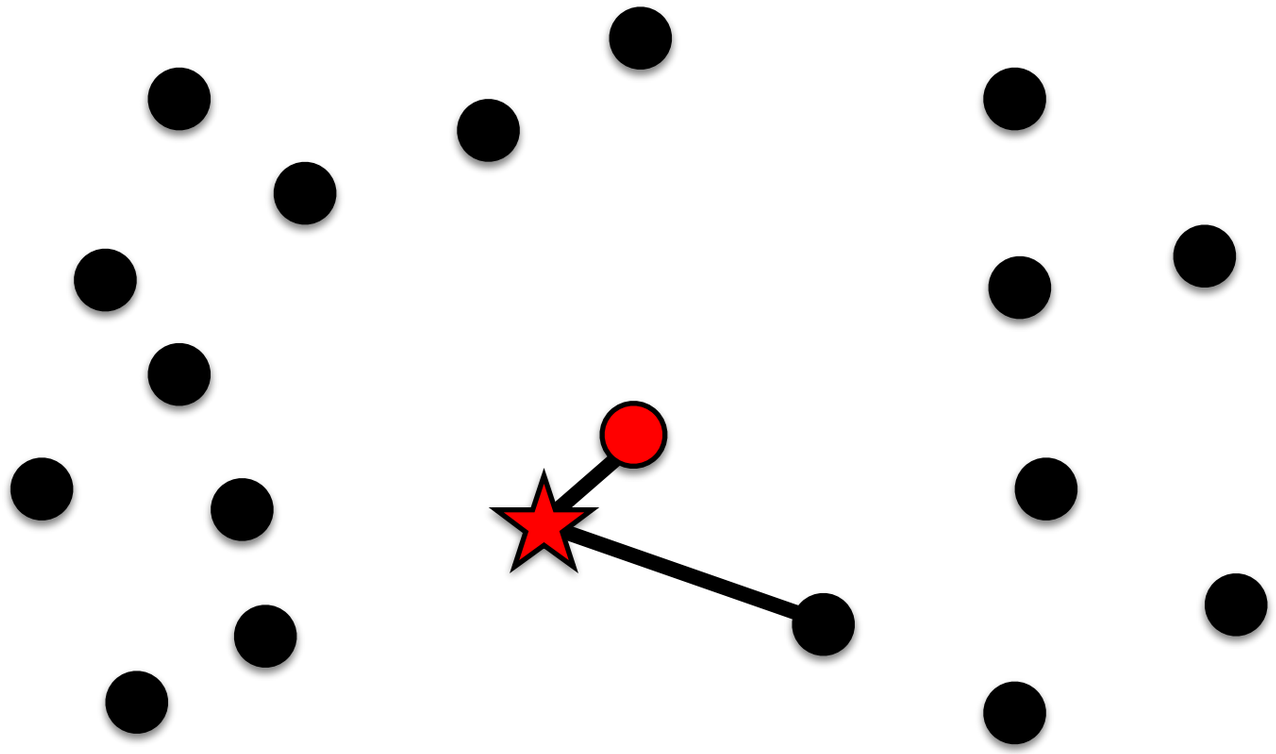
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

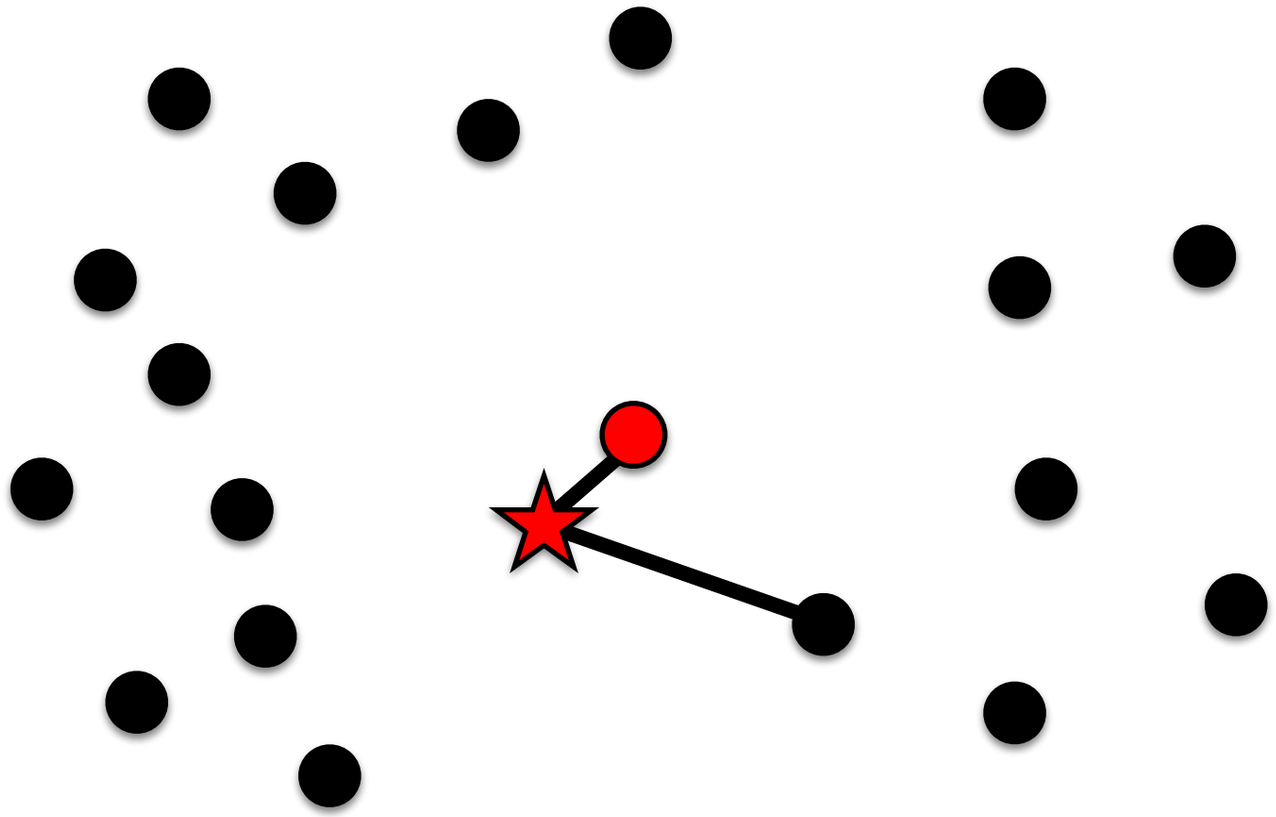
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

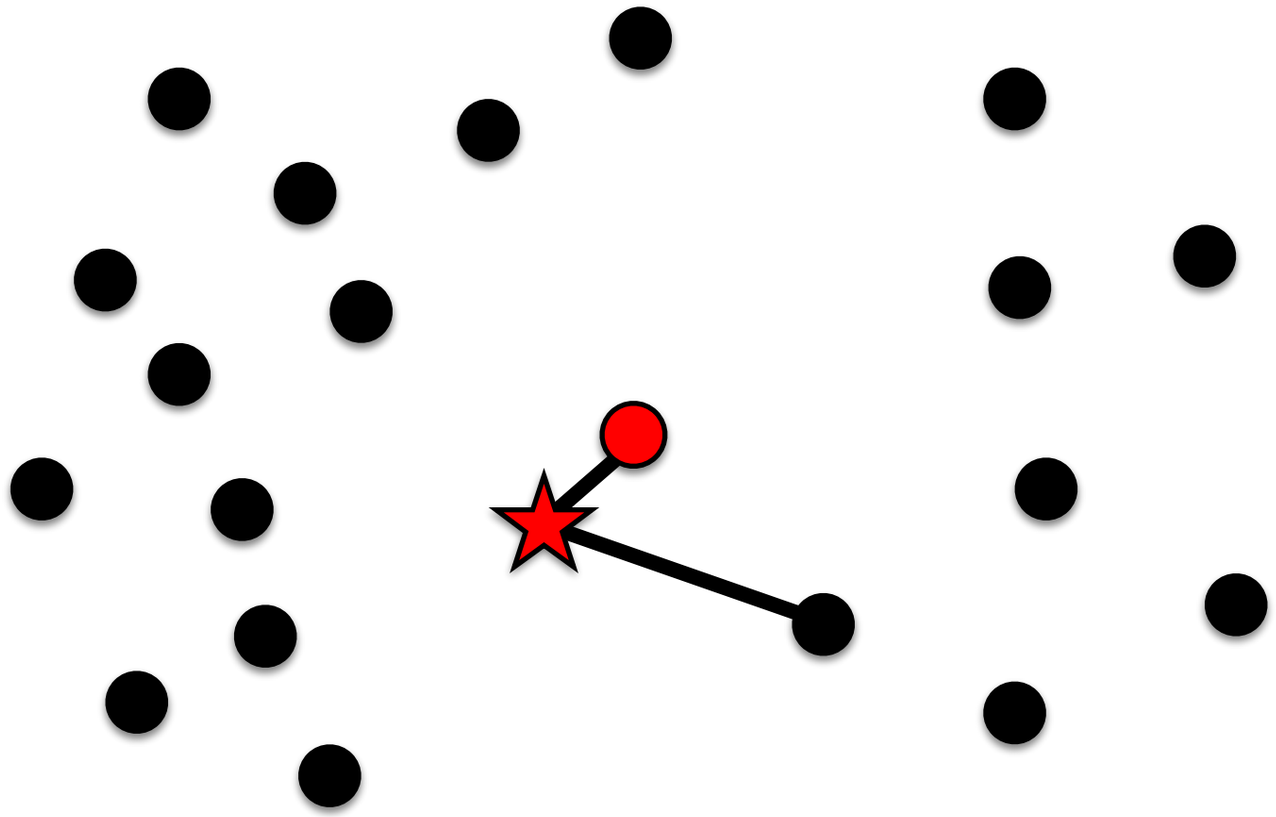
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

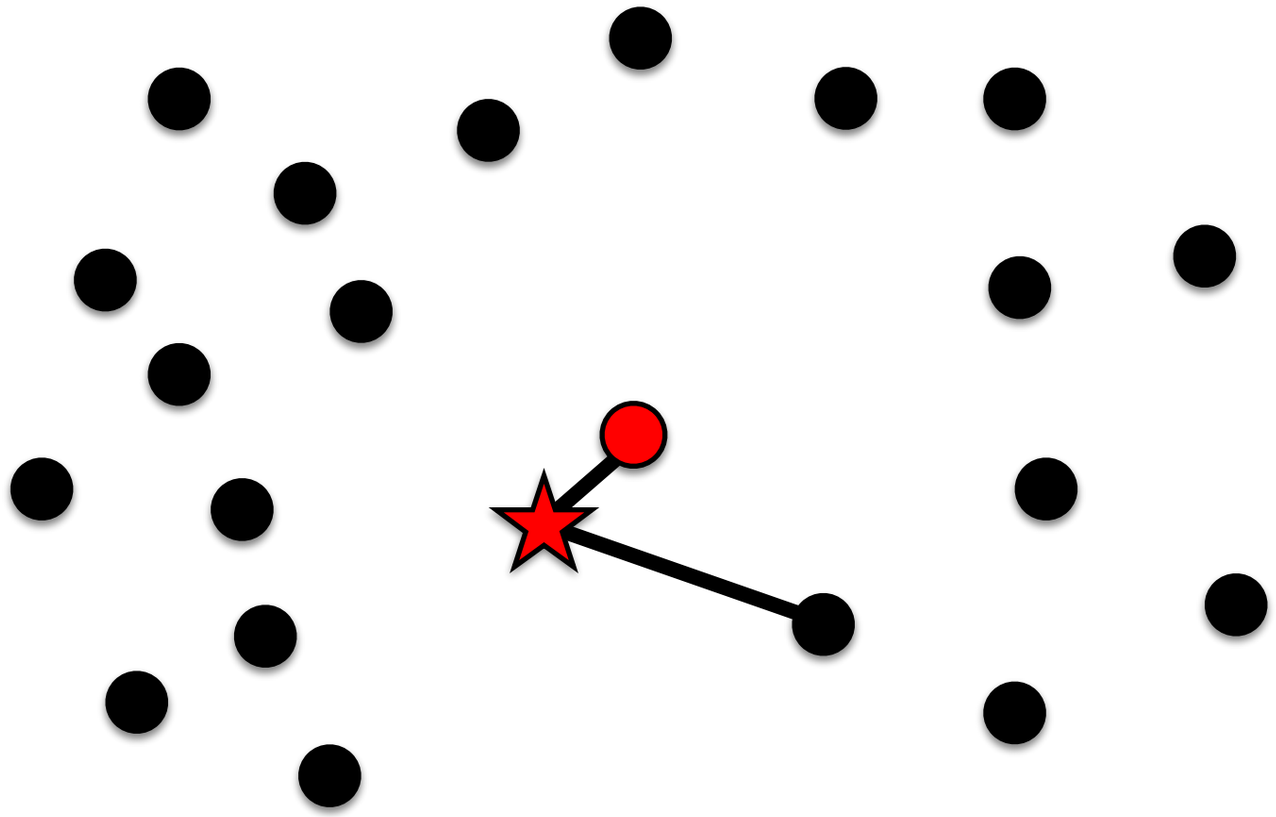
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

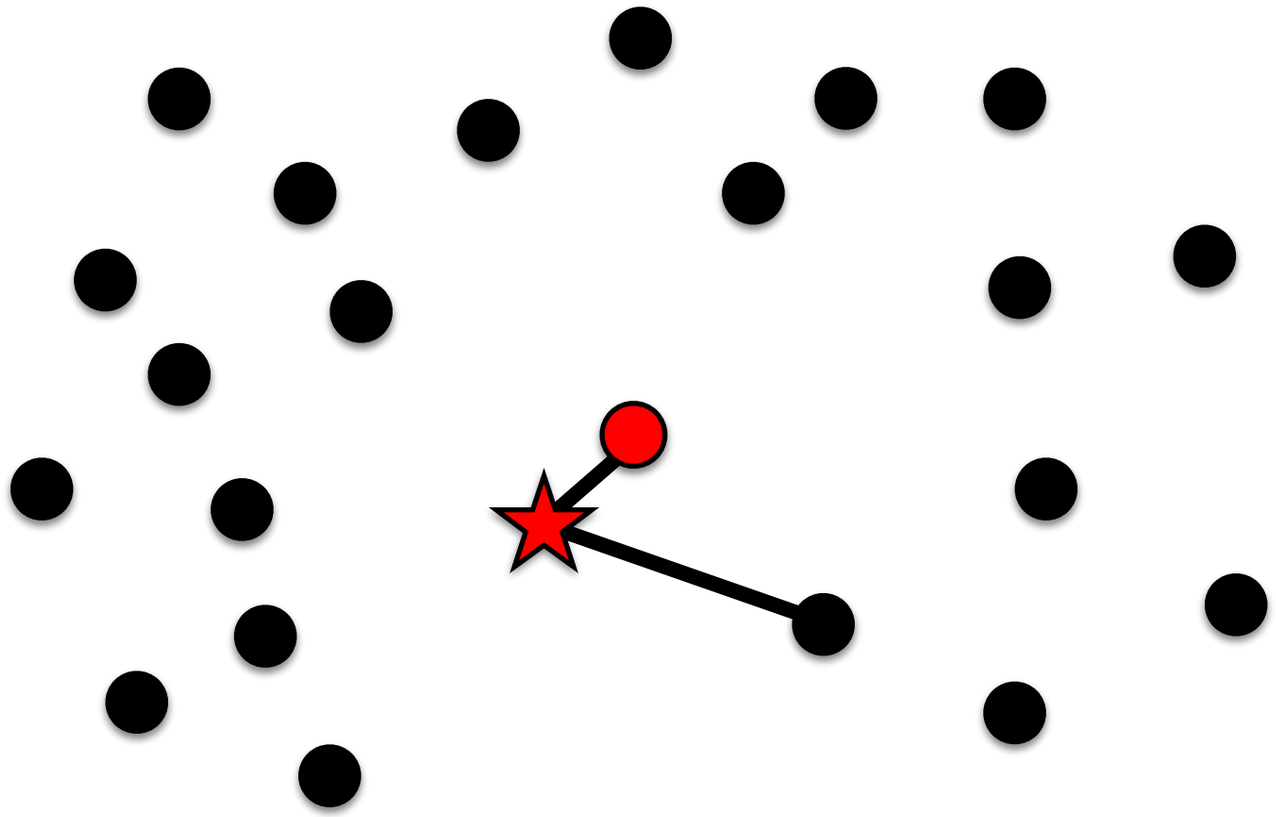
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

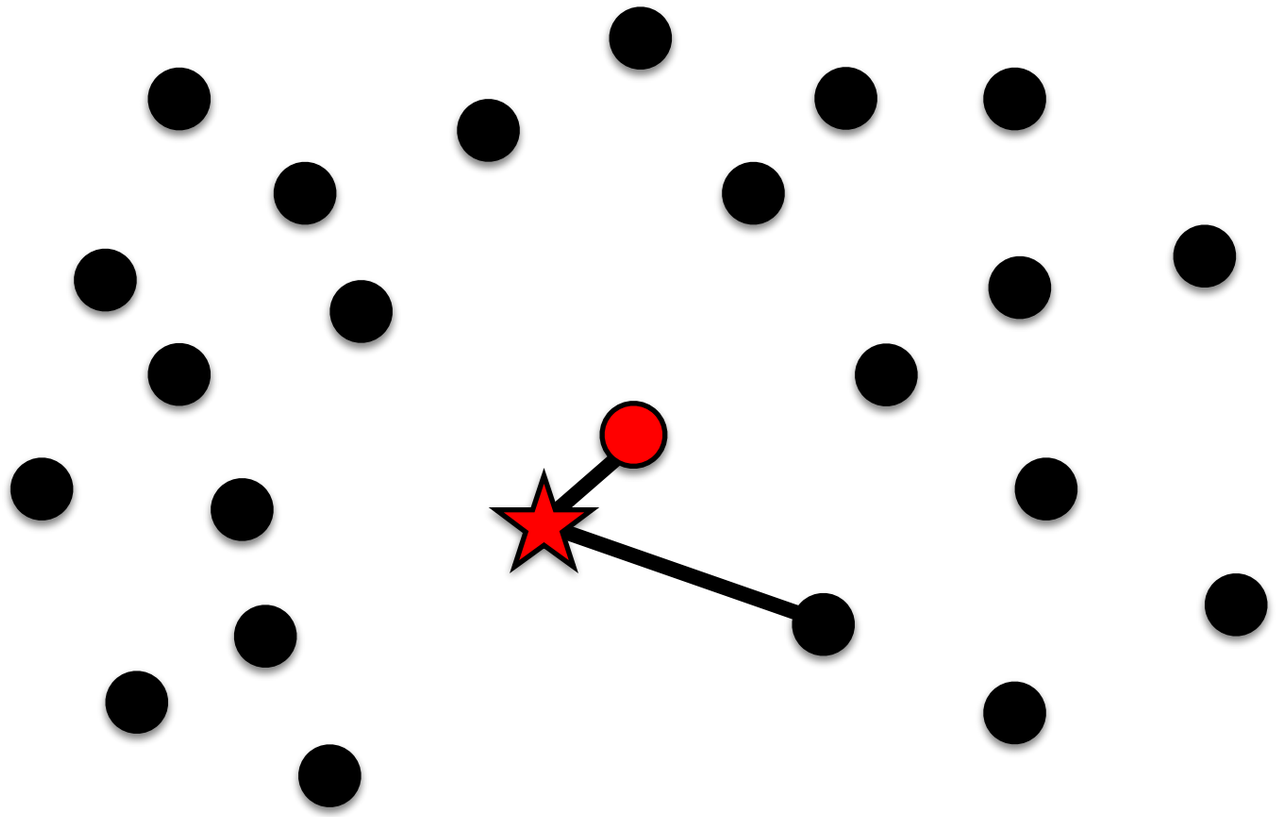
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

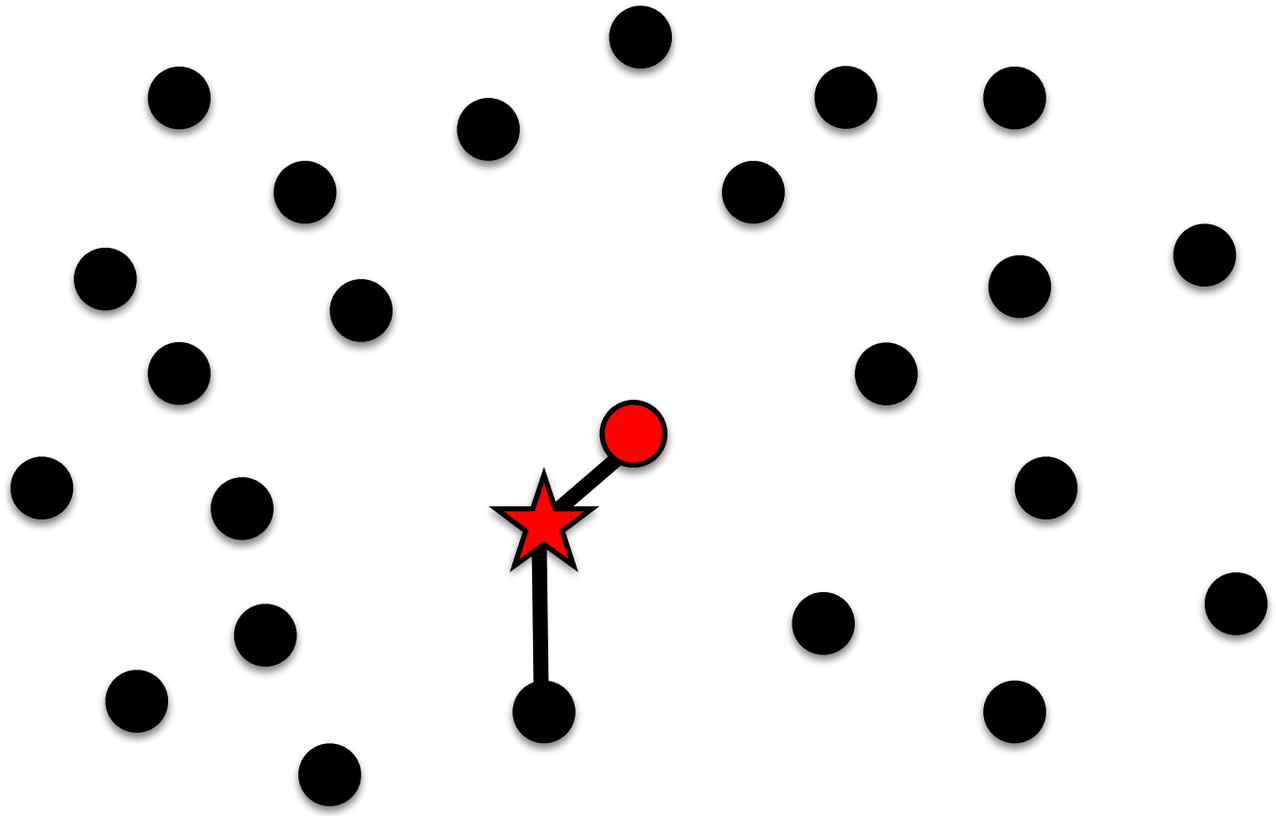
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

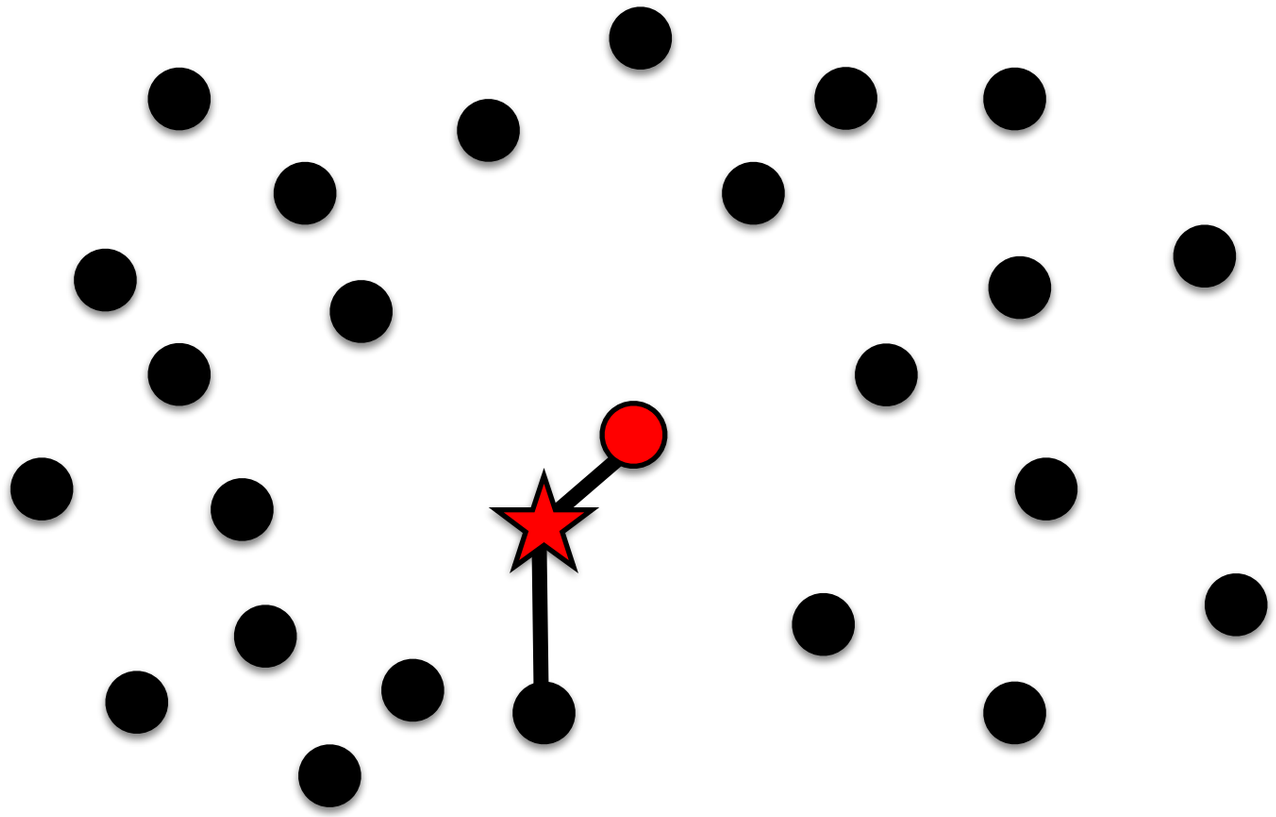
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

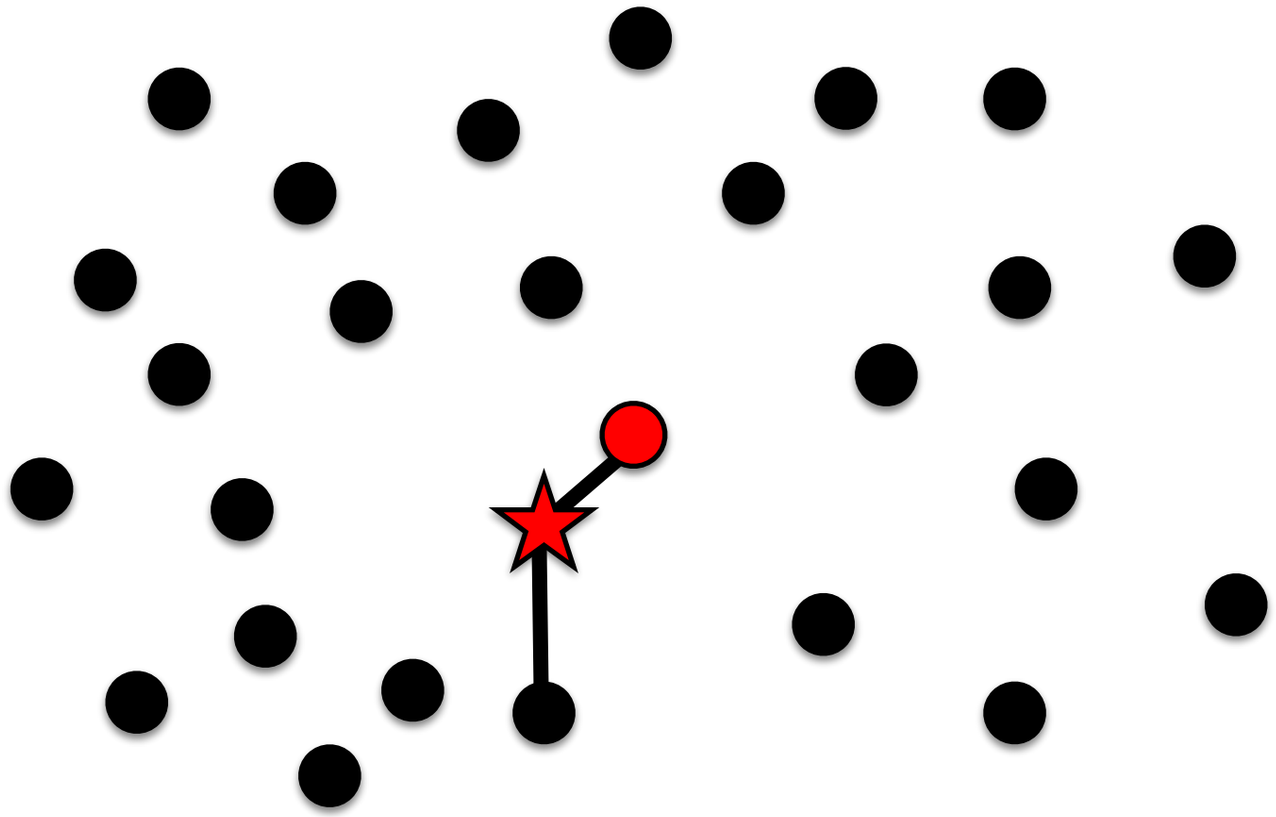
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

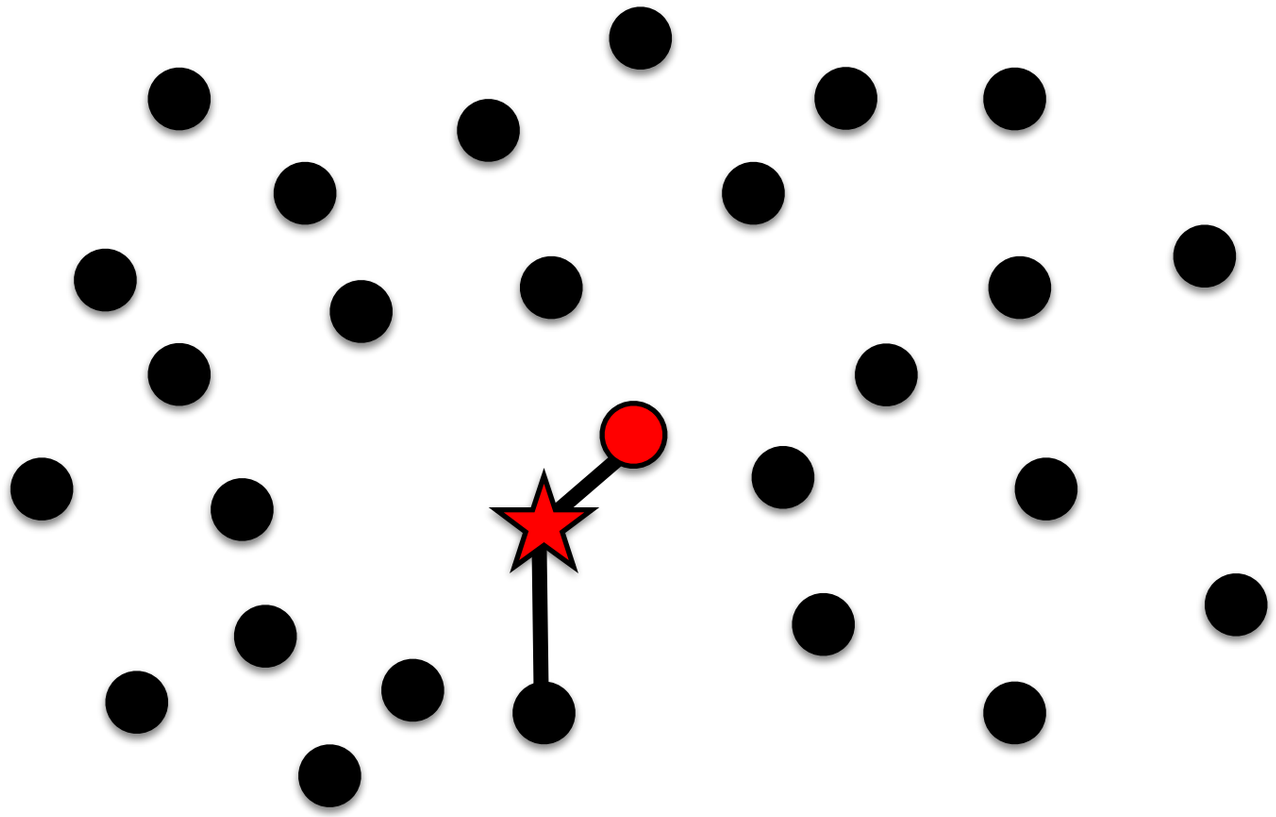
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

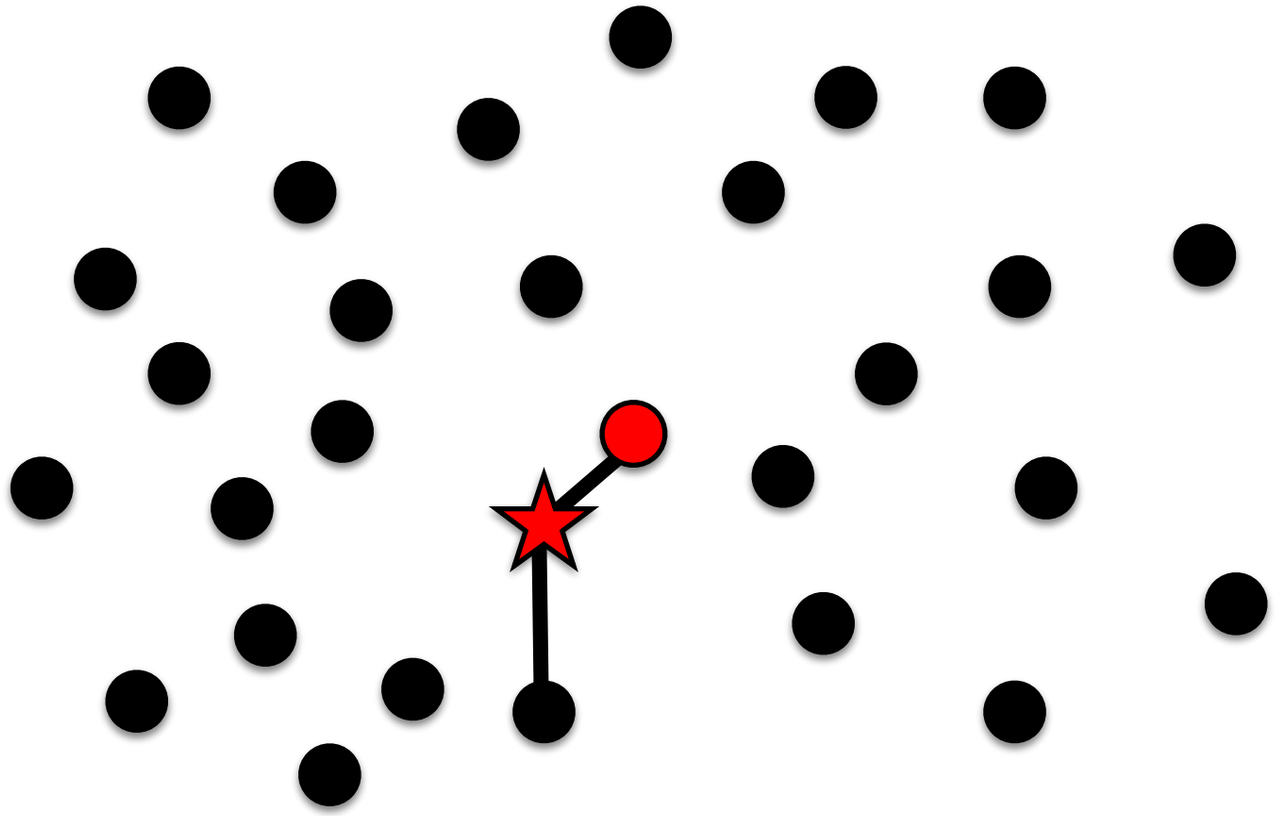
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

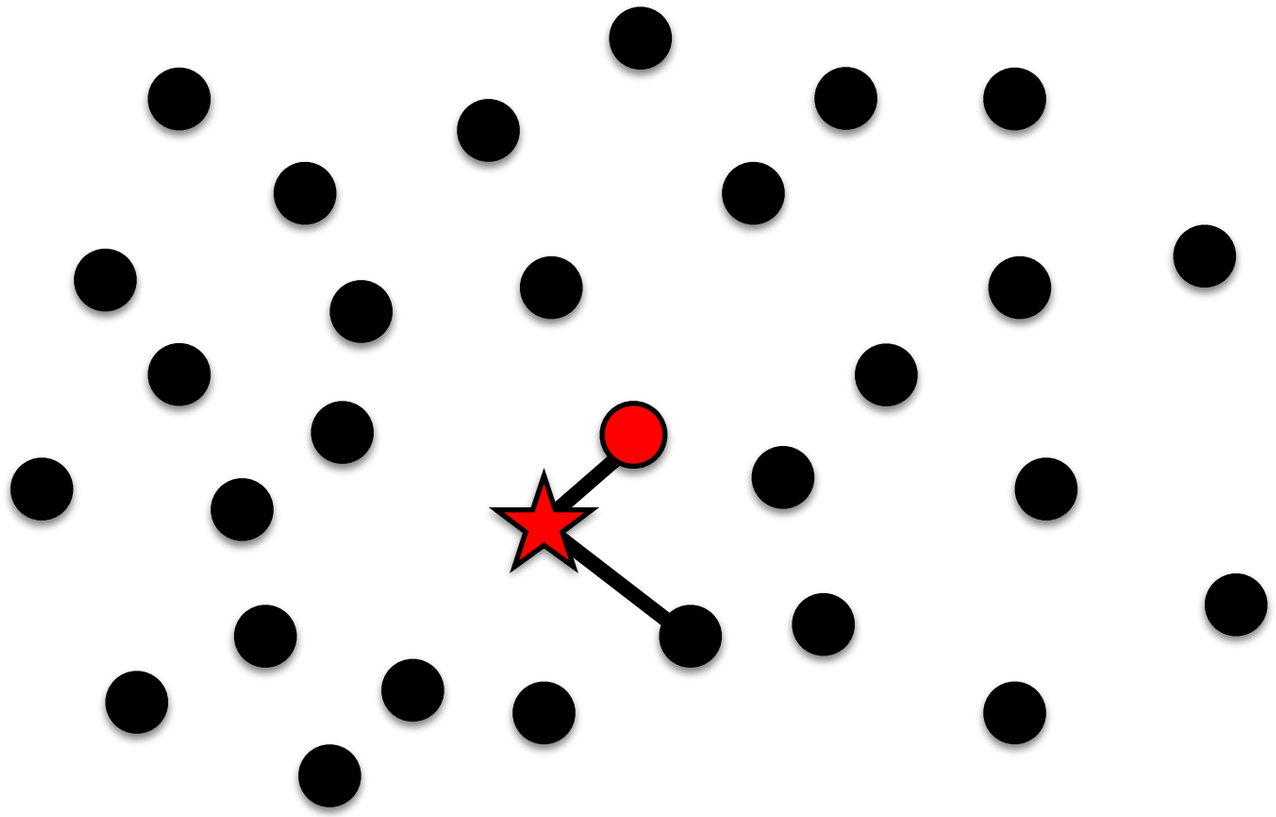
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

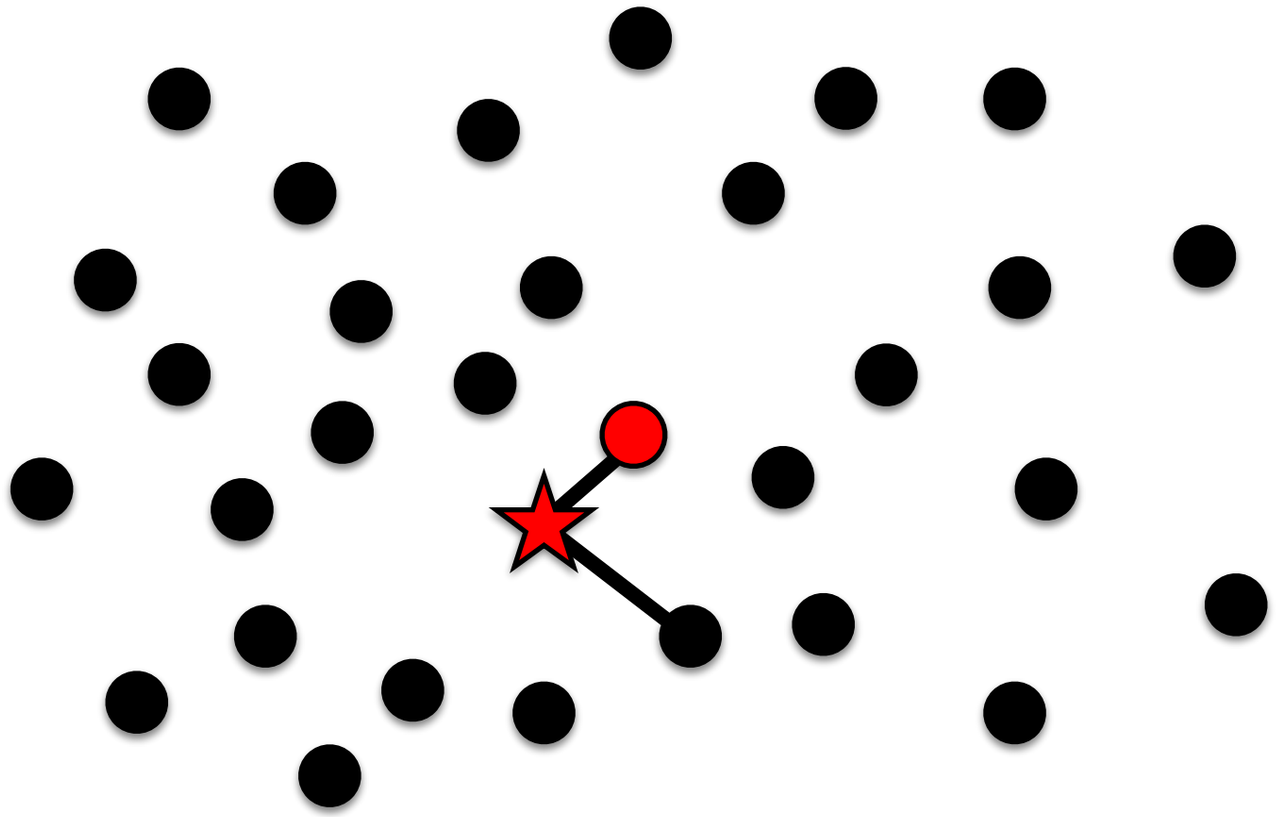
The Ratio Test Problem



- Larger models contain more locally similar structures

  ➡ Distance to second nearest neighbor decreases

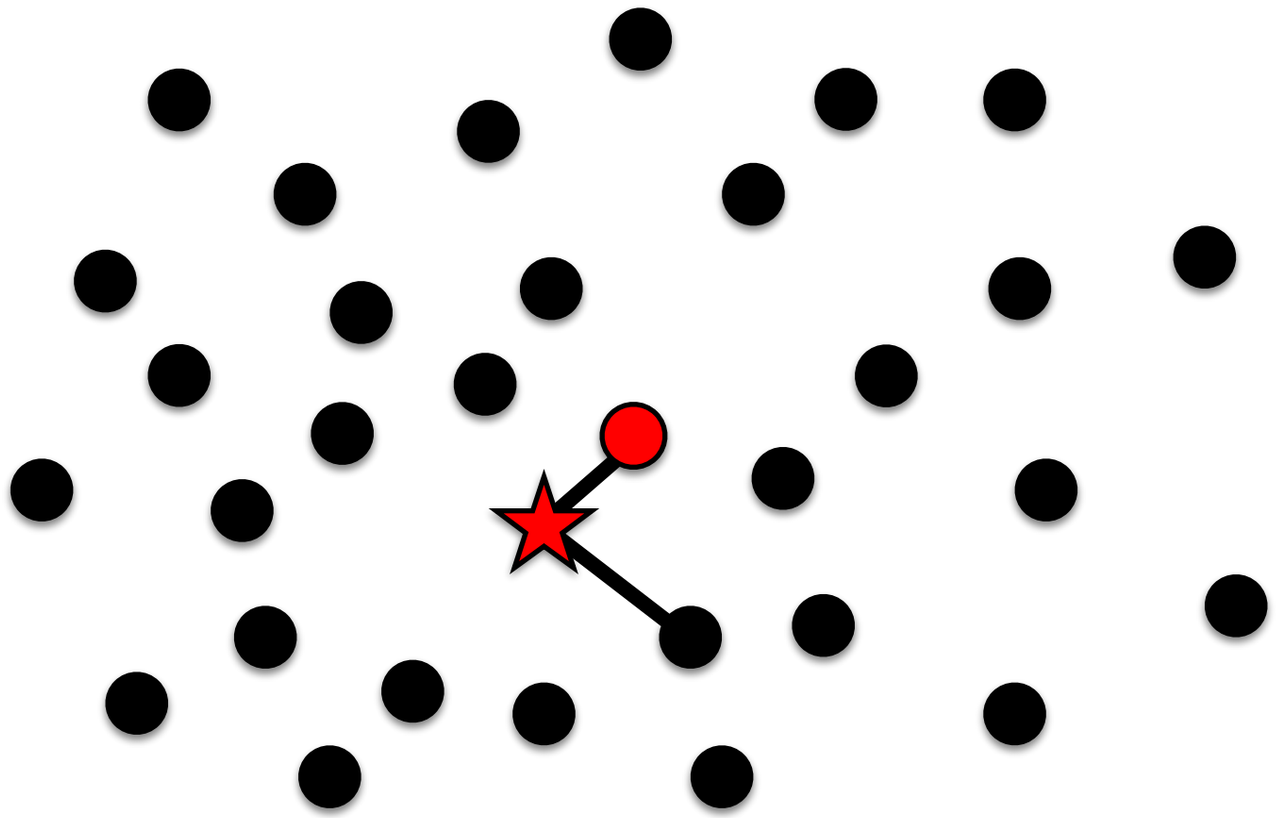
The Ratio Test Problem



- Larger models contain more locally similar structures

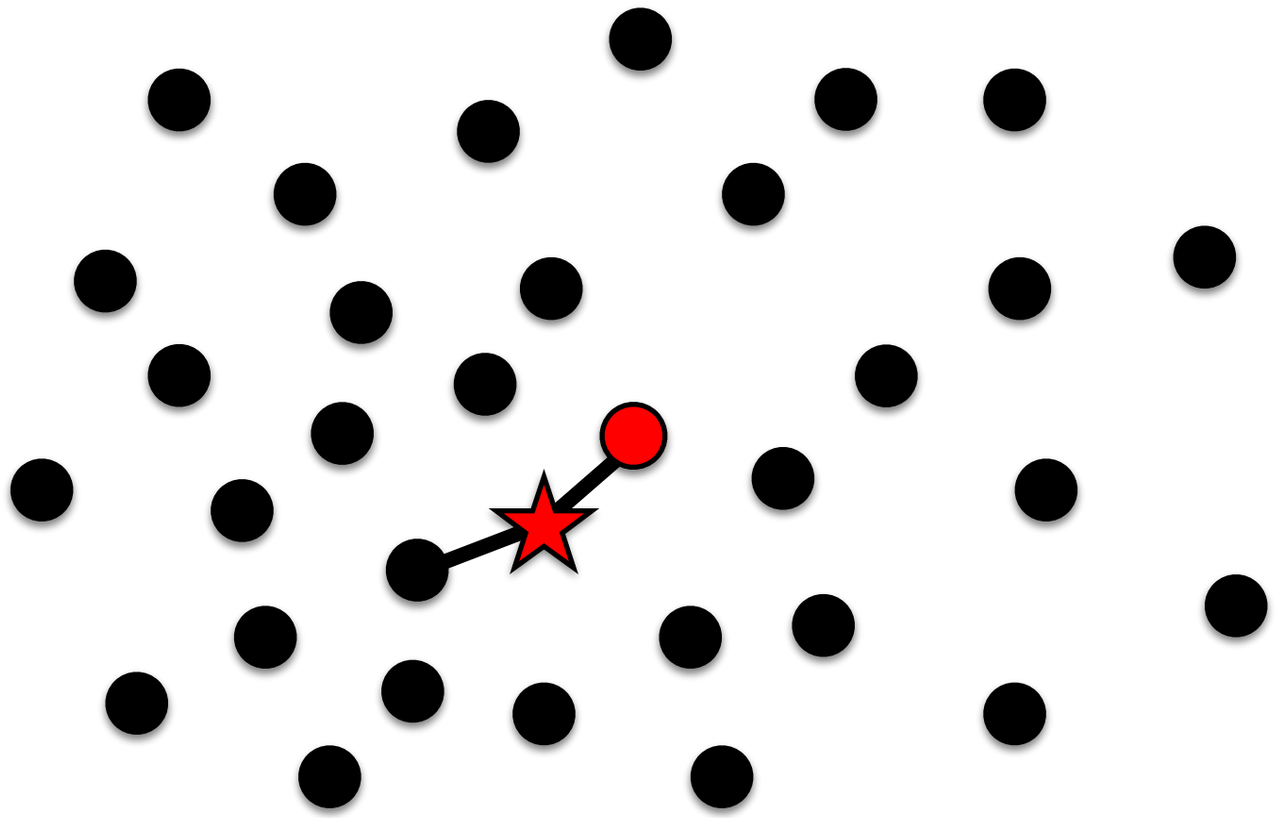
  ➡ Distance to second nearest neighbor decreases

The Ratio Test Problem



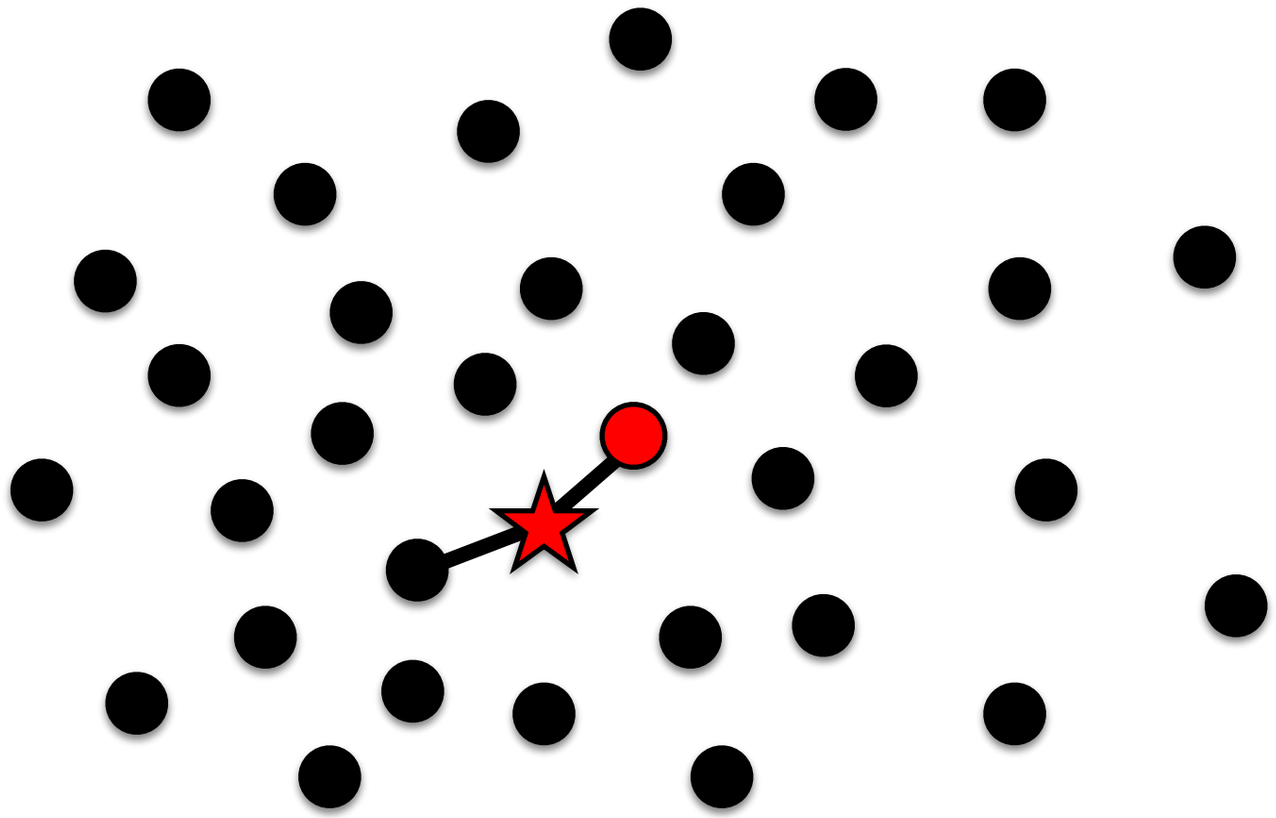
- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases

The Ratio Test Problem



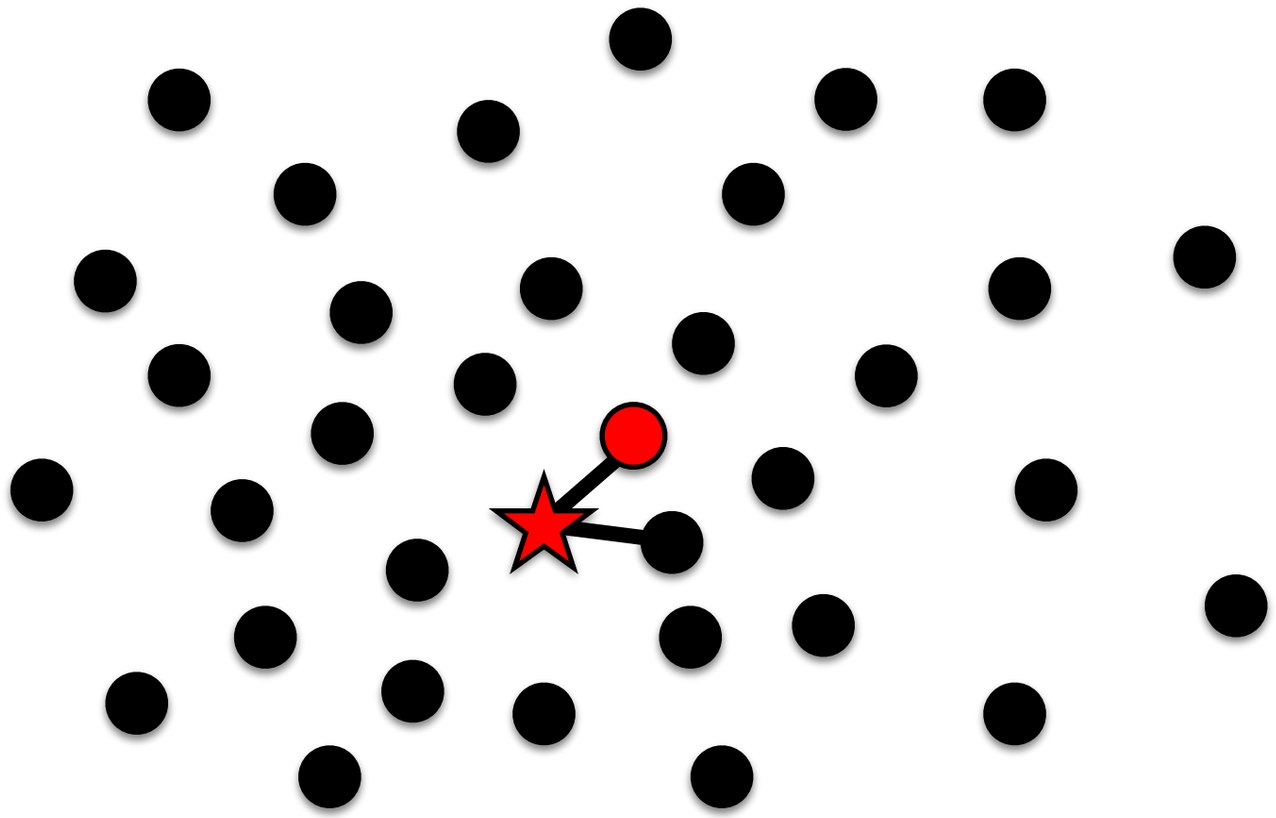
- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases

The Ratio Test Problem



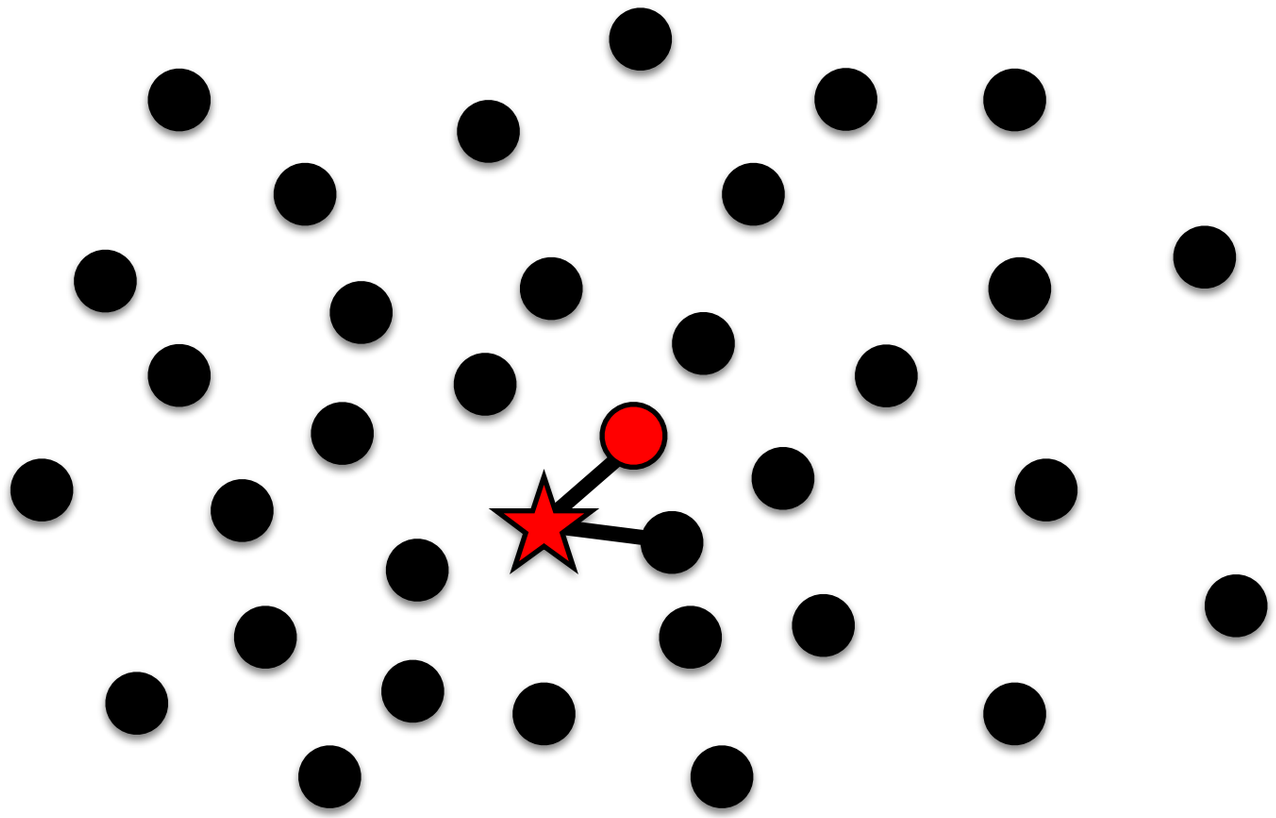
- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases

The Ratio Test Problem



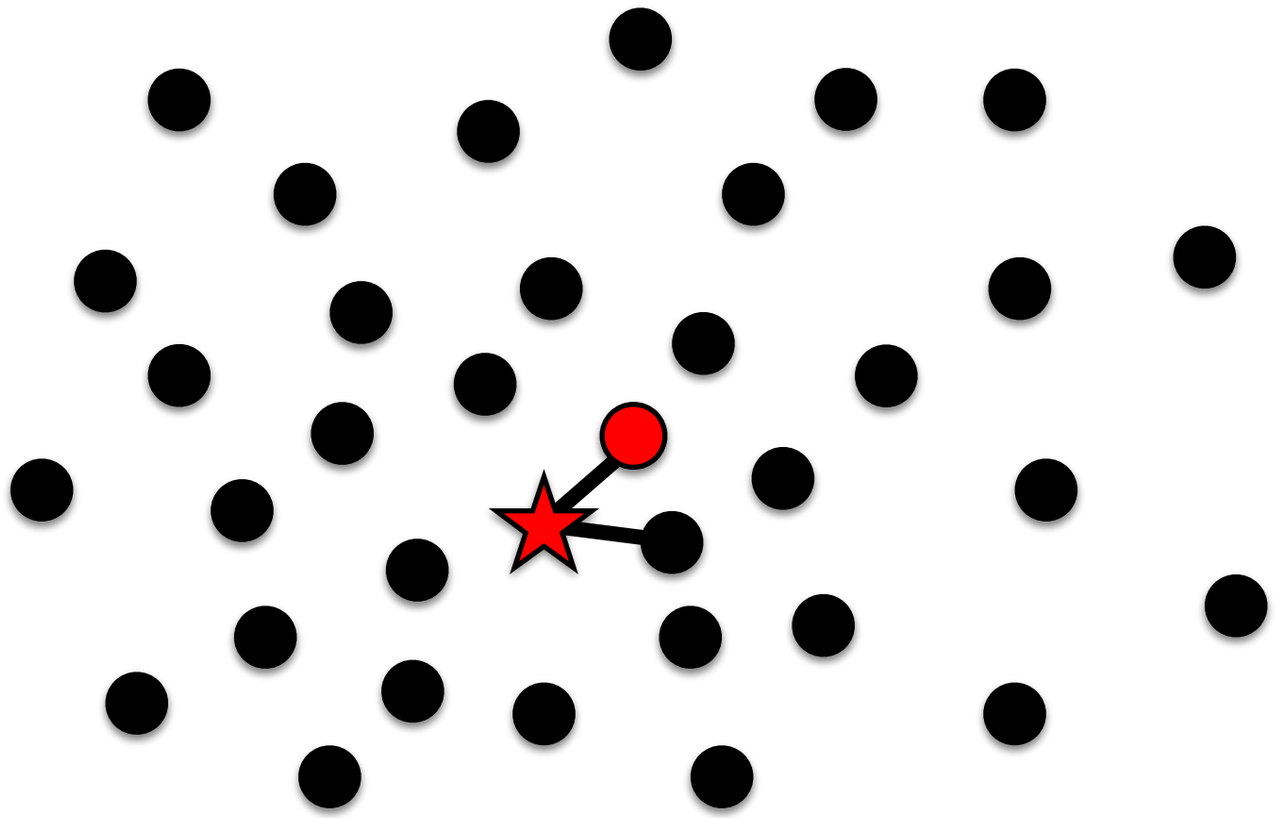
- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases

The Ratio Test Problem



- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases
  - ➡ Ratio test for 2D-to-3D search rejects more and more matches

The Ratio Test Problem



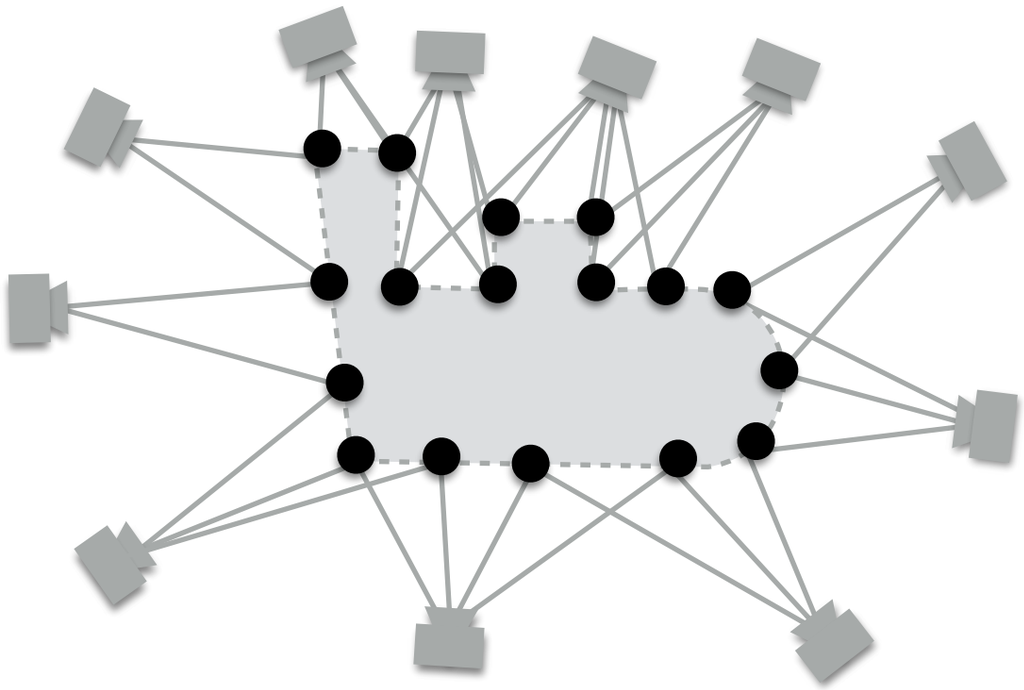
- Larger models contain more locally similar structures
  - ➡ Distance to second nearest neighbor decreases
  - ➡ Ratio test for 2D-to-3D search rejects more and more matches
- Active Search lessens this problem, but still relies on 2D-to-3D matches

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

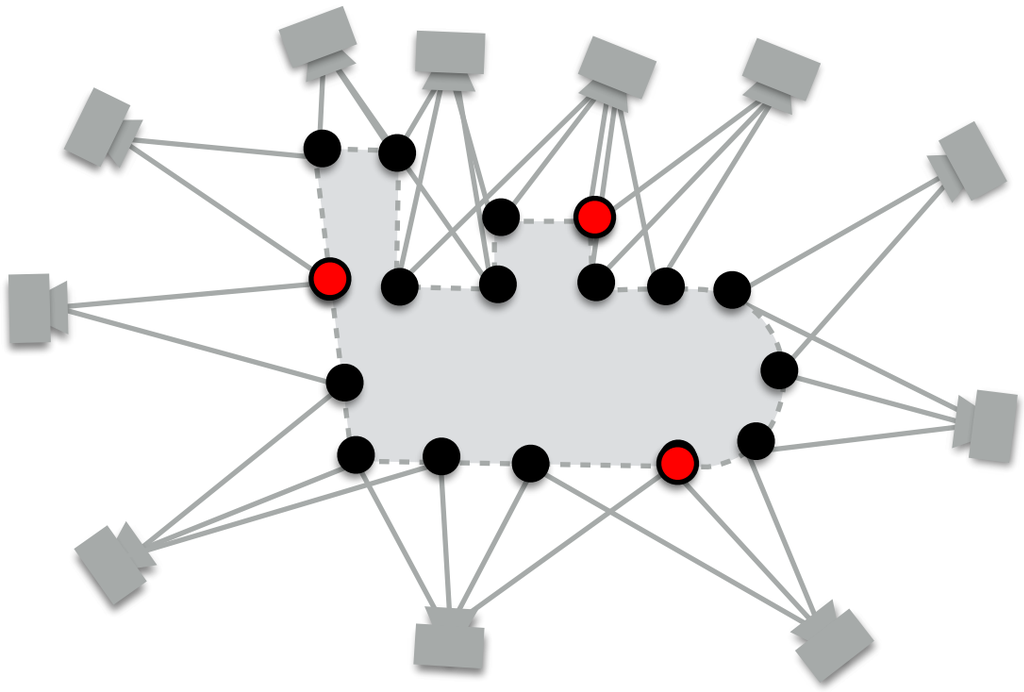
Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC



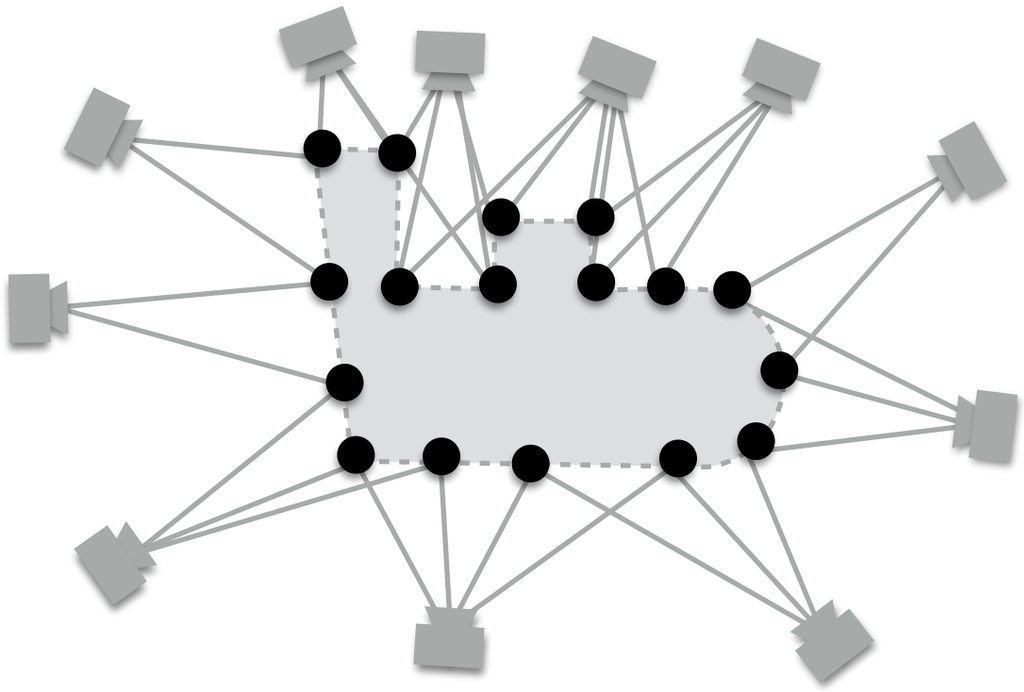
Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC



Sampling With Co-occurrence Priors

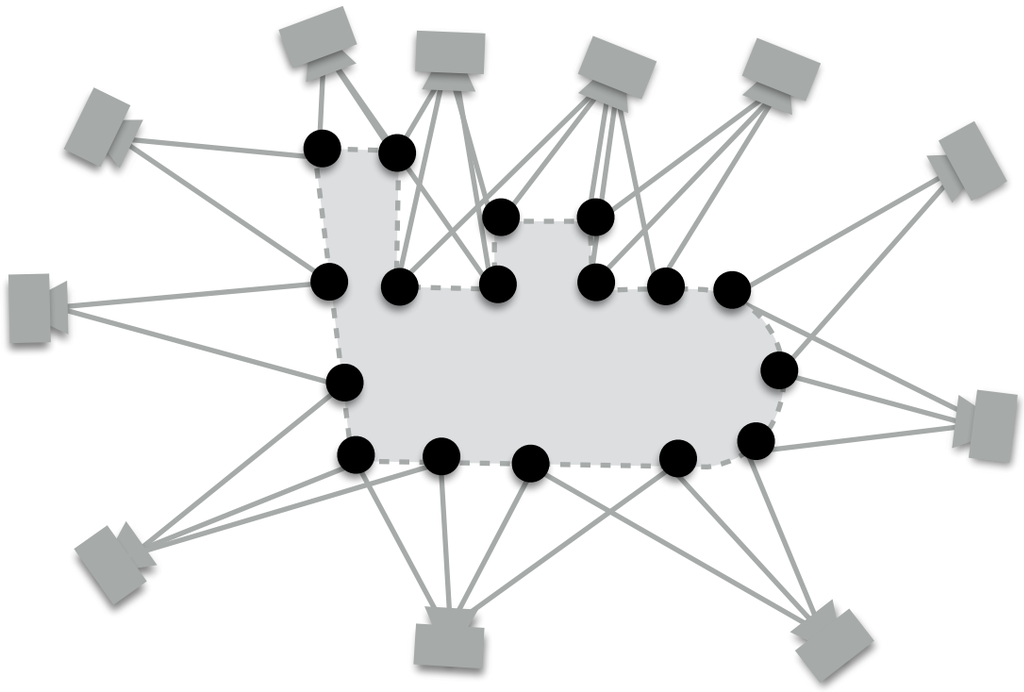
Idea: Increase ratio test threshold, shift responsibility to RANSAC



- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]

Sampling With Co-occurrence Priors

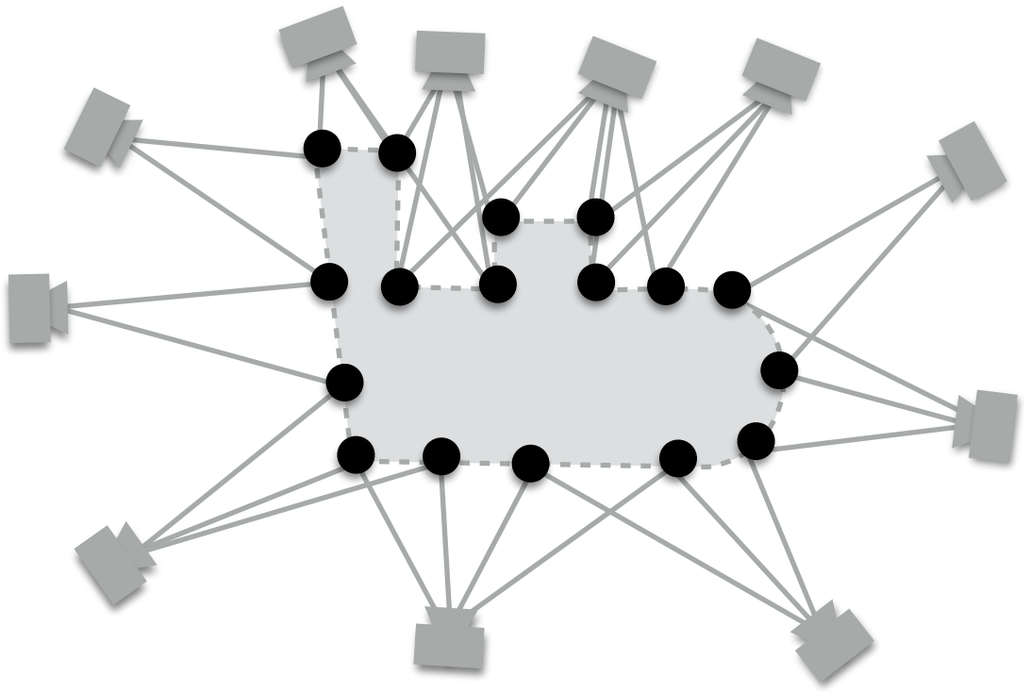
Idea: Increase ratio test threshold, shift responsibility to RANSAC



- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}

Sampling With Co-occurrence Priors

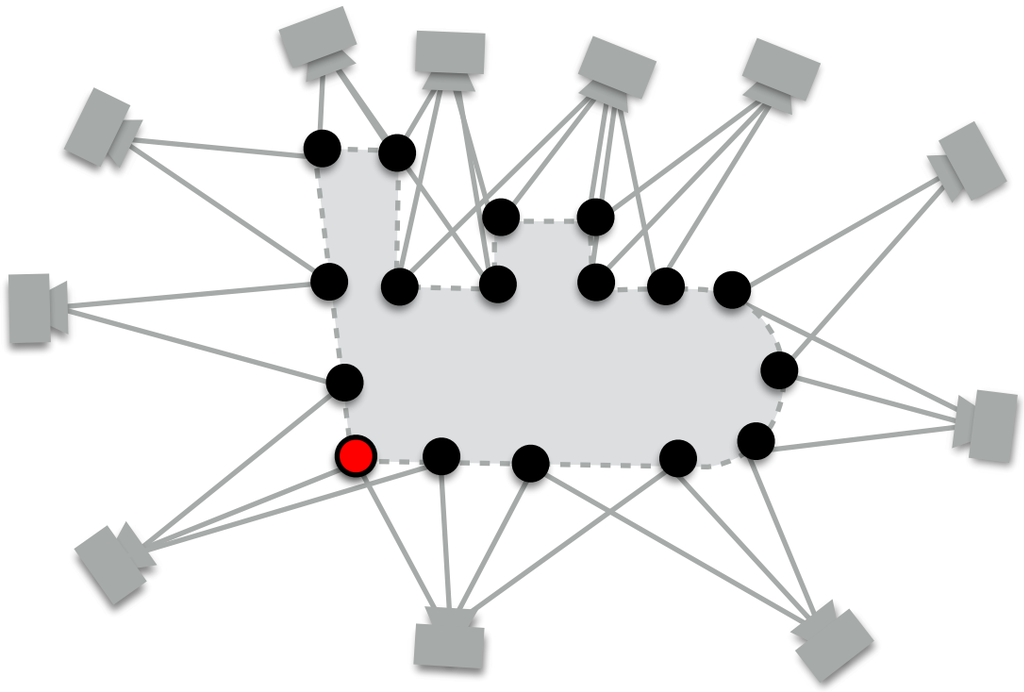
Idea: Increase ratio test threshold, shift responsibility to RANSAC



- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random

Sampling With Co-occurrence Priors

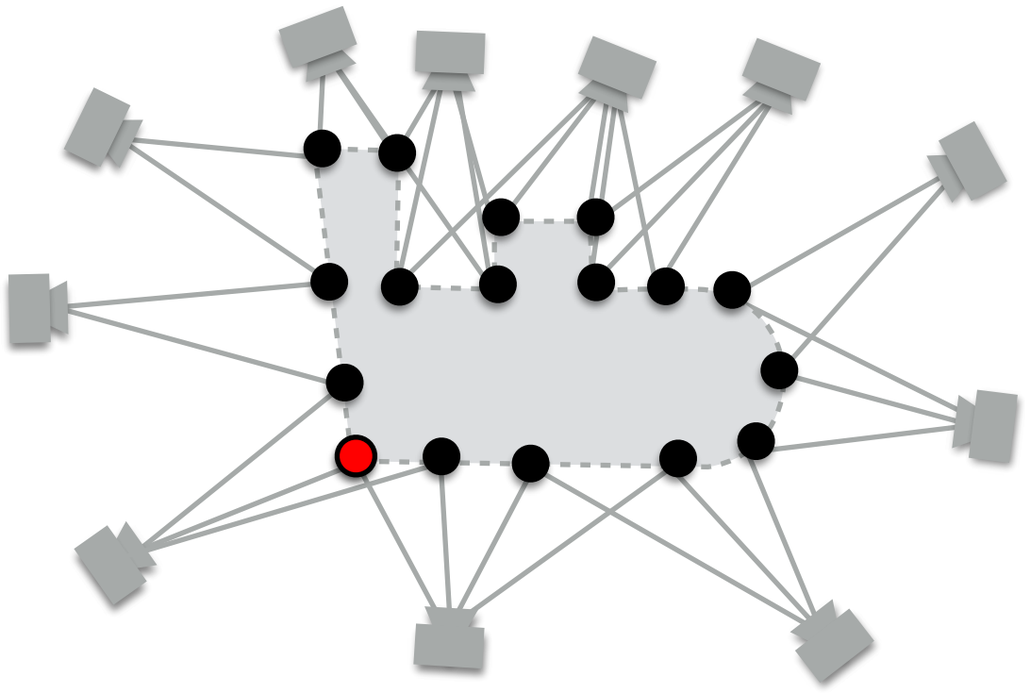
Idea: Increase ratio test threshold, shift responsibility to RANSAC



- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

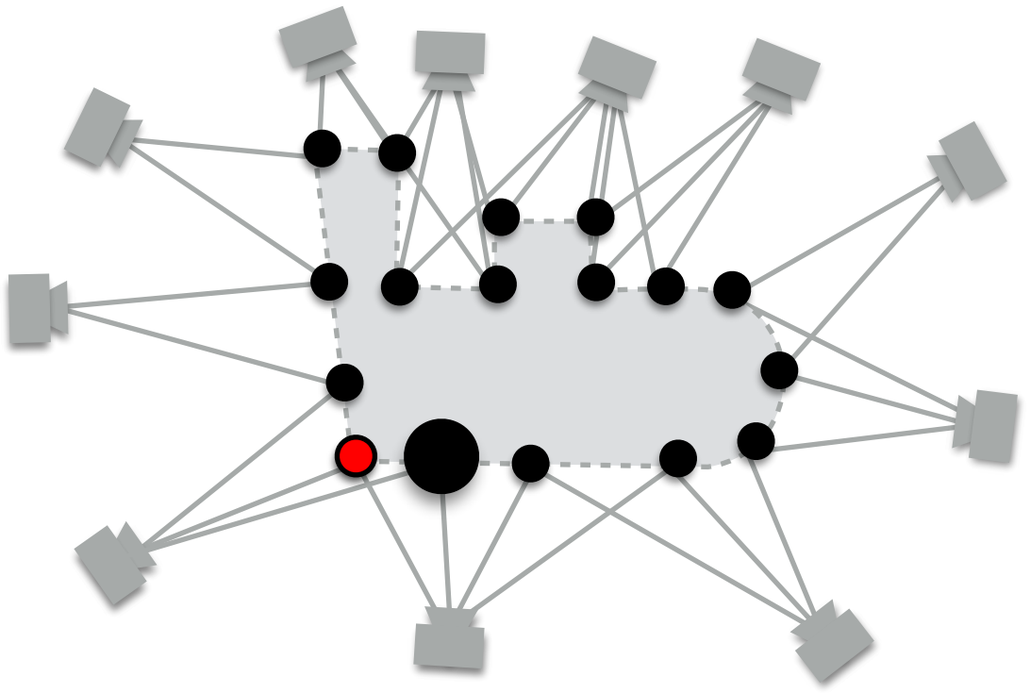


- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

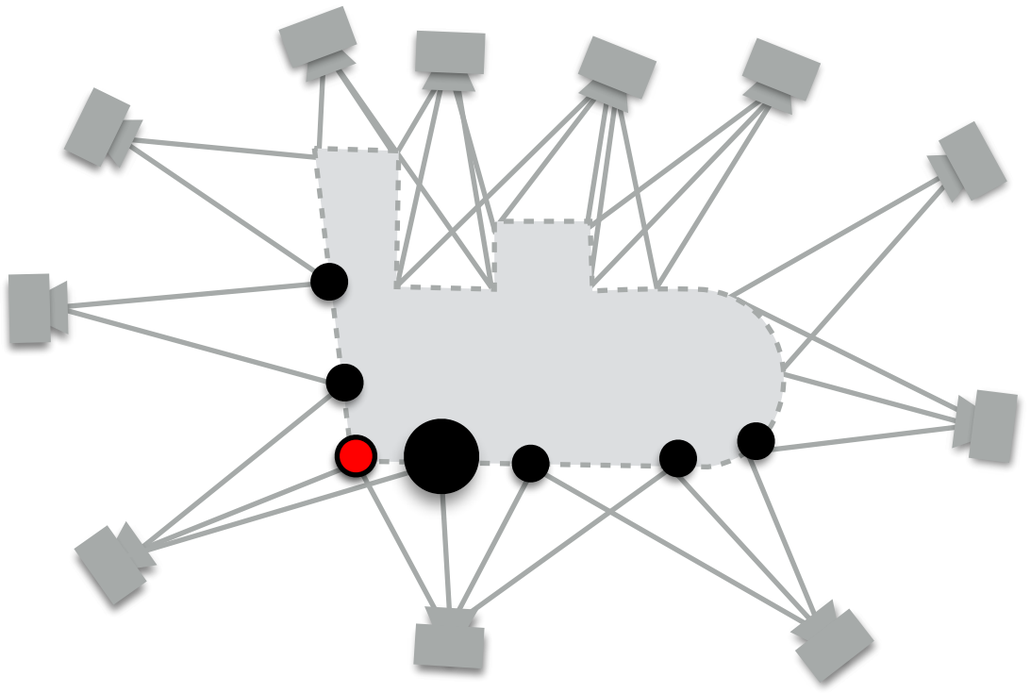


- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

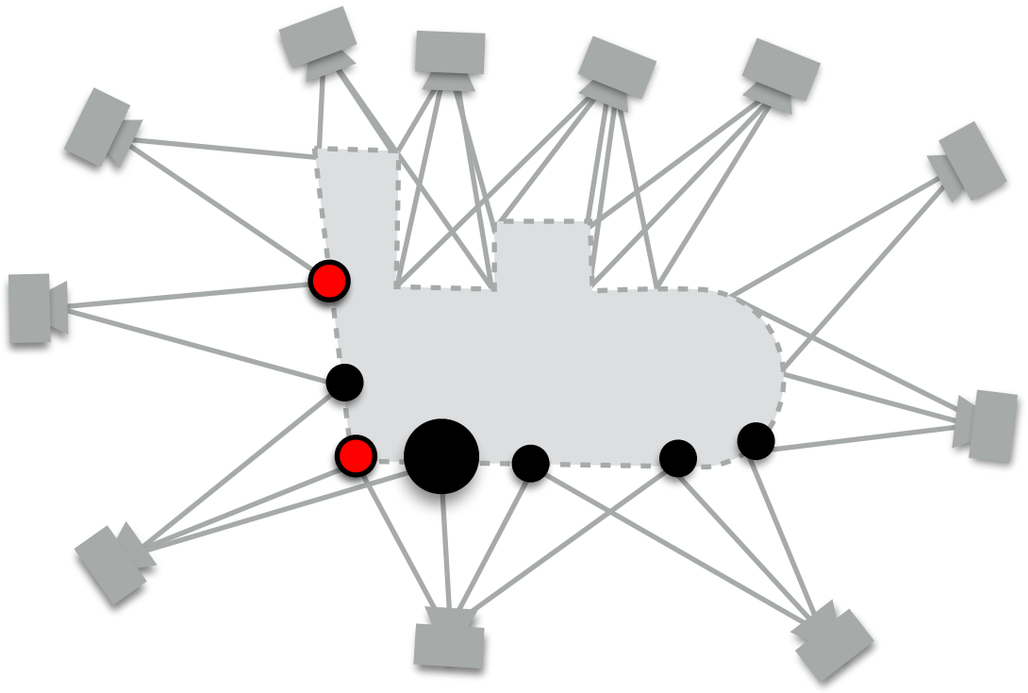


- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

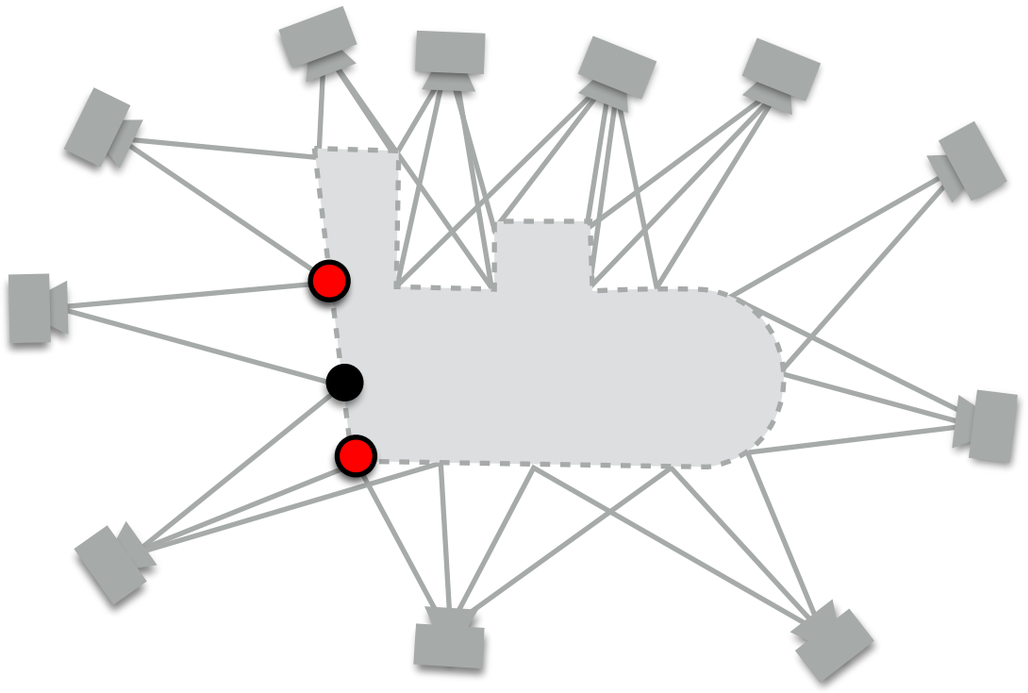


- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

Sampling With Co-occurrence Priors

Idea: Increase ratio test threshold, shift responsibility to RANSAC

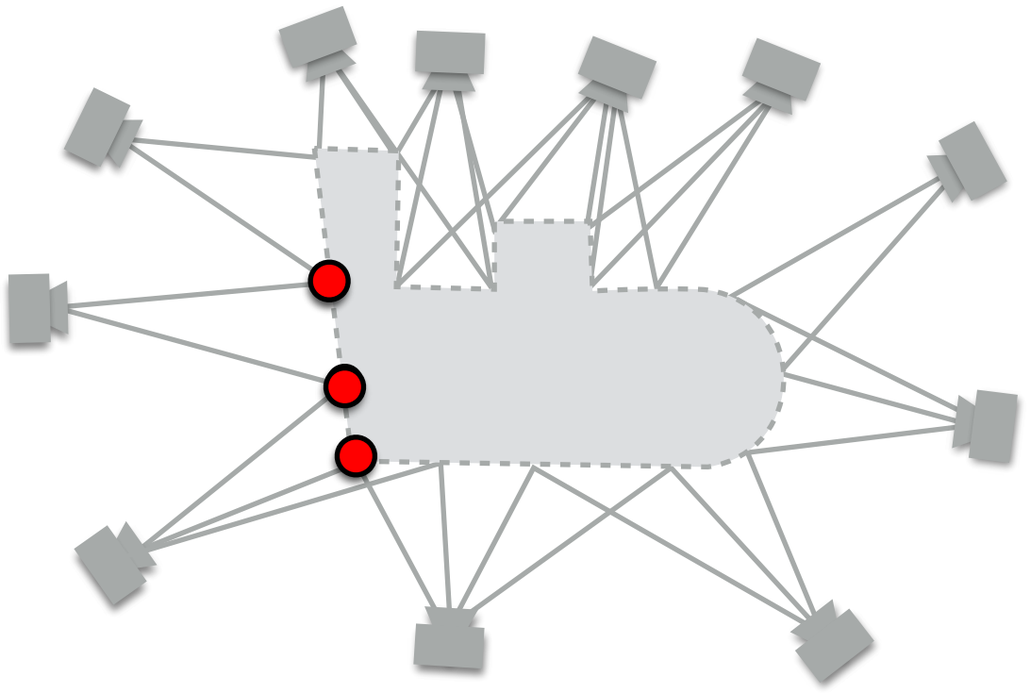


- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

Sampling With Co-occurrence Priors

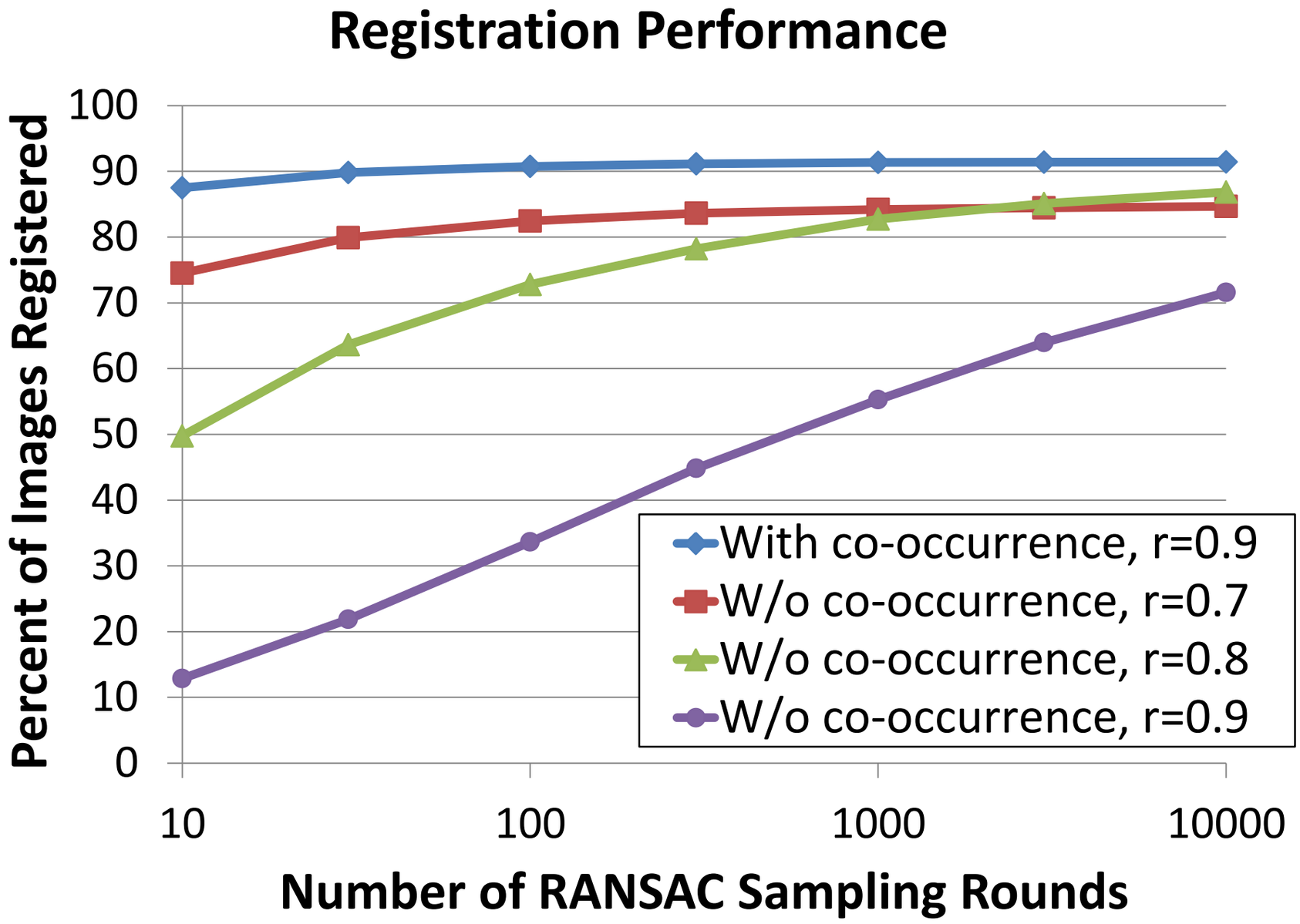
Idea: Increase ratio test threshold, shift responsibility to RANSAC



- **Sampling with Co-occurrence Priors** [Li et al., ECCV'12]
  - A_i = {database images observing point p_i}
  - Select first match uniformly at random
  - Select i^{th} sample match with probability distribution

$$\Pr\left[p_i \mid p_1, \ldots, p_{i-1}\right] \propto \left|A_1 \cap \cdots \cap A_i\right|$$

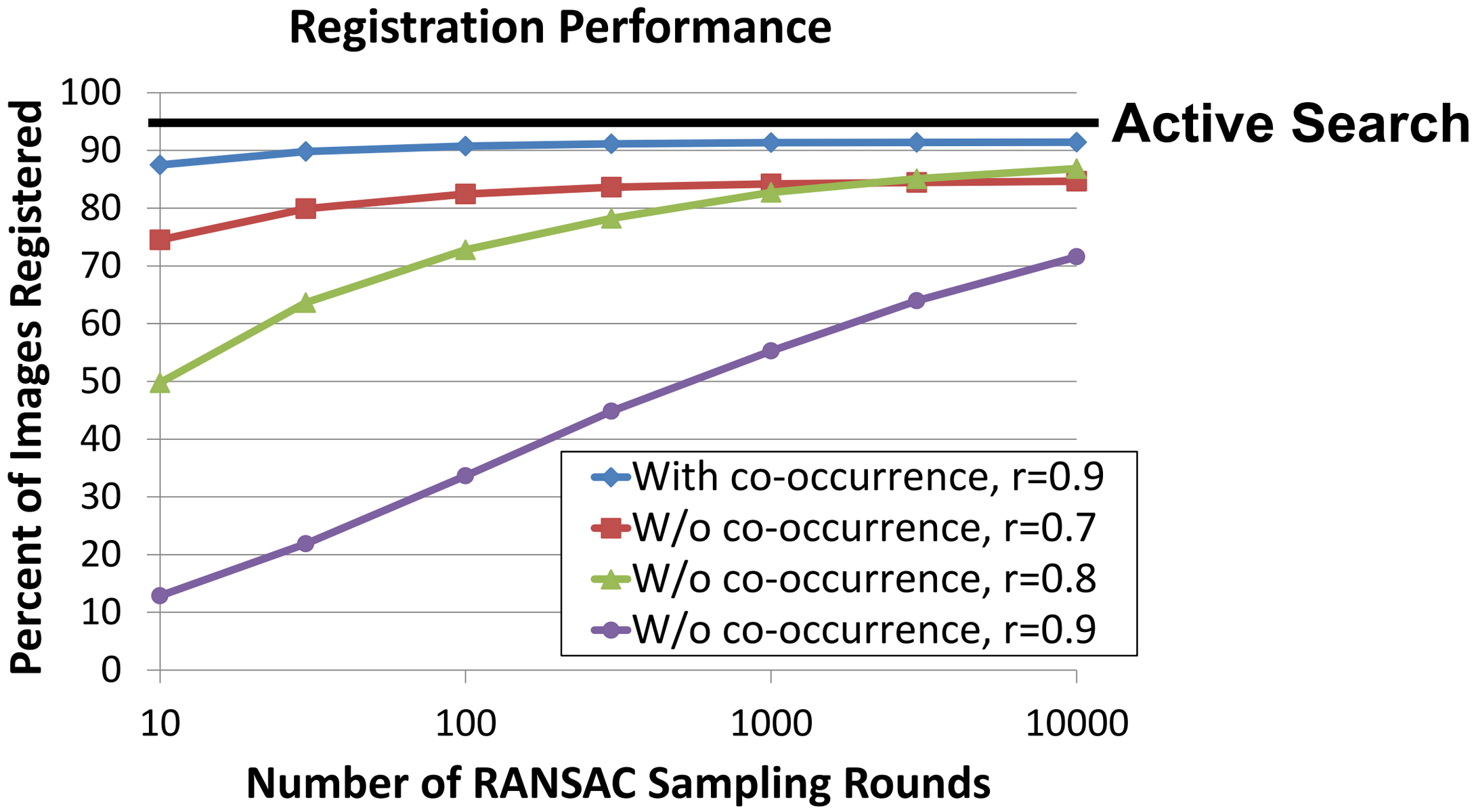
Results



2D-to-3D matching using kd-tree search and ratio test threshold `r`, Landmarks 1k dataset

[Li et al., ECCV'12]

Results



2D-to-3D matching using kd-tree search and ratio test threshold `r`, Landmarks 1k dataset

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

Camera Pose Estimation: RANSAC + P3P / P4Pf + co-occurrence sampling

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

Camera Pose Estimation: RANSAC + P3P / P4Pf + co-occurrence sampling

If no pose estimated

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

Camera Pose Estimation: RANSAC + P3P / P4Pf + co-occurrence sampling

If no pose estimated

Use 2D-to-3D matches with small threshold value as seeds

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

Camera Pose Estimation: RANSAC + P3P / P4Pf + co-occurrence sampling

If no pose estimated

- Use 2D-to-3D matches with small threshold value as seeds
- Run **P2F** using seeds

[Li et al., ECCV'12]

Worldwide Pose Estimation (WPE)

Find 2D-to-3D matches using **kd-tree**, ratio threshold 0.9

Camera Pose Estimation: RANSAC + P3P / P4Pf + co-occurrence sampling

If no pose estimated

- Use 2D-to-3D matches with small threshold value as seeds
- Run **P2F** using seeds
- Re-try Camera Pose Estimation

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

✓State-of-the-art effectiveness

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

✓State-of-the-art effectiveness

✓Scales very well

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

✓State-of-the-art effectiveness

✓Scales very well

✓Better handles repetitive structures than Active Search

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

✓State-of-the-art effectiveness

✓Scales very well

✓Better handles repetitive structures than Active Search

✗Significantly slower than Active Search due to kd-tree

[Li et al., ECCV'12]

Results

	Dubrovnik	Rome	Landmarks 1k
WPE	**100 %**	**99.7 %**	**98.95 %**
Active Search	99.5 %	99.0 %	95.35 %

	SF-0	SF-1
WPE	**54.2 %**	**62.5 %**
[Chen et al., CVPR'11], No GPS	20 %	41 %
[Chen et al., CVPR'11], With GPS		49 %

✓State-of-the-art effectiveness

✓Scales very well

✓Better handles repetitive structures than Active Search

✗Significantly slower than Active Search due to kd-tree

✗Still high memory consumption

[Li et al., ECCV'12]

Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

- Why not use place recognition / image retrieval?

Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

- Why not use place recognition / image retrieval?

  ✓No ratio test required during voting!

Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

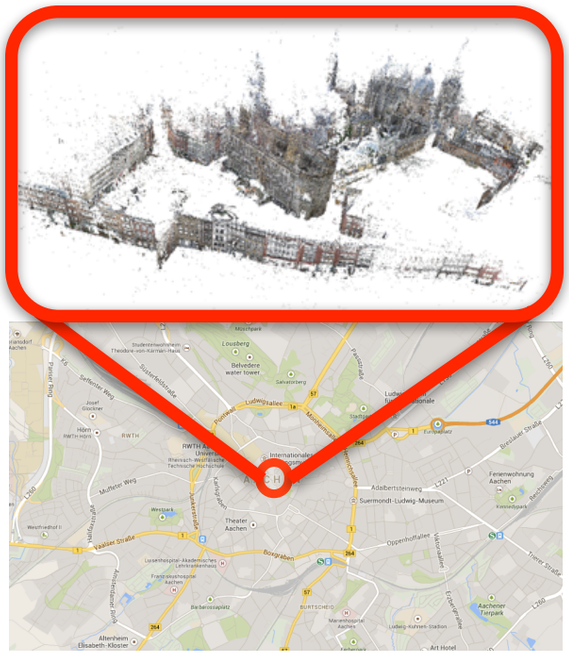
- Why not use place recognition / image retrieval?
  - ✓No ratio test required during voting!
  - ✓Ratio test safe to use during 2D-to-2D feature matching

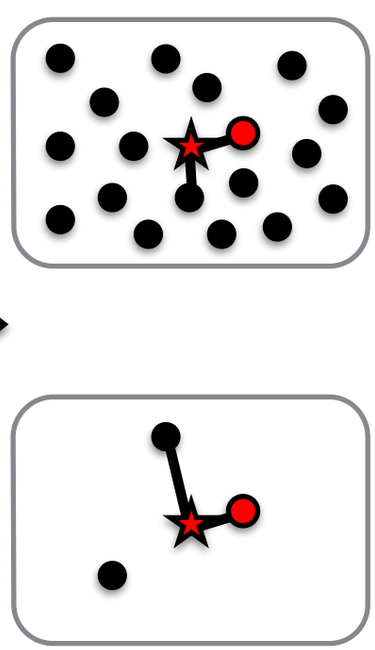
Alternative Approaches

- Pose priors available (GPS, WiFi, …)? Use Active Search to localize against sub model → Much more efficient

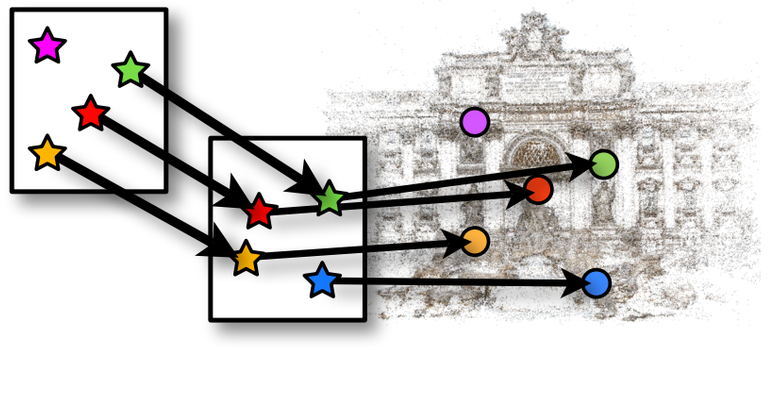
- Why not use place recognition / image retrieval?
  - ✓No ratio test required during voting!
  - ✓Ratio test safe to use during 2D-to-2D feature matching

Scalable Image-Based Localization









Memory Consumption Problem

Ratio Test Problem

Place Recognition for Localization

Place Recognition For Localization

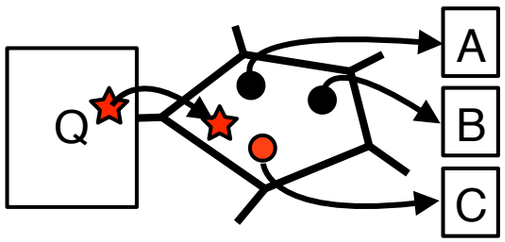
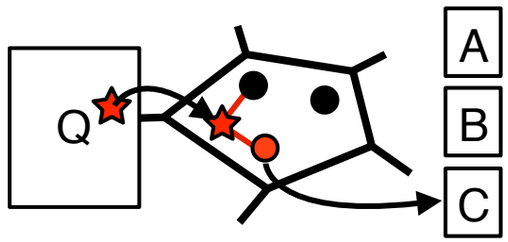


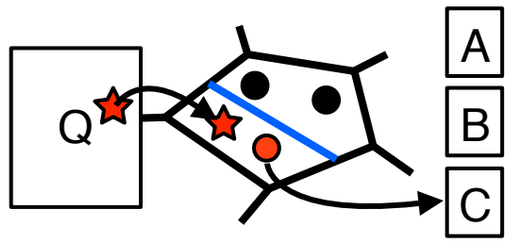
Image Retrieval

✗ images
✓ memory efficient
✓ no ratio test



Correspondence Voting

✓ images
✗ memory efficient
✗ no ratio test

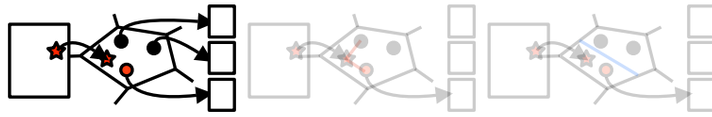


Hamming Voting

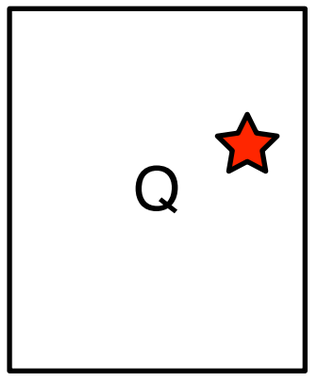
✓ images
✓ memory efficient
✓ no ratio test

[Sattler et al., BMVC'12]

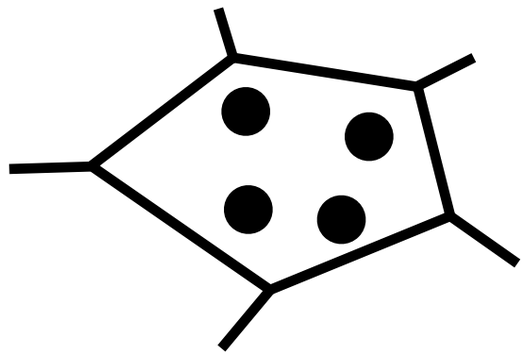
Image Retrieval



Query Image

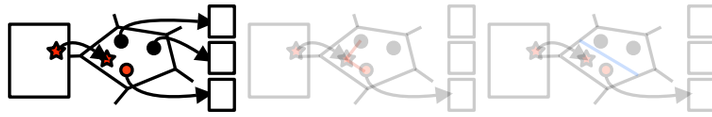


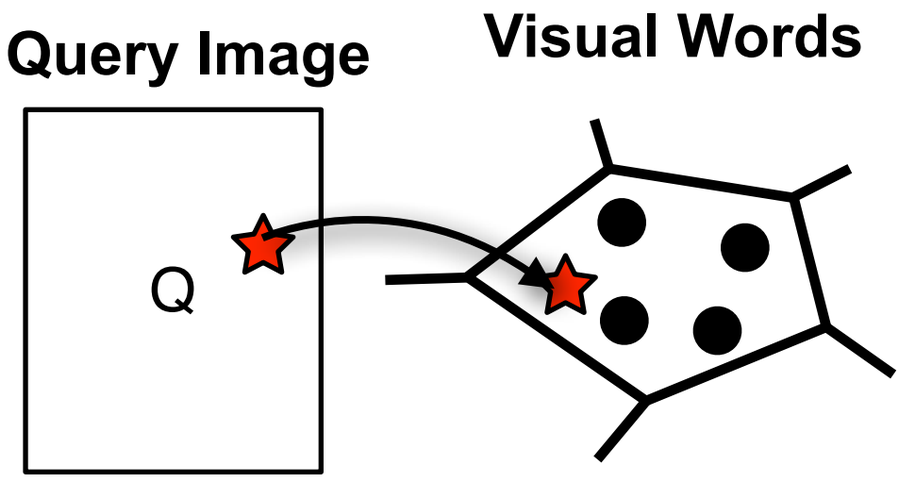
Visual Words



[Irschara et al., CVPR'09]

Image Retrieval

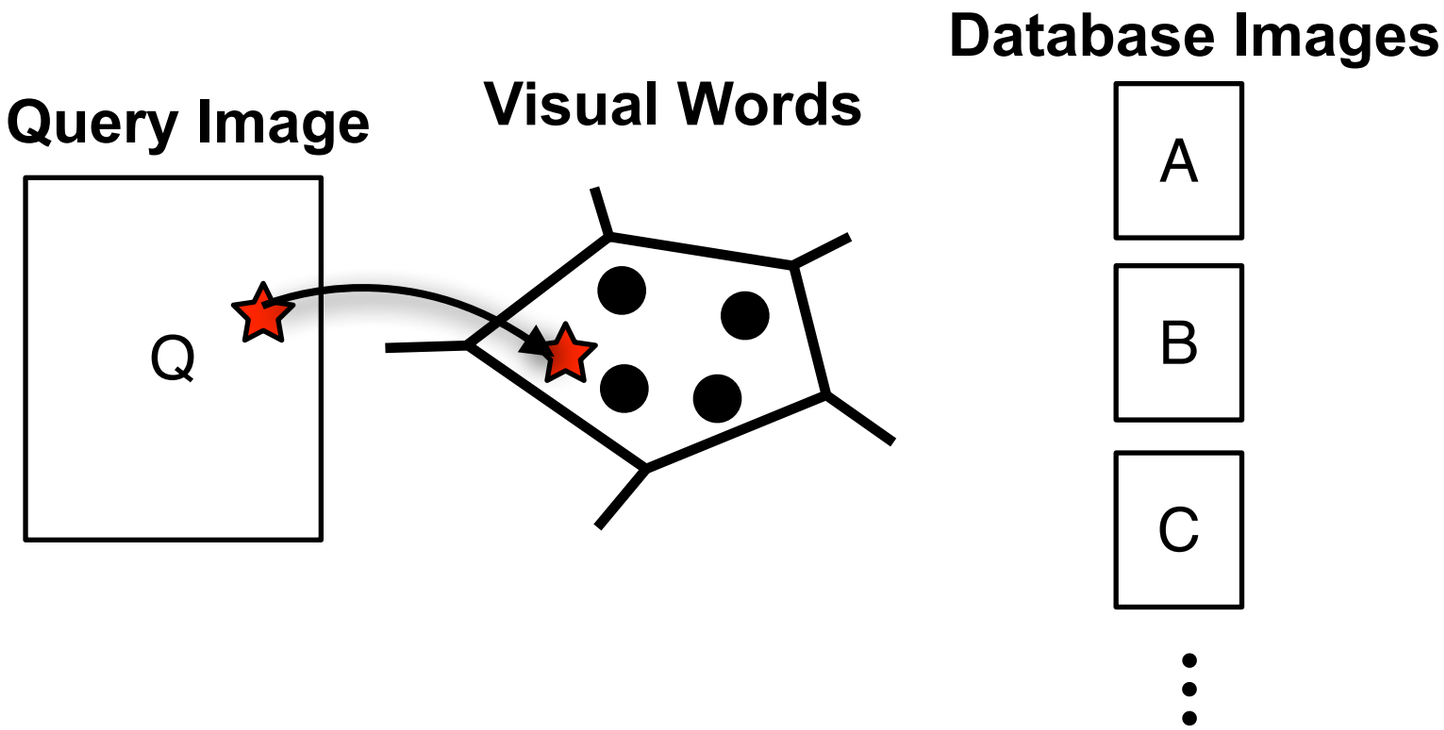




[Irschara et al., CVPR'09]

Image Retrieval

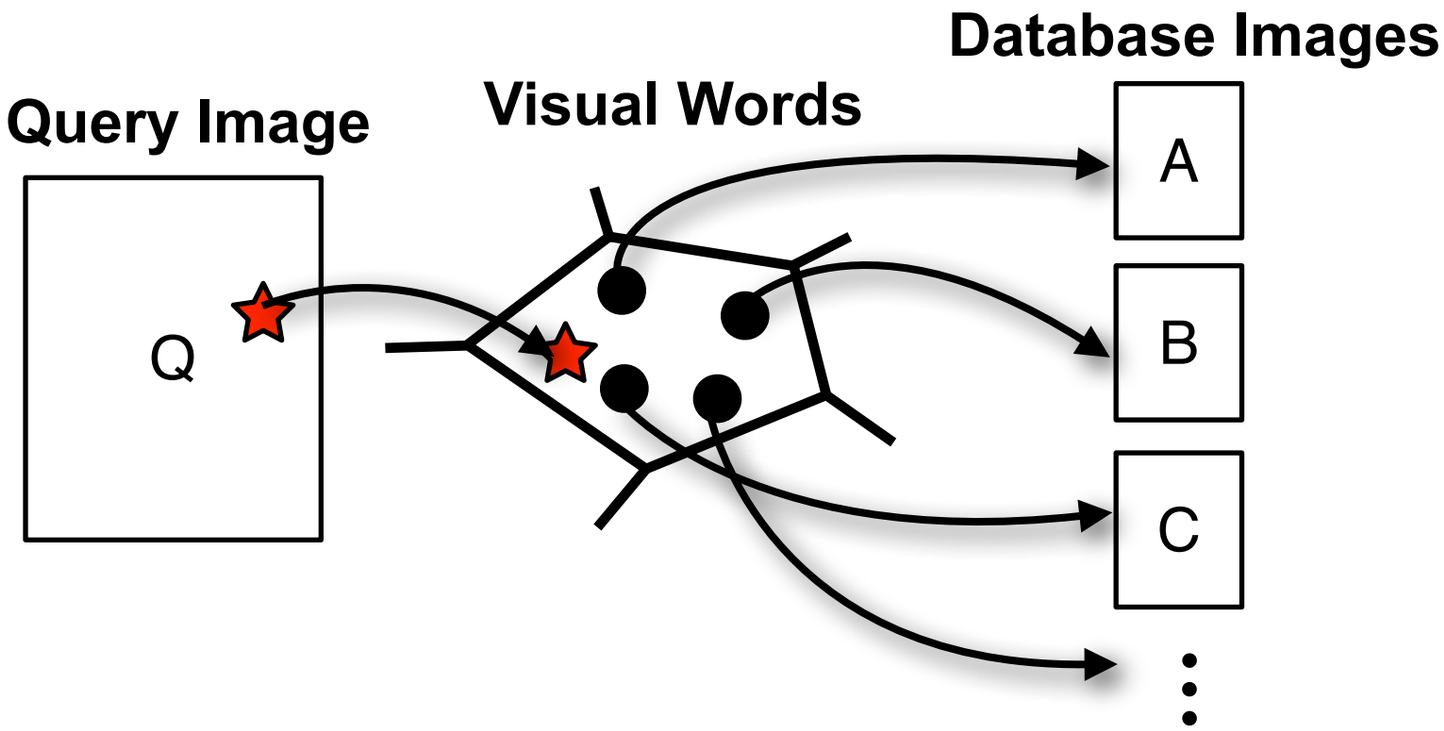




[Irschara et al., CVPR'09]

Image Retrieval

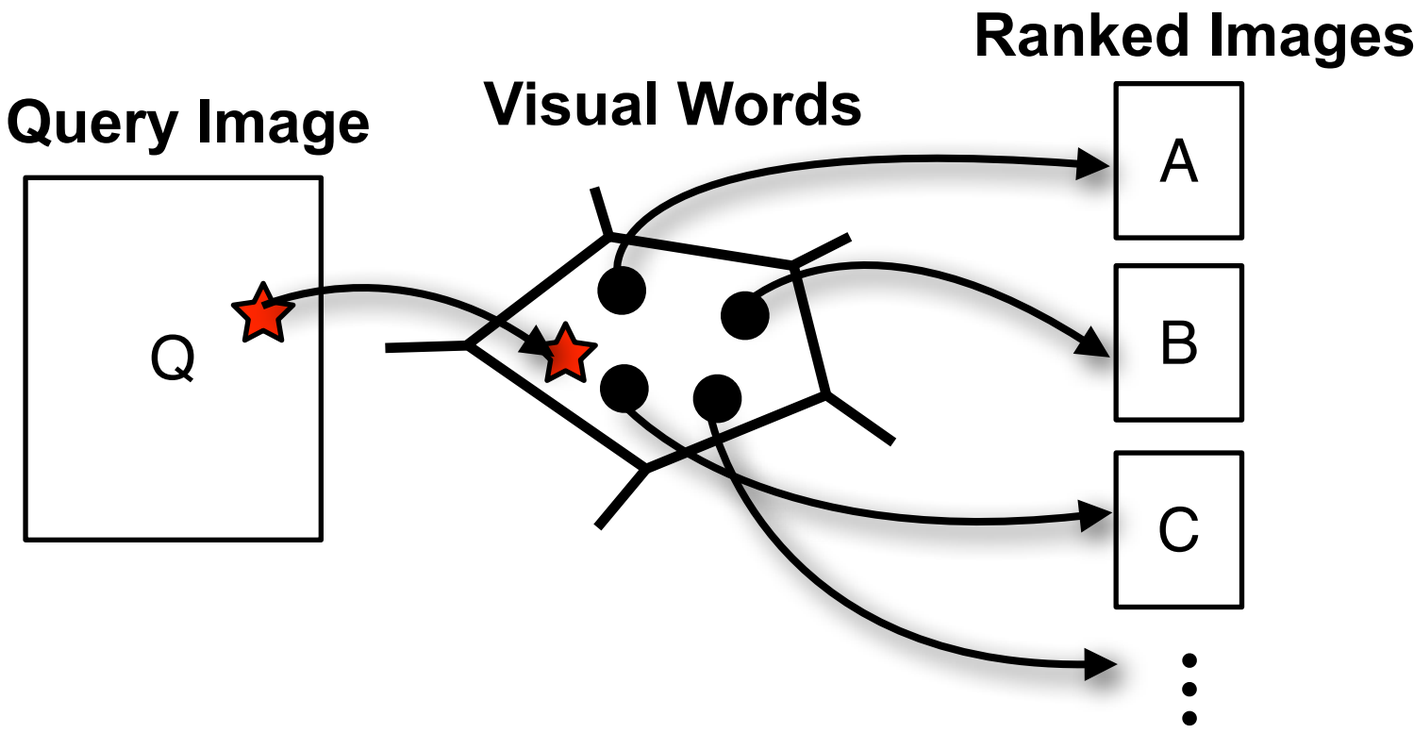




[Irschara et al., CVPR'09]

Image Retrieval

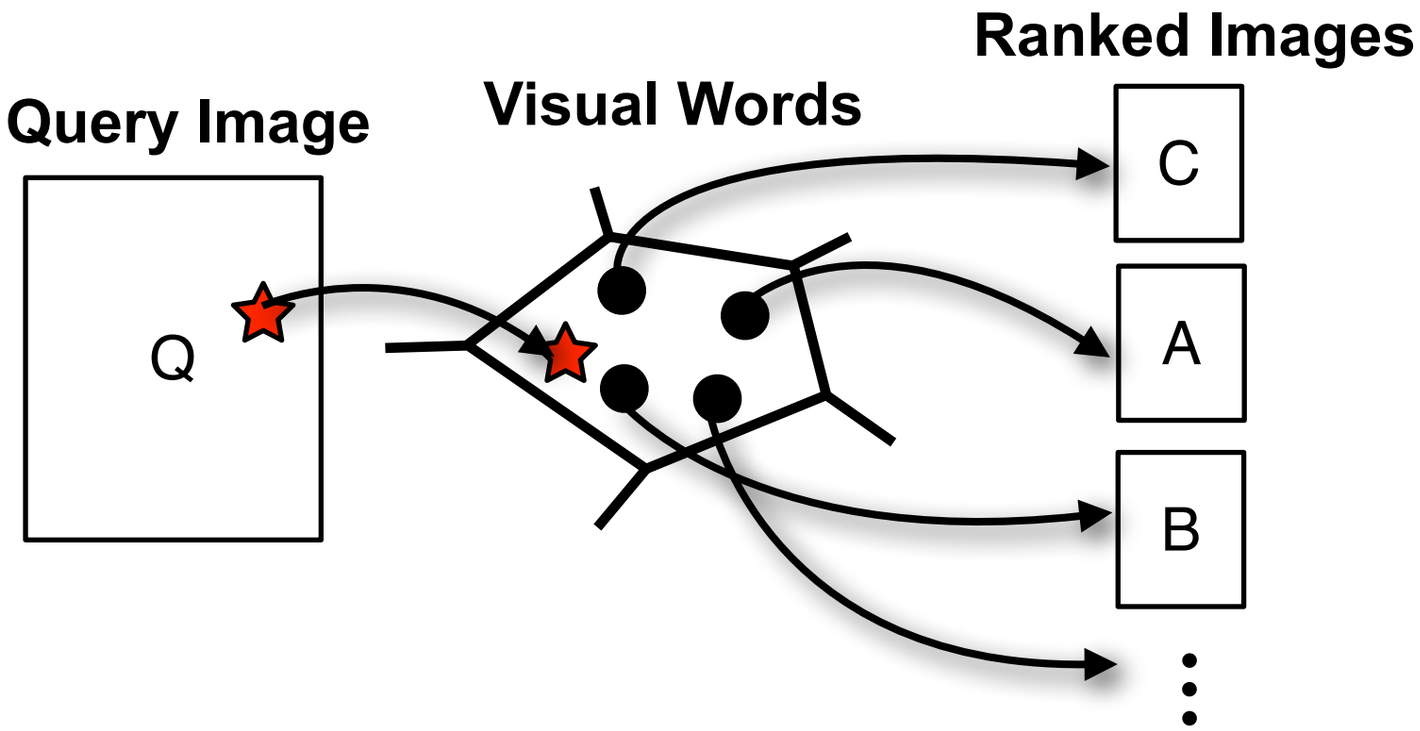




[Irschara et al., CVPR'09]

Image Retrieval

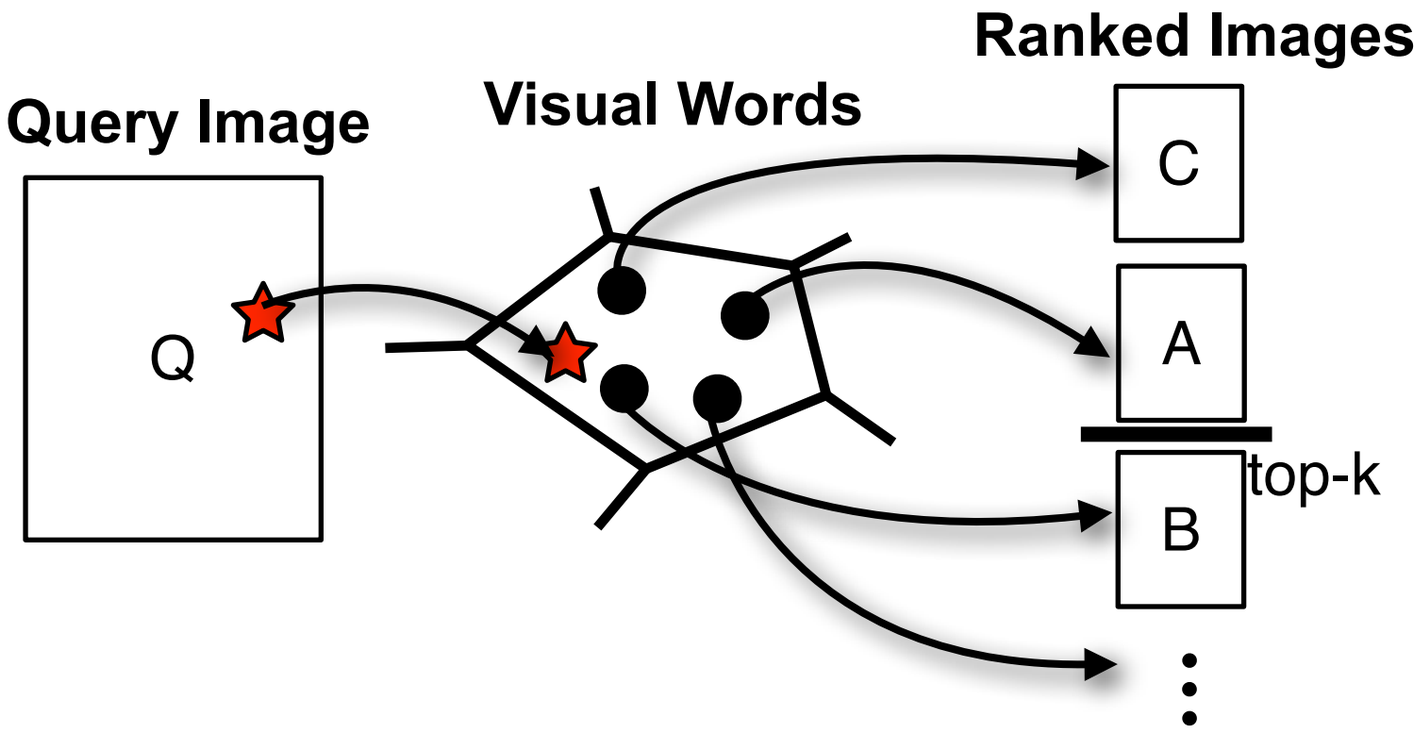




[Irschara et al., CVPR'09]

Image Retrieval

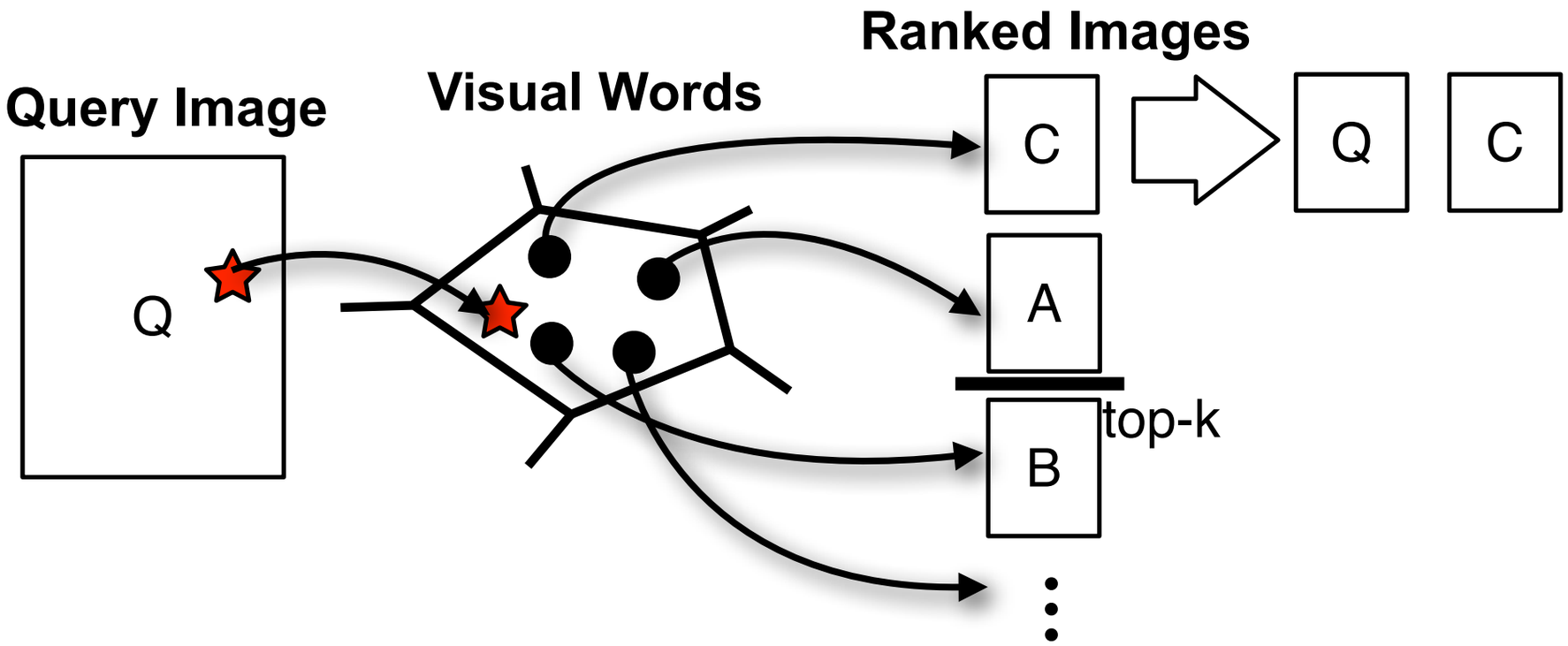




[Irschara et al., CVPR'09]

Image Retrieval

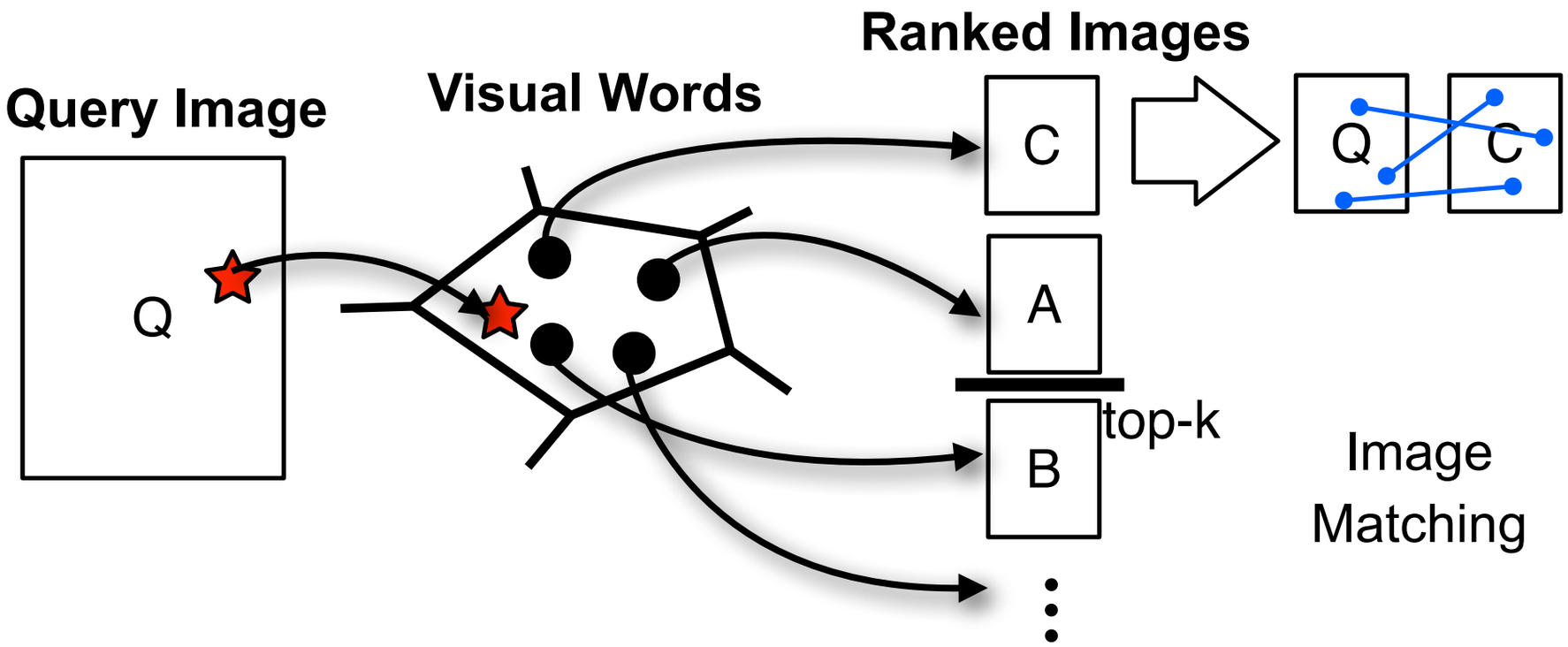




[Irschara et al., CVPR'09]

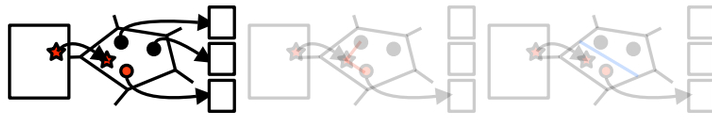
Image Retrieval

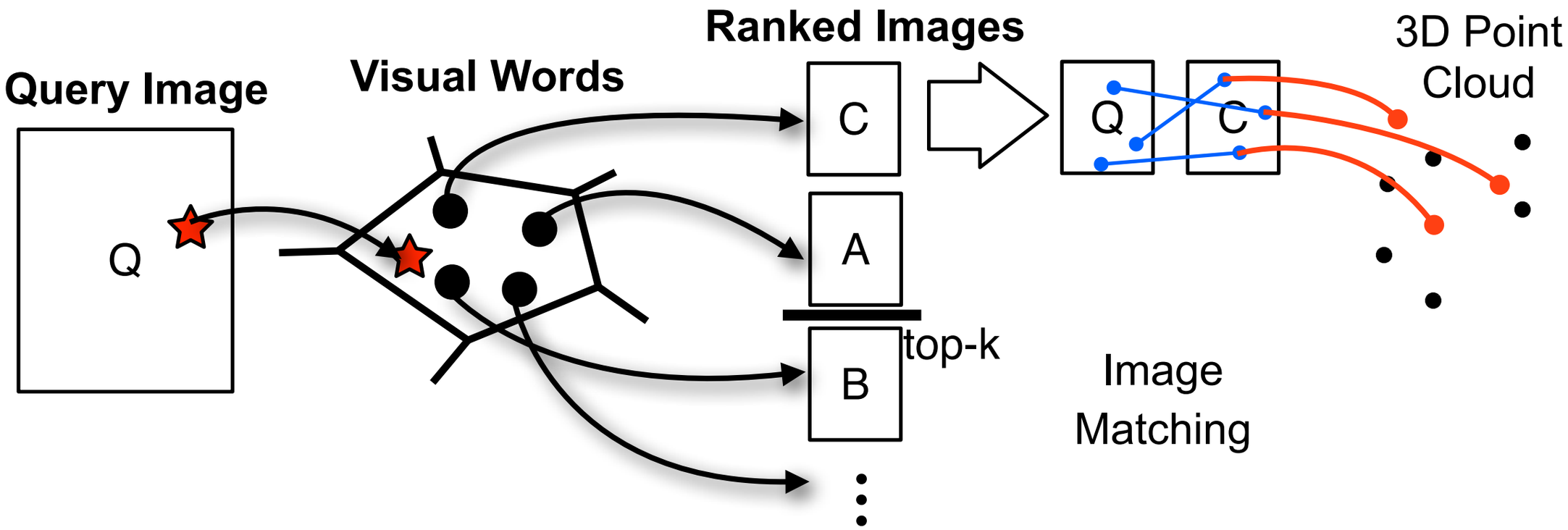




[Irschara et al., CVPR'09]

Image Retrieval

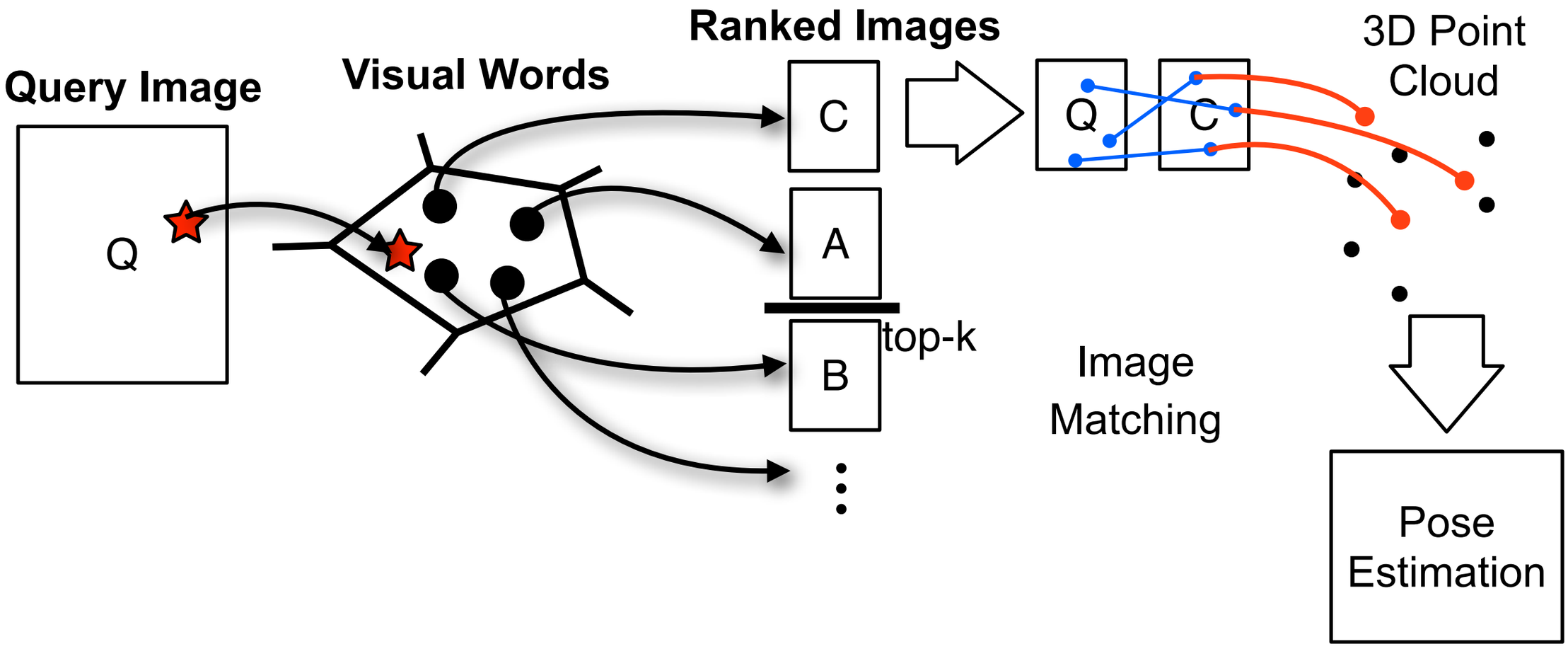




[Irschara et al., CVPR'09]

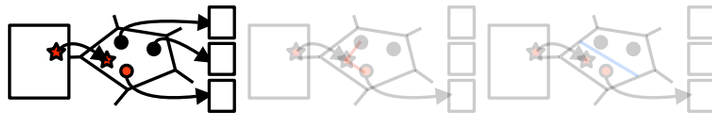
Image Retrieval

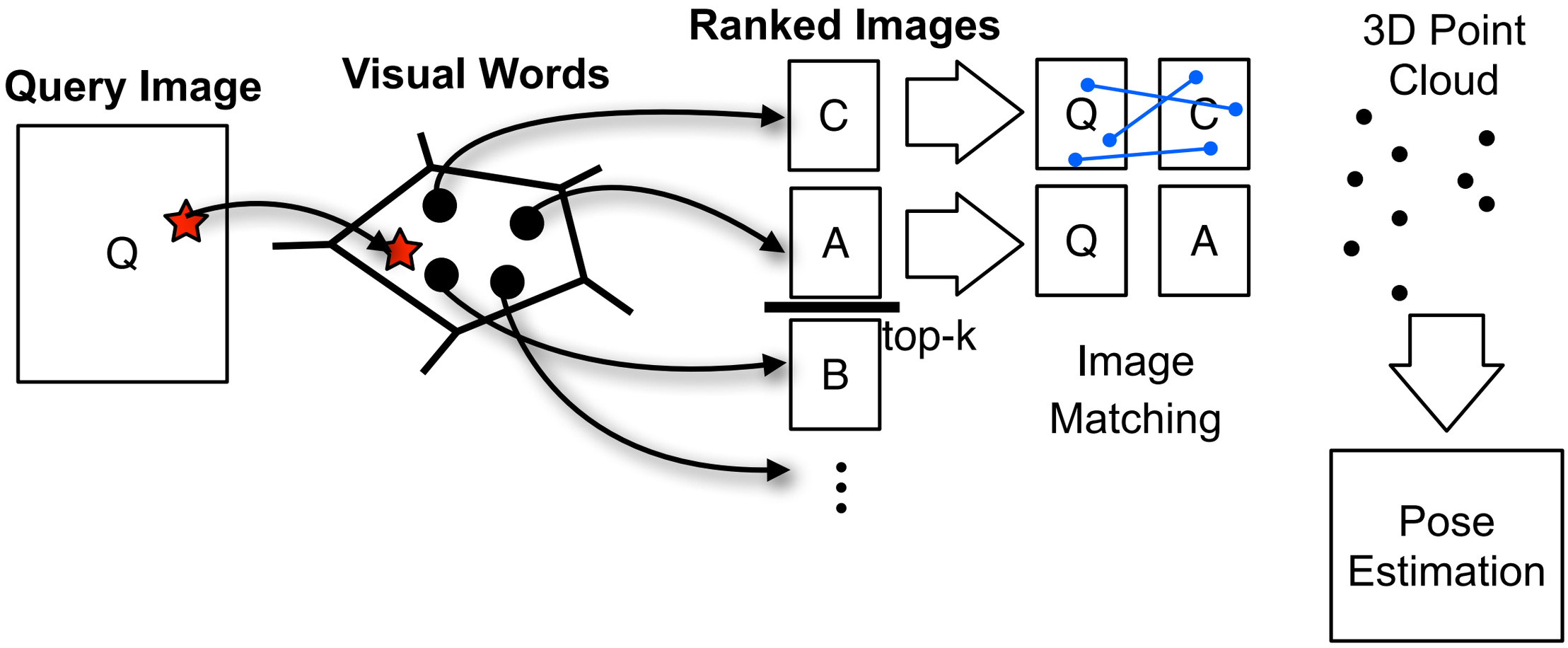




[Irschara et al., CVPR'09]

Image Retrieval

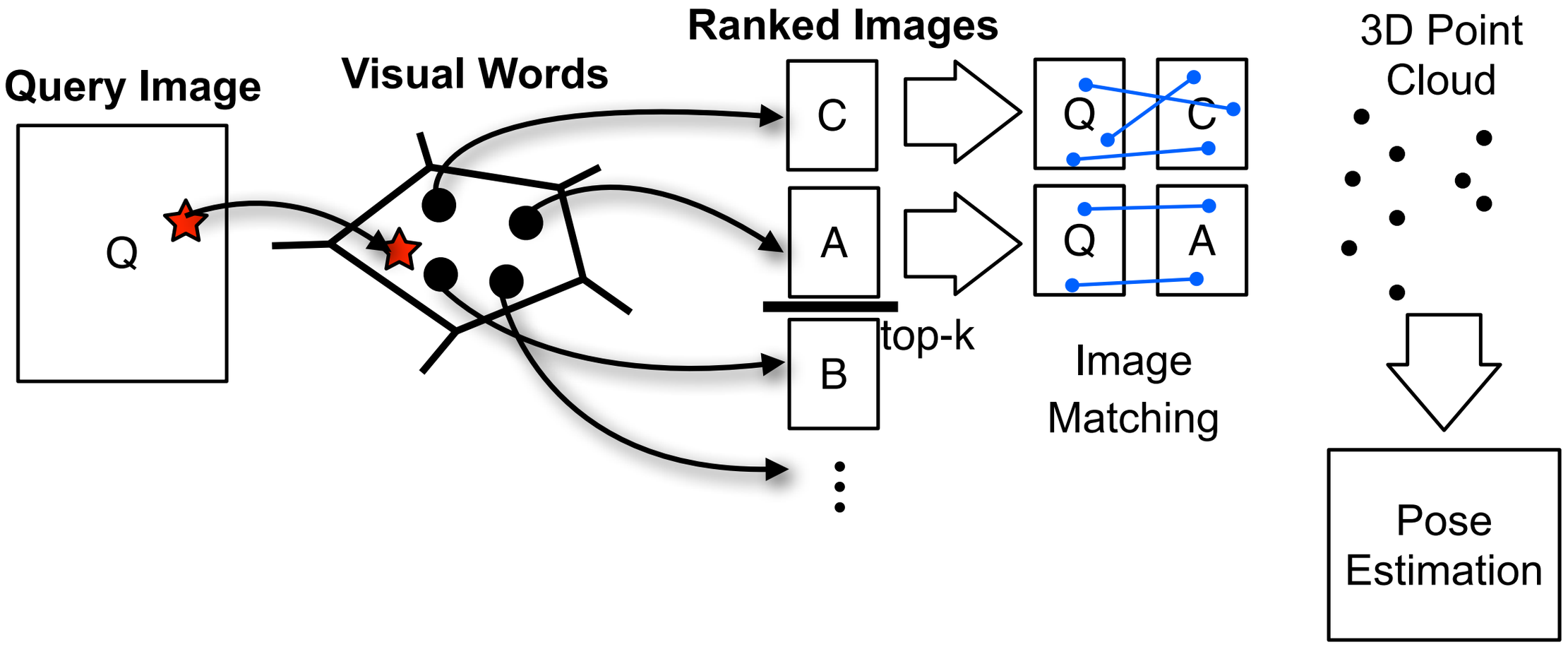




[Irschara et al., CVPR'09]

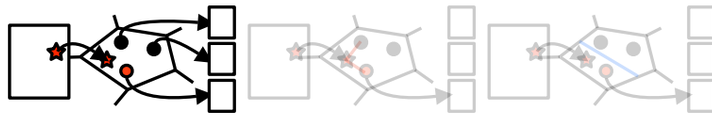
Image Retrieval

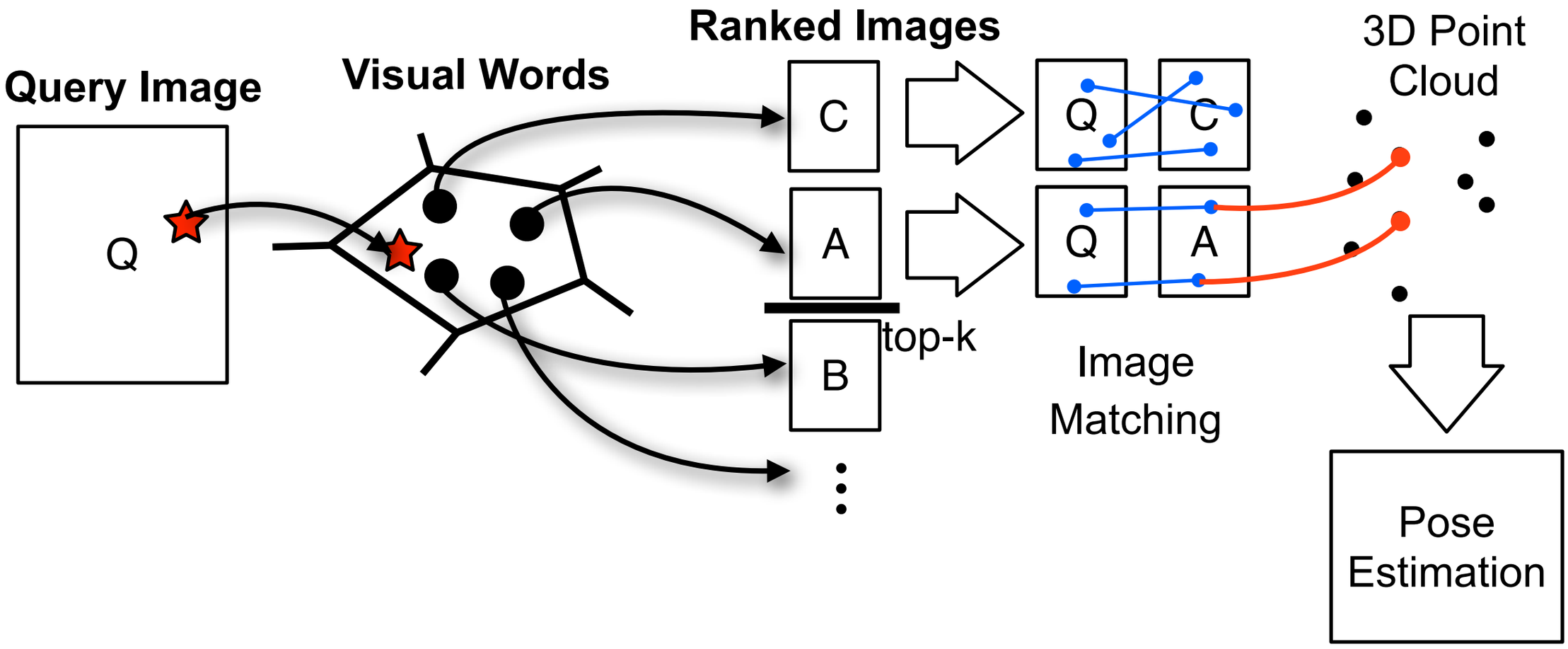




[Irschara et al., CVPR'09]

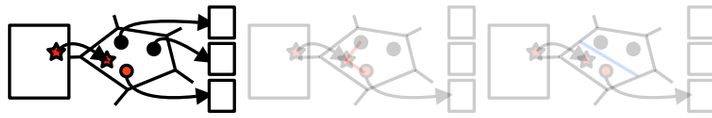
Image Retrieval

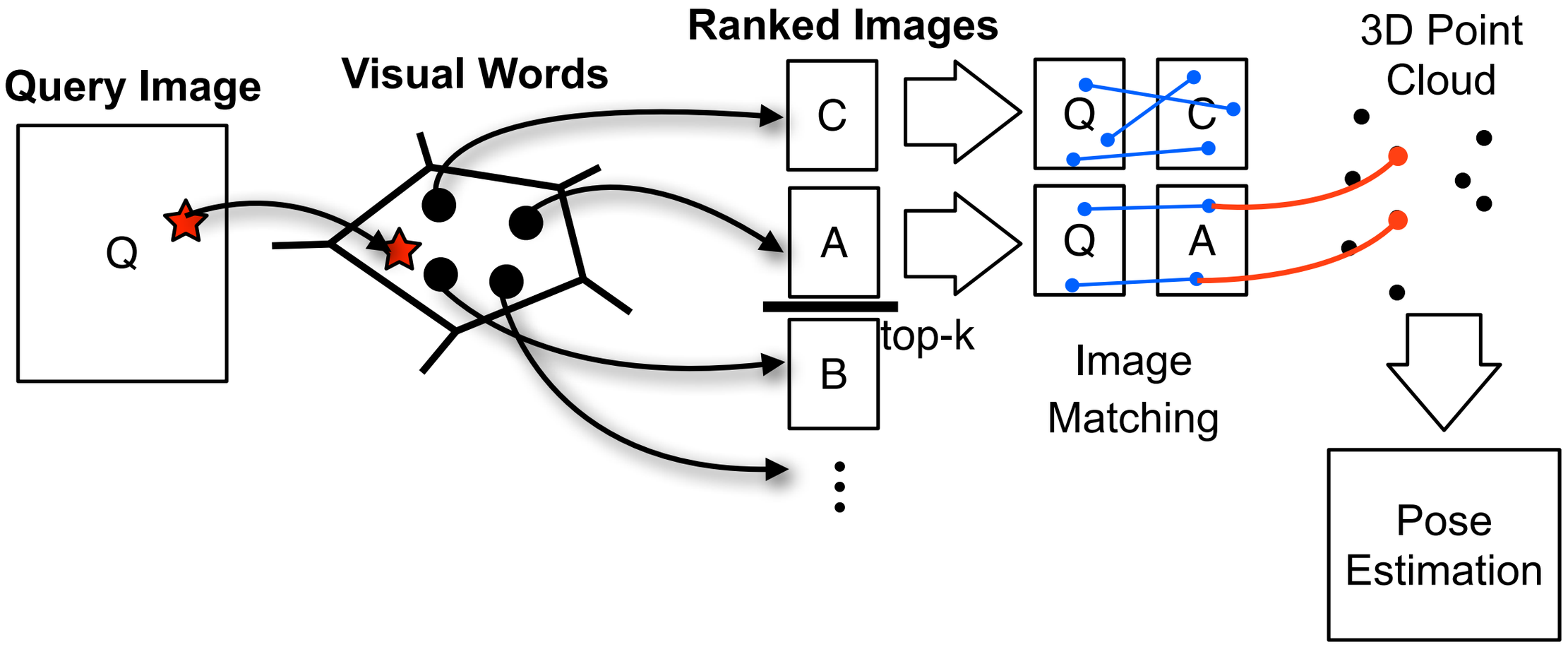




[Irschara et al., CVPR'09]

Image Retrieval



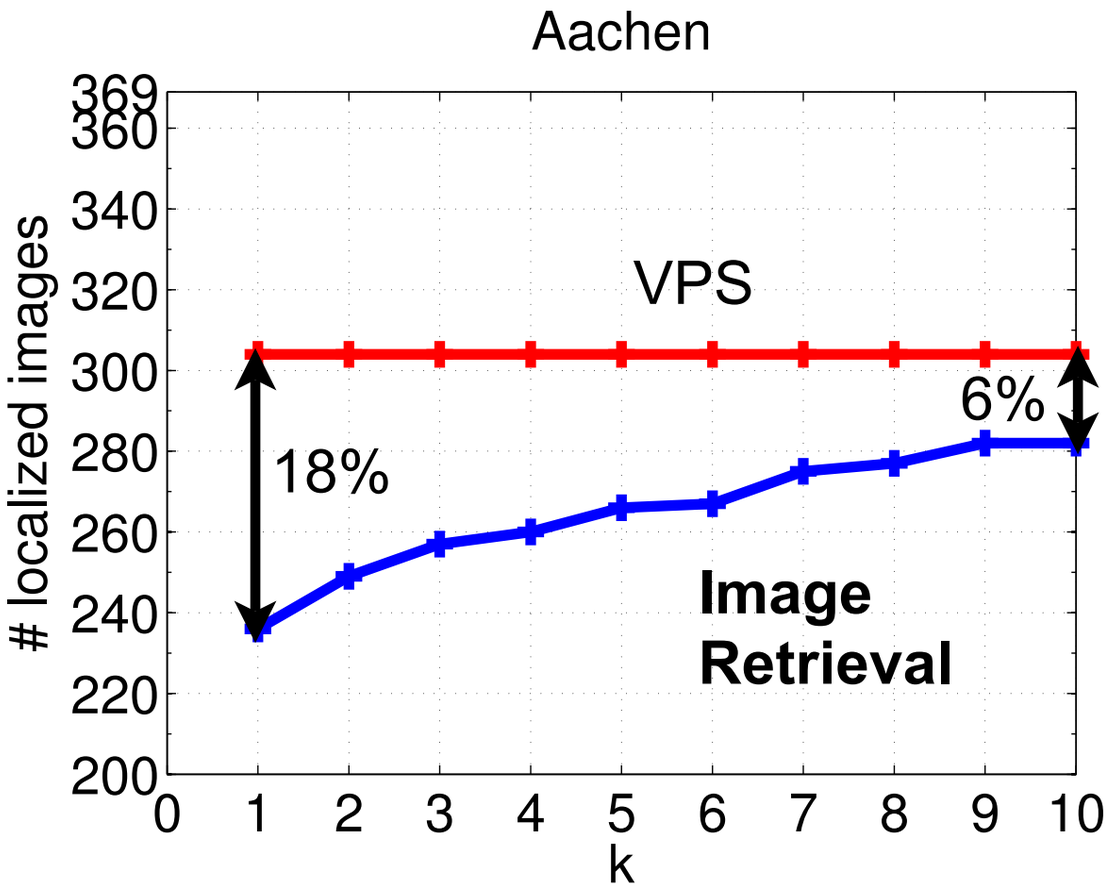


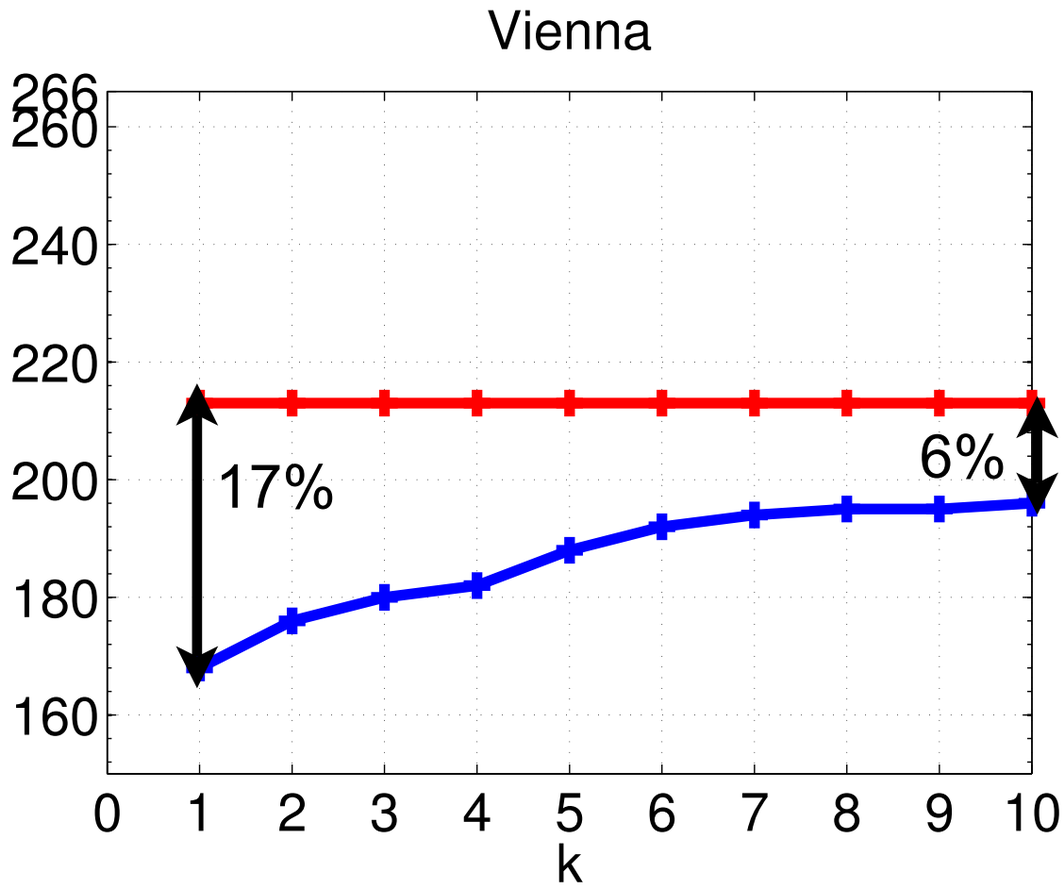
- Very similar pipeline to VPS but
  - ✓SIFT descriptors not required for voting
  - ✓ Ratio test only in sparser descriptor space of image features

[Irschara et al., CVPR'09]

Results



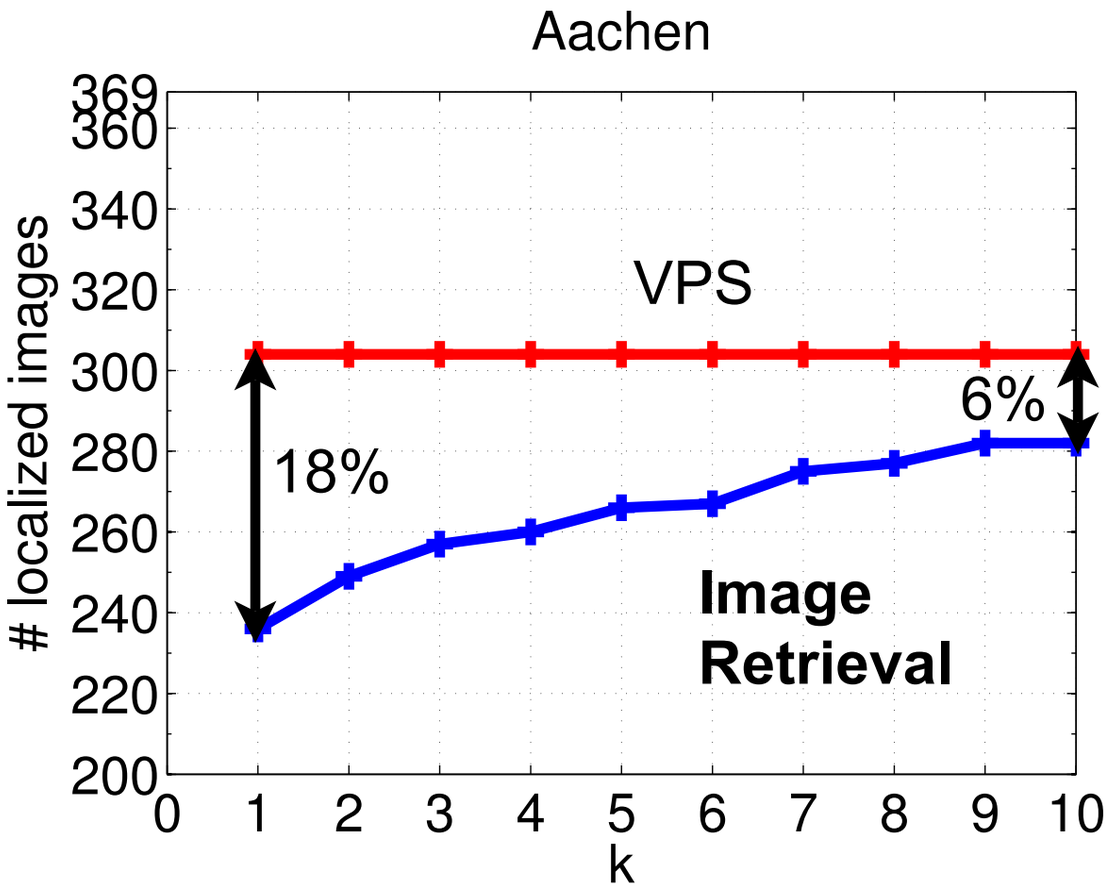


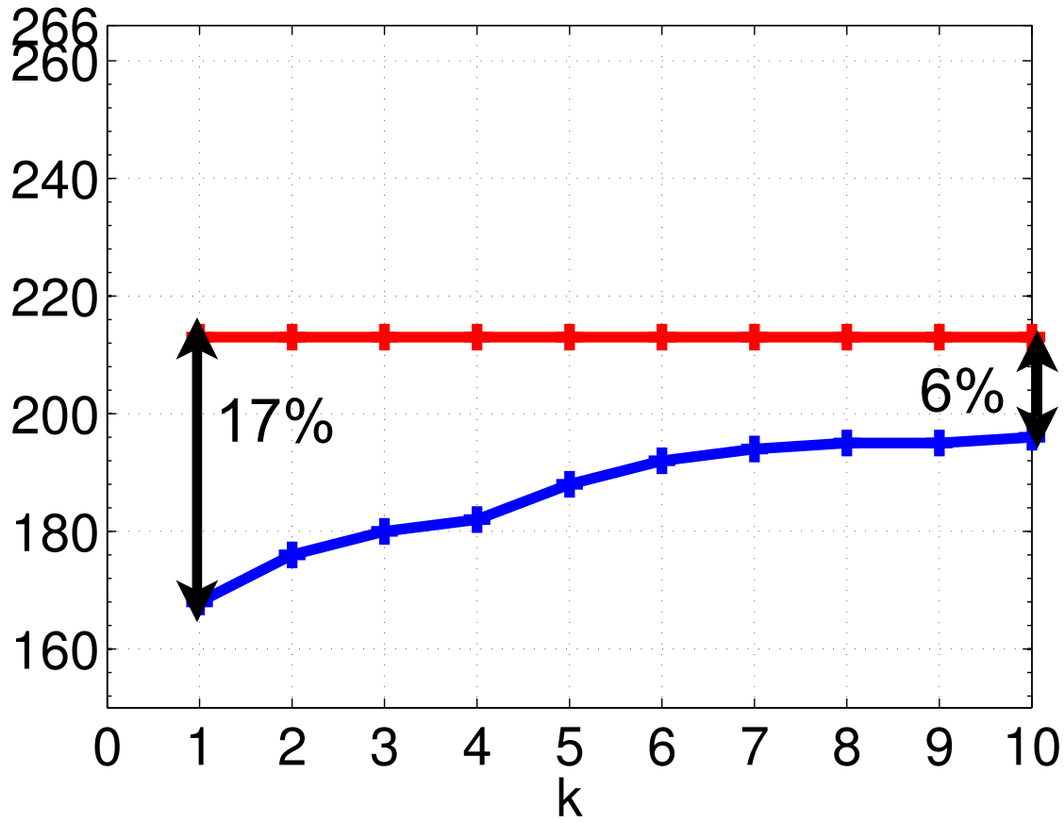


[Sattler et al., BMVC'12]

Results





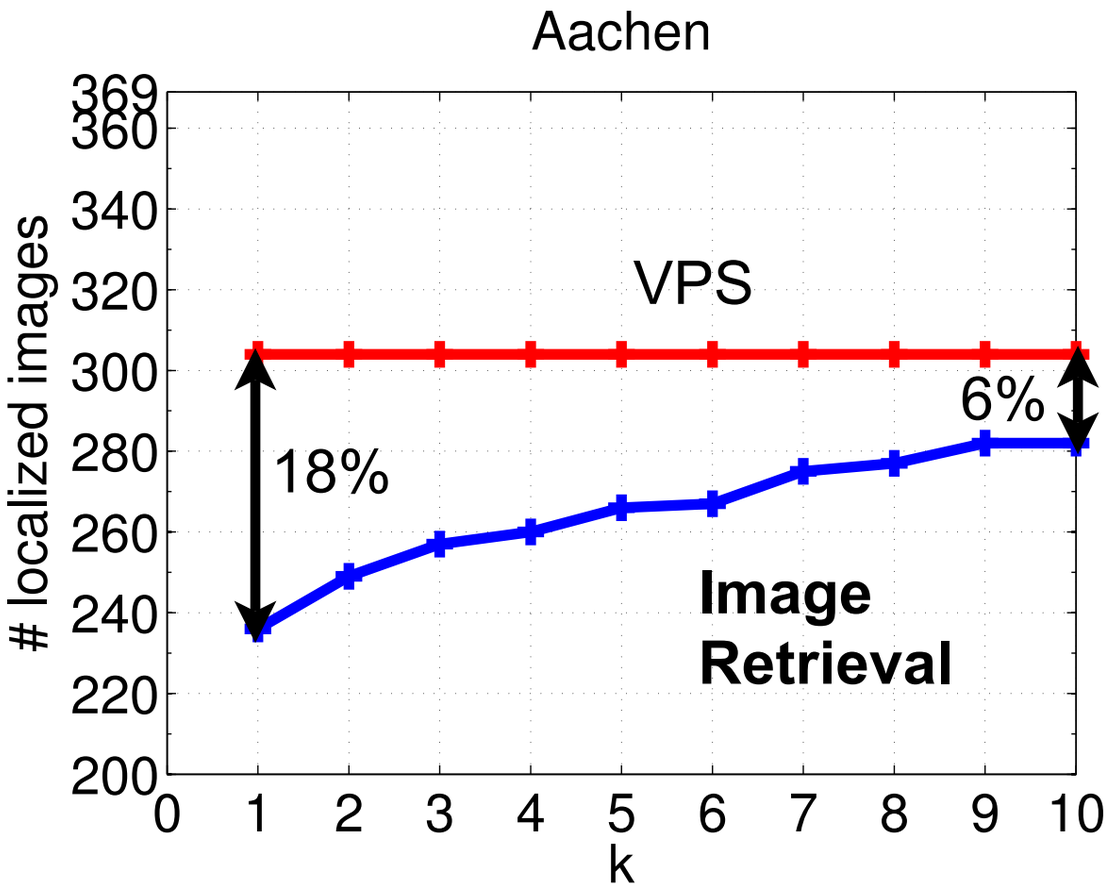


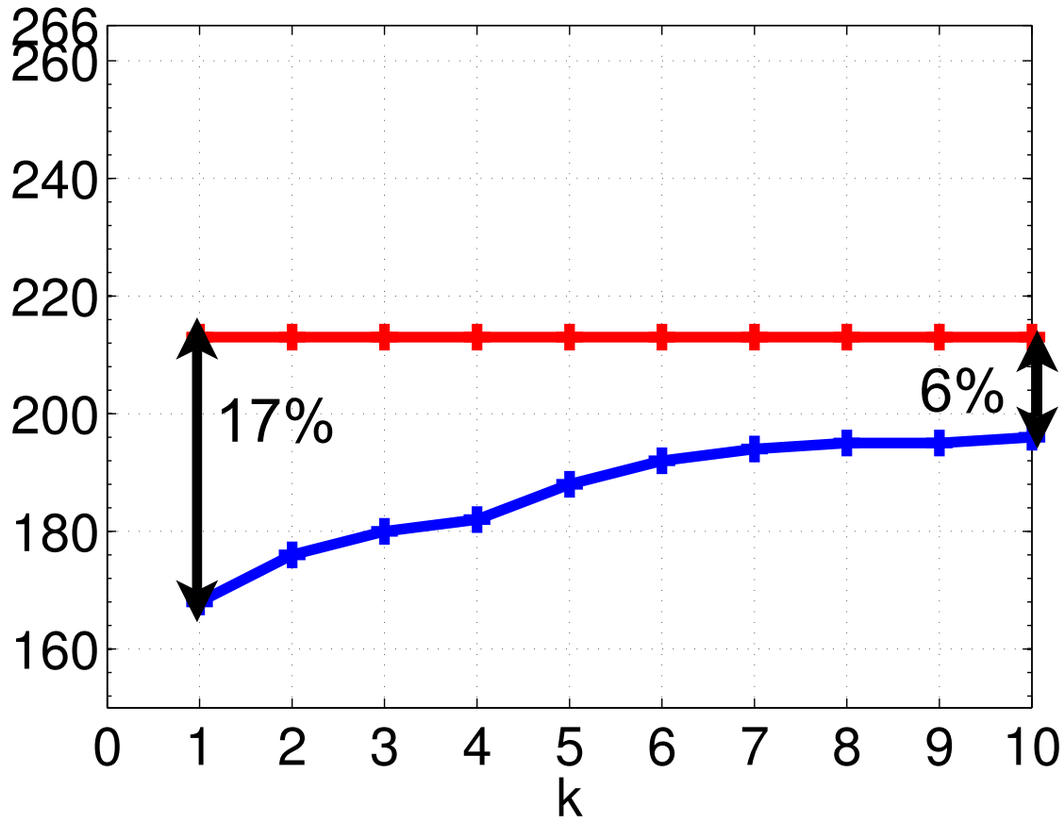
Two possible reasons:

[Sattler et al., BMVC'12]

Results

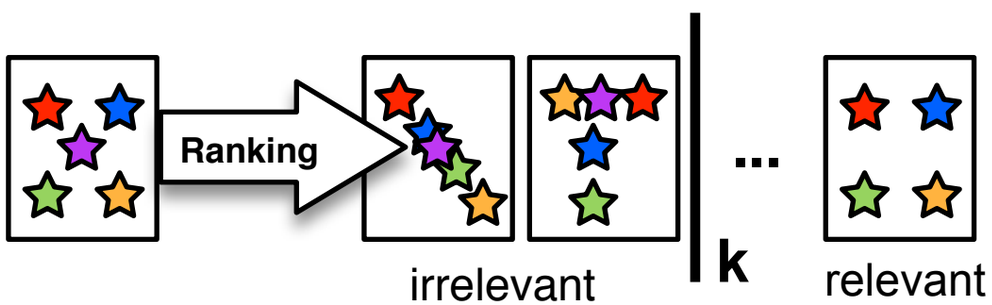






Two possible reasons:

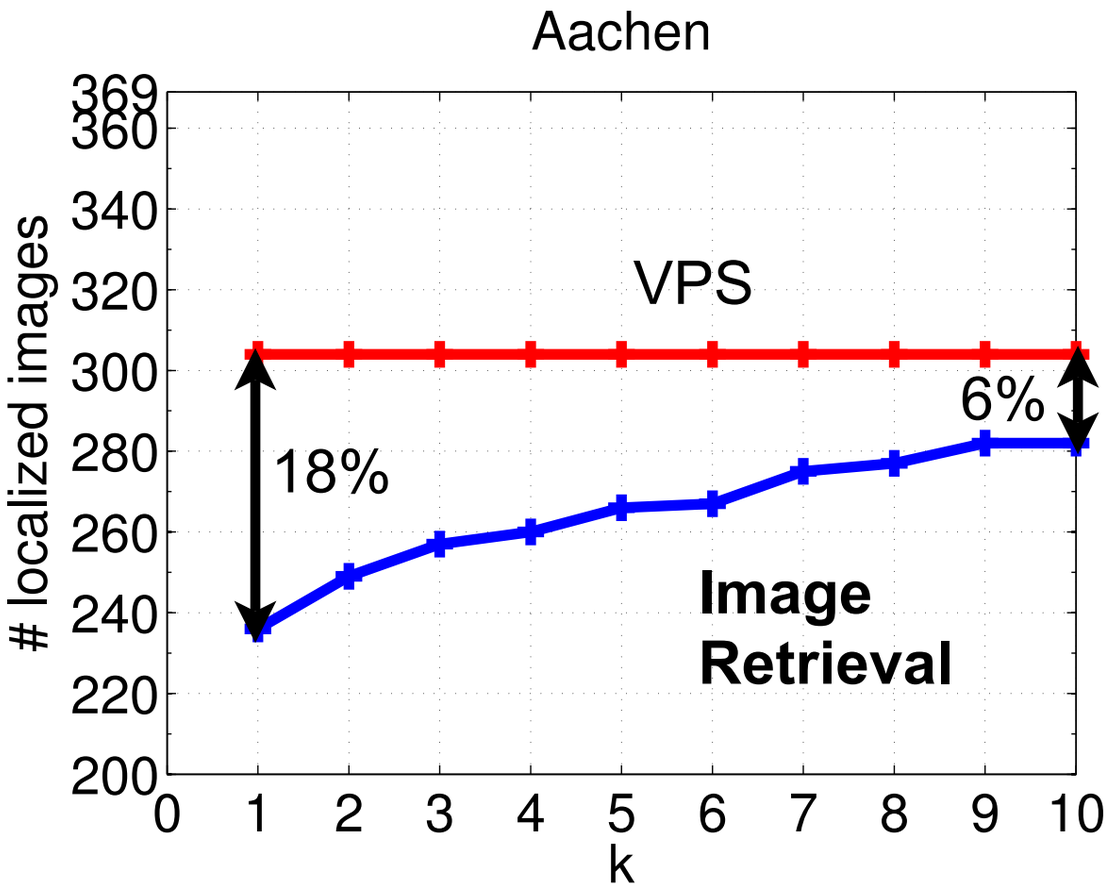
1. Polluted image ranking

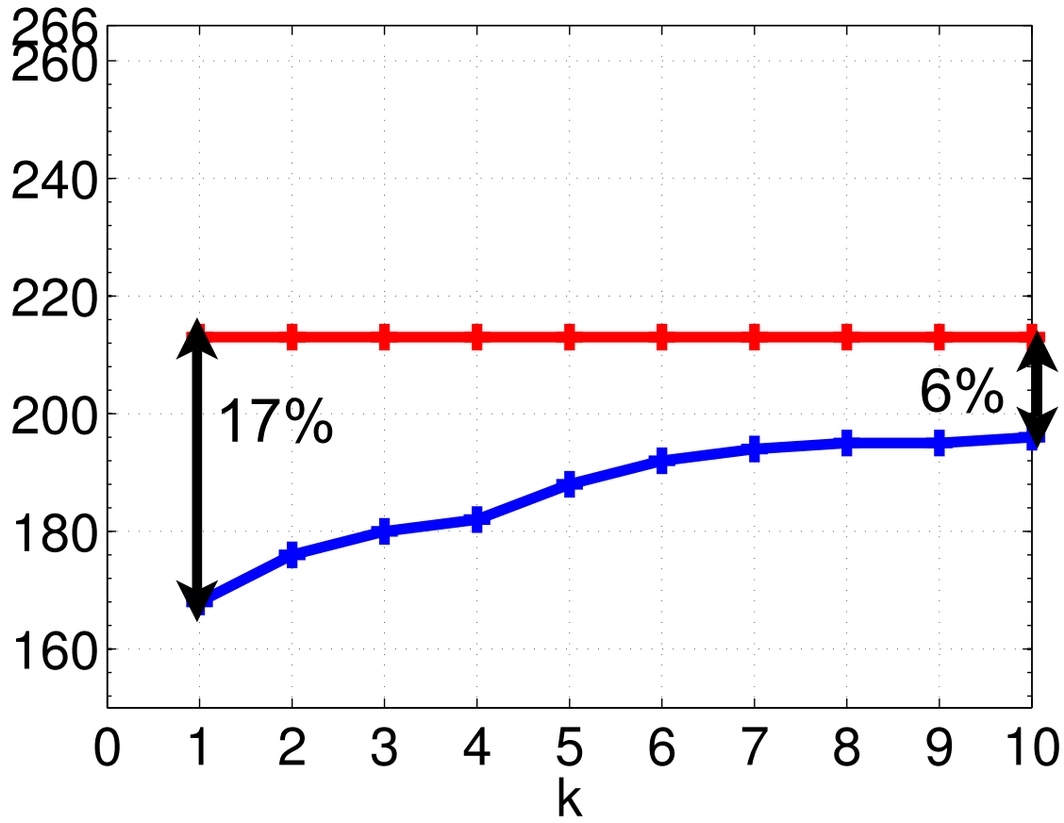


[Sattler et al., BMVC'12]

Results



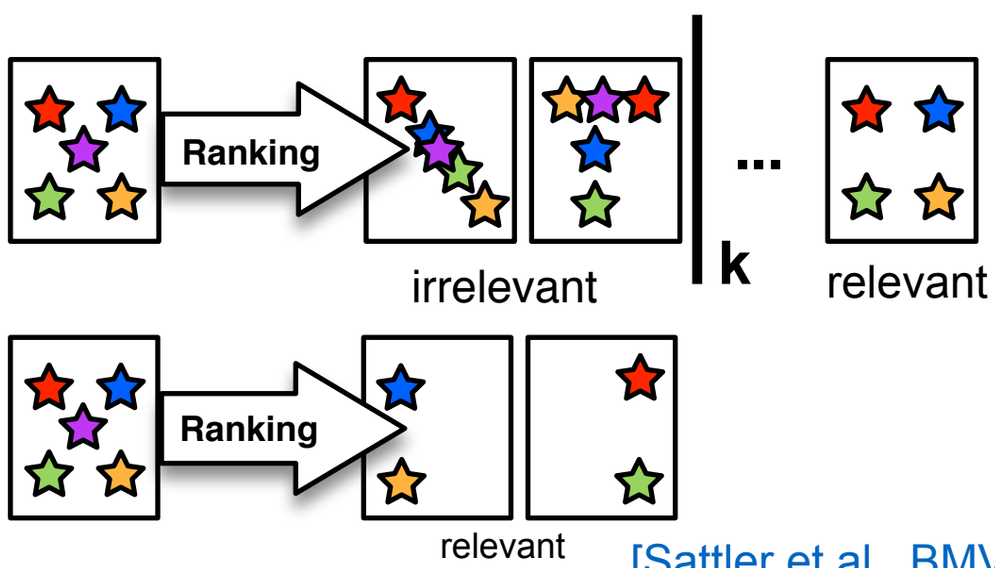




Two possible reasons:

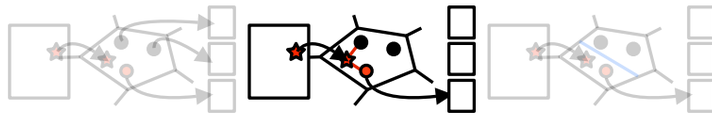
1. Polluted image ranking

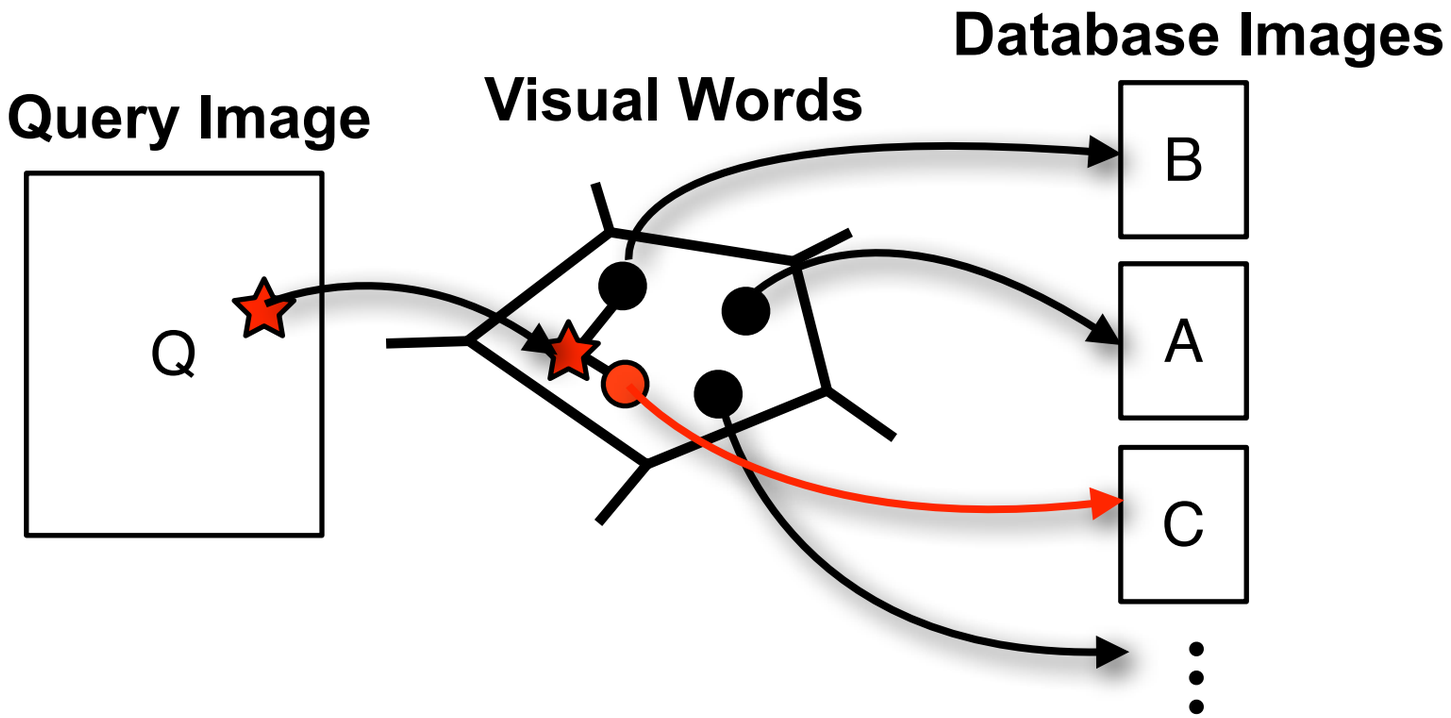
2. Not enough overlap



[Sattler et al., BMVC'12]

Correspondence Voting

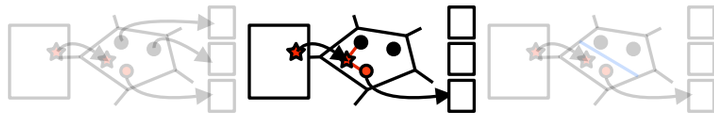


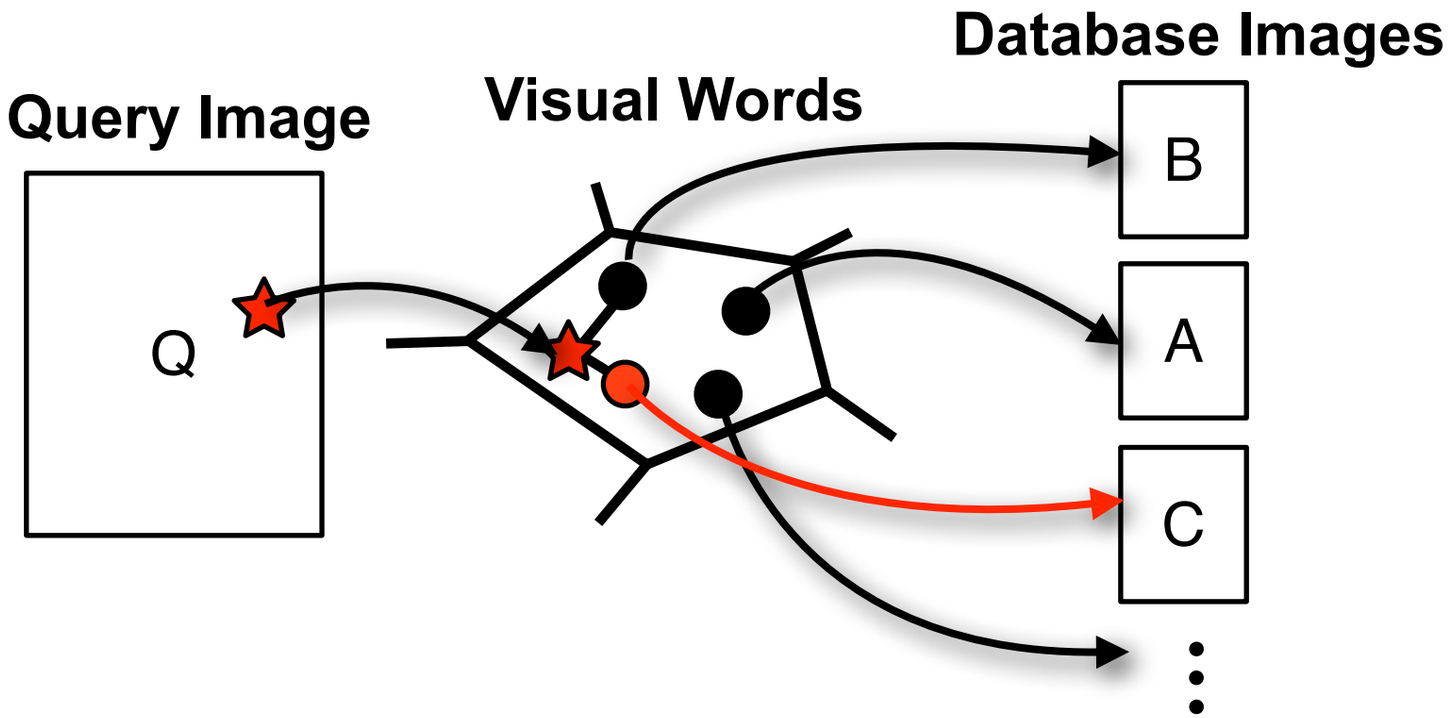


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**

[Sattler et al., BMVC'12]

Correspondence Voting

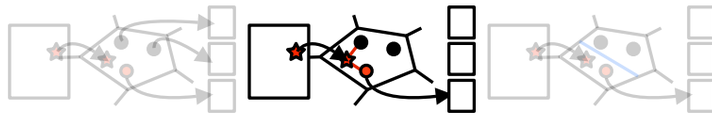


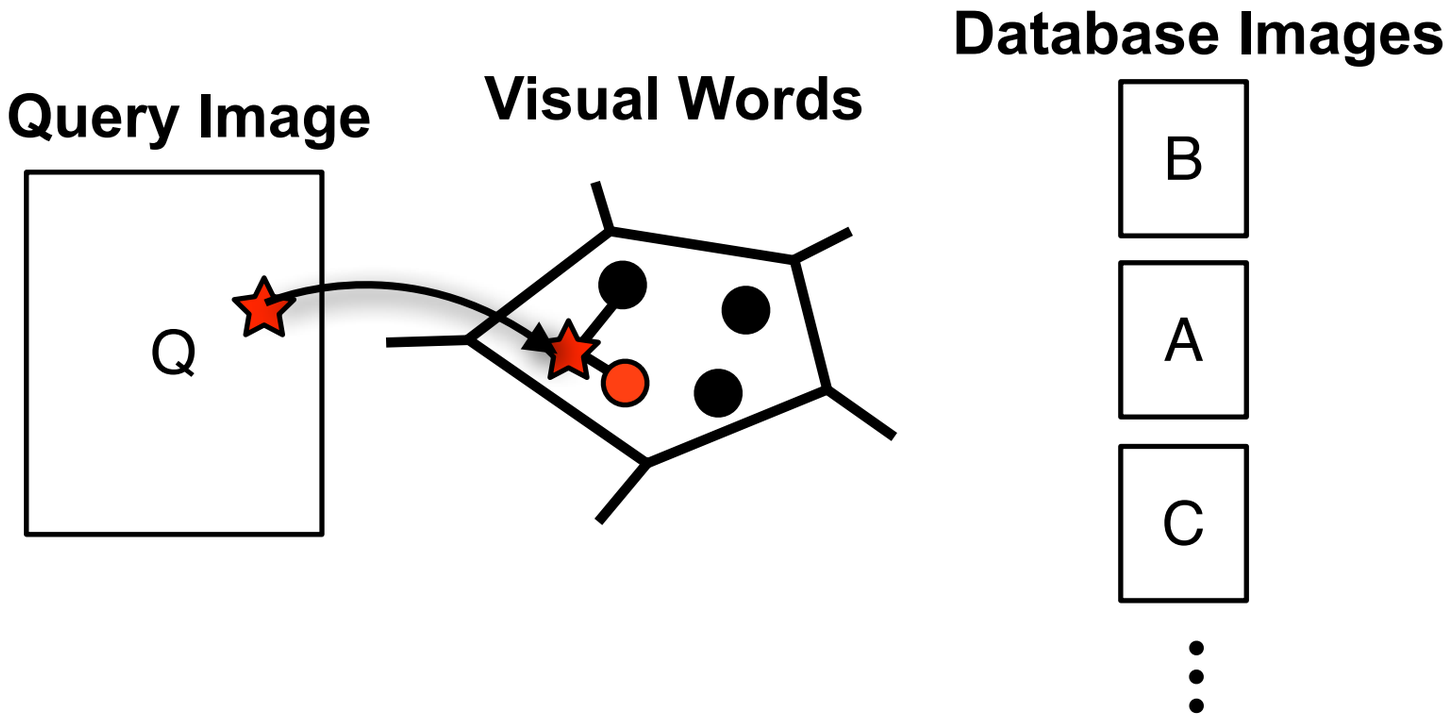


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**
- Use **Correspondence Voting** to avoid incorrect votes:

[Sattler et al., BMVC'12]

Correspondence Voting

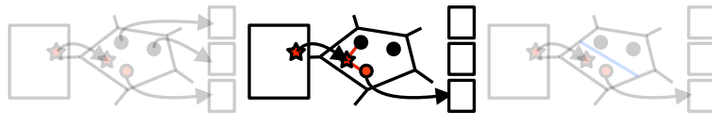


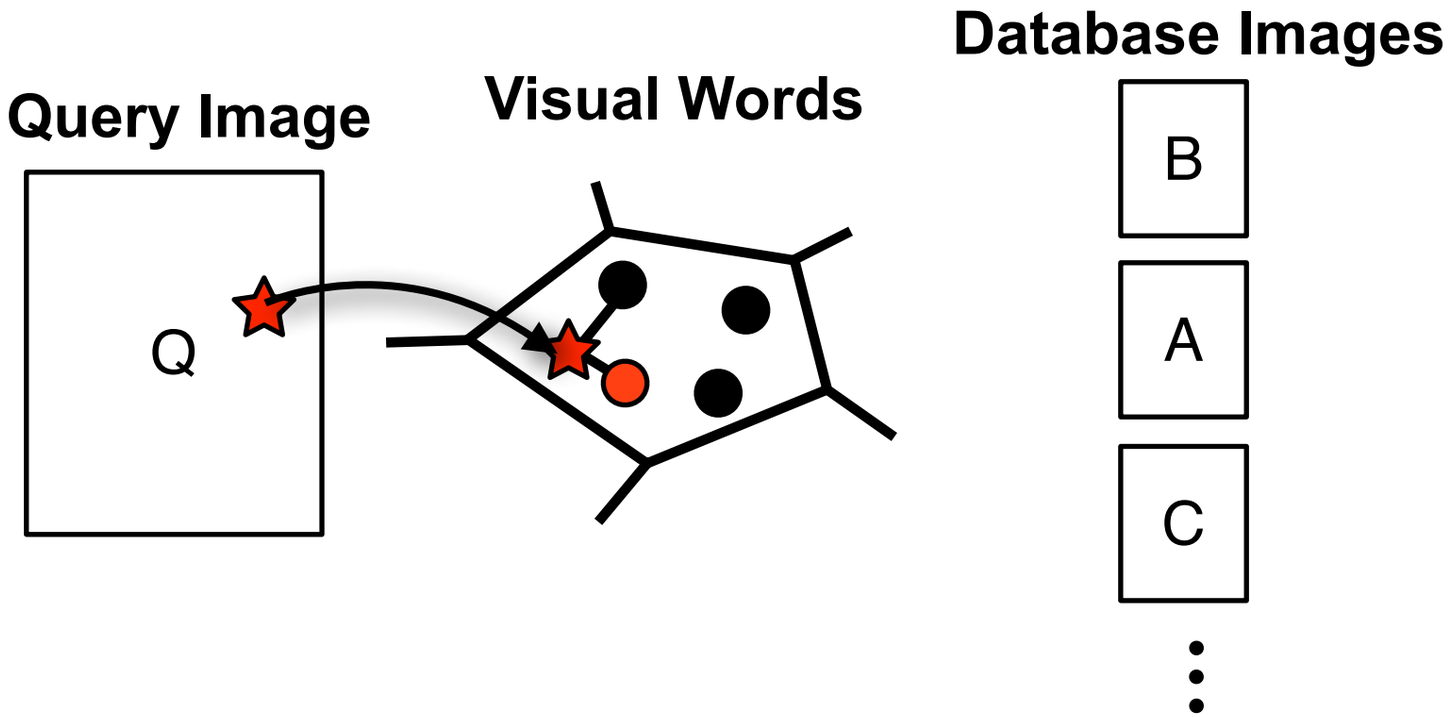


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**
- Use **Correspondence Voting** to avoid incorrect votes:

[Sattler et al., BMVC'12]

Correspondence Voting

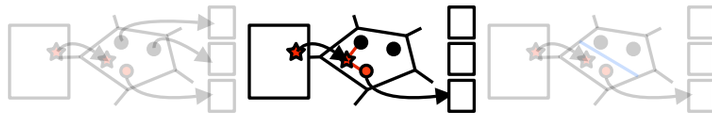


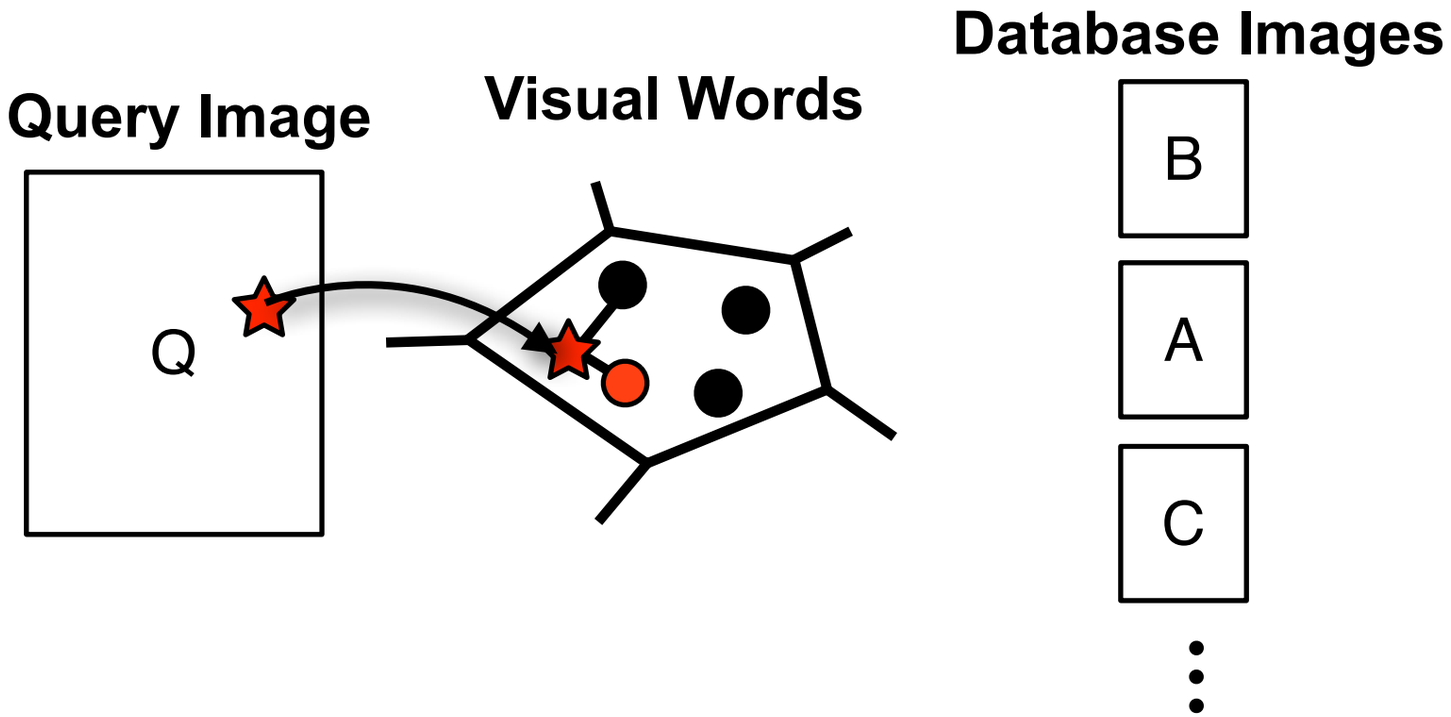


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**
- Use **Correspondence Voting** to avoid incorrect votes:
  - Inside word: Find 2 nearest neighbors & apply ratio test

[Sattler et al., BMVC'12]

Correspondence Voting

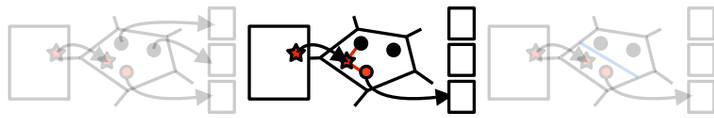


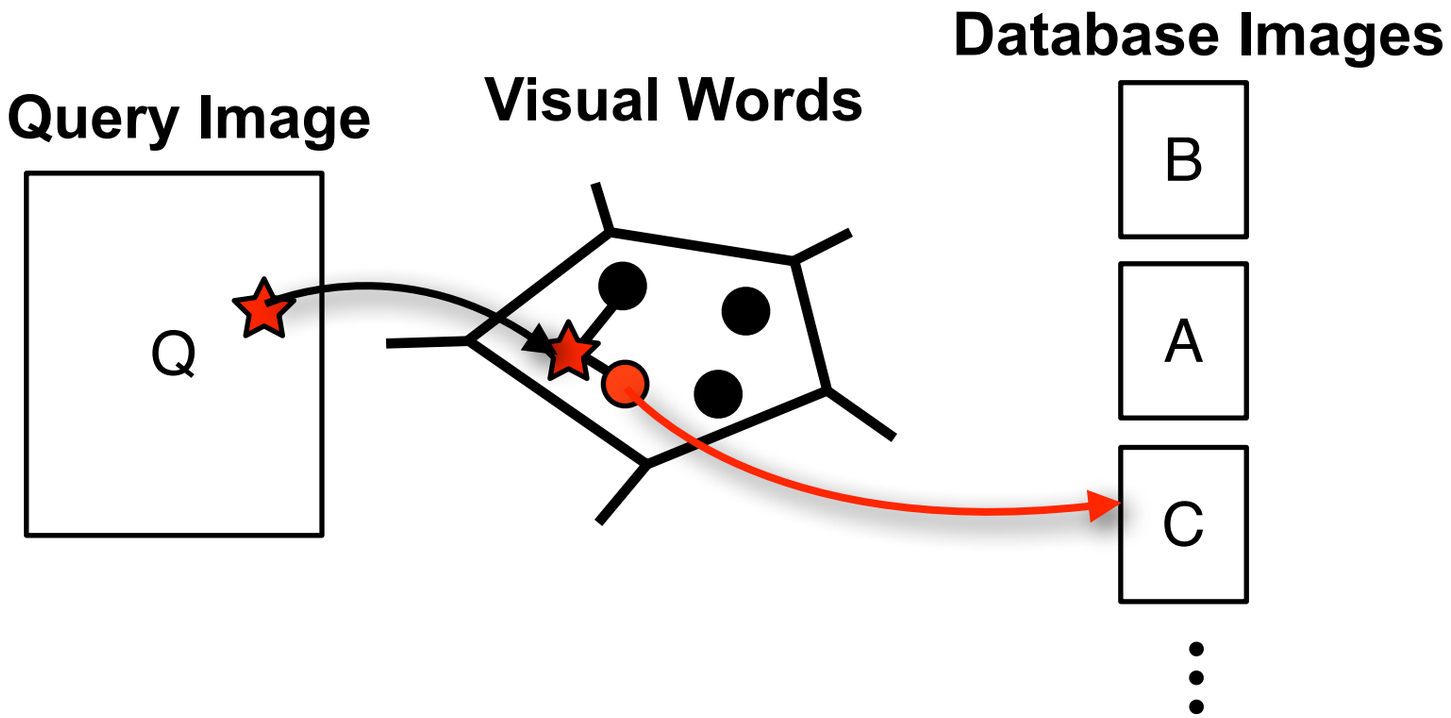


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**
- Use **Correspondence Voting** to avoid incorrect votes:
  - Inside word: Find 2 nearest neighbors & apply ratio test
  - Vote only for image of nearest neighbor

[Sattler et al., BMVC'12]

Correspondence Voting

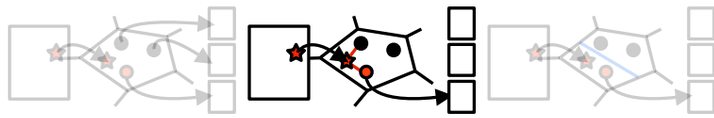


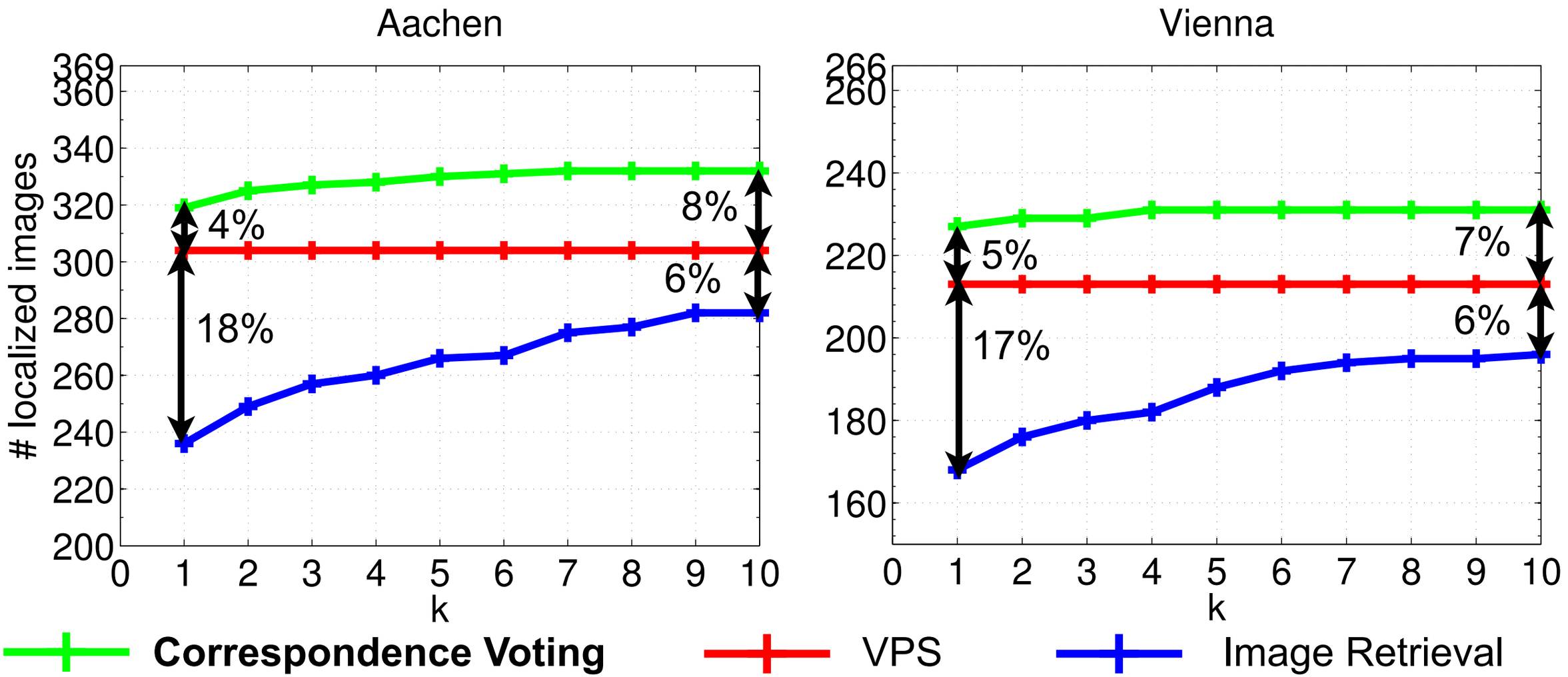


- 2D feature corresponds to at most one 3D point
- All other votes for unrelated images = **incorrect votes**
- Use **Correspondence Voting** to avoid incorrect votes:
  - Inside word: Find 2 nearest neighbors & apply ratio test
  - Vote only for image of nearest neighbor

[Sattler et al., BMVC'12]

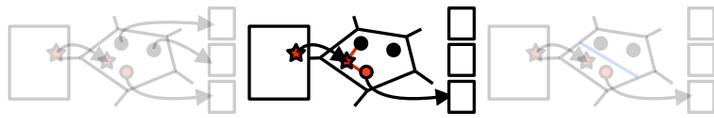
Results

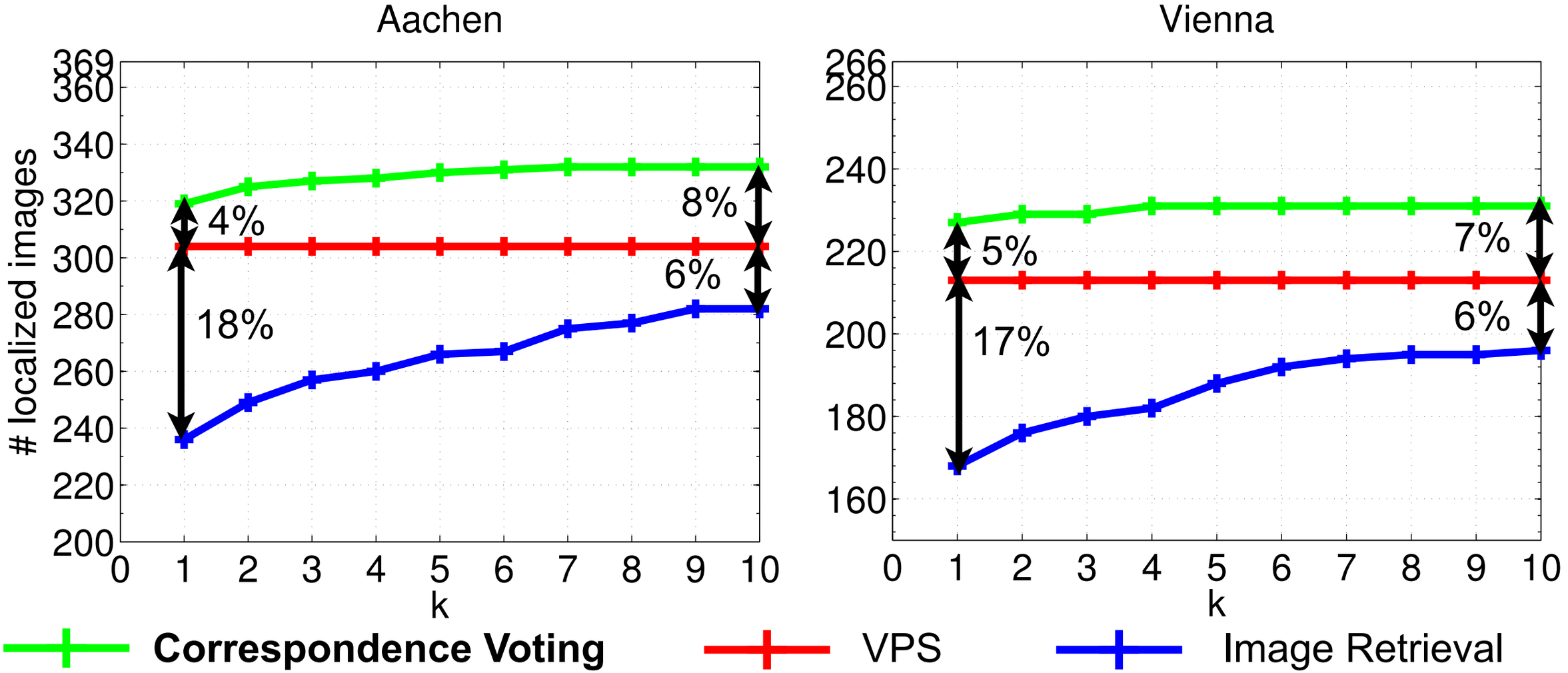




[Sattler et al., BMVC'12]

Results

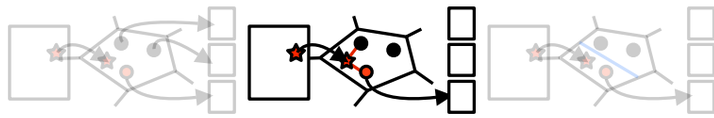


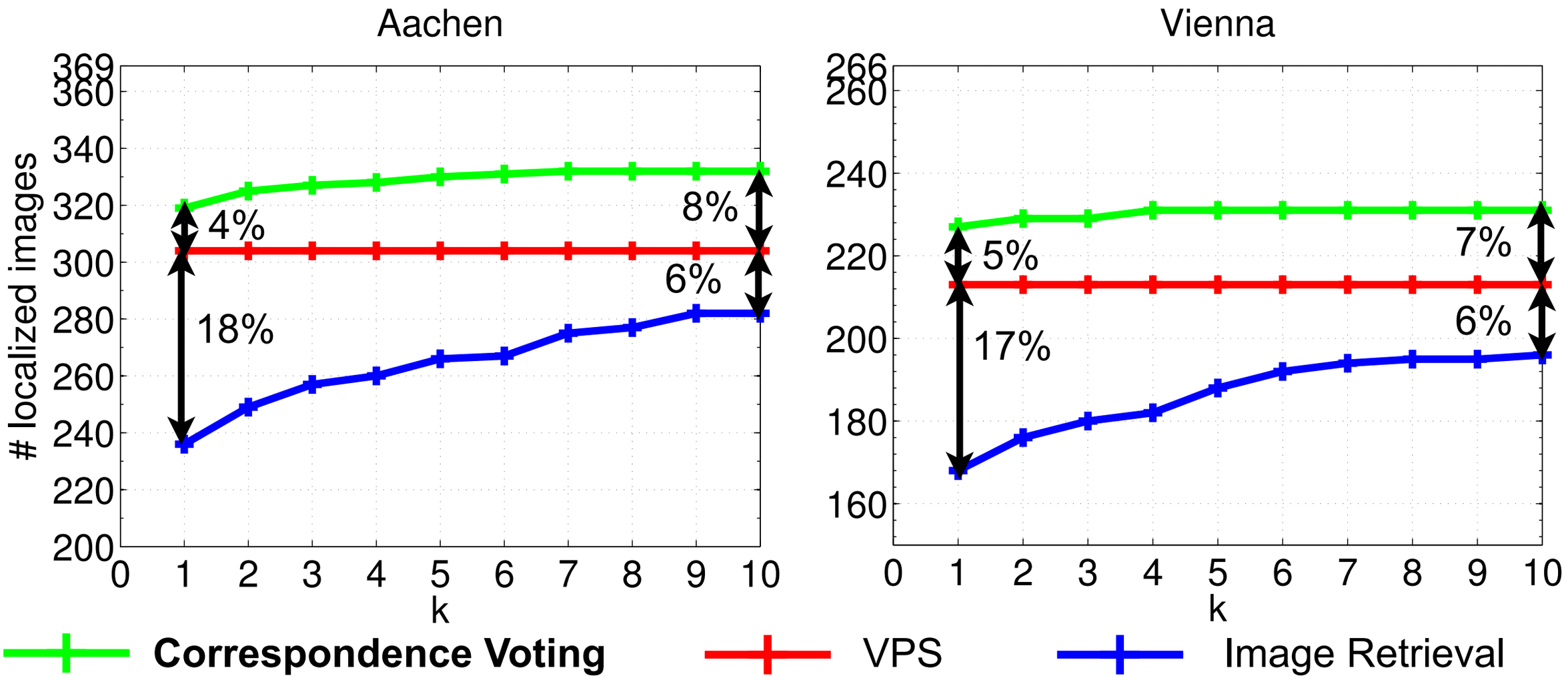


✓More effective than VPS

[Sattler et al., BMVC'12]

Results



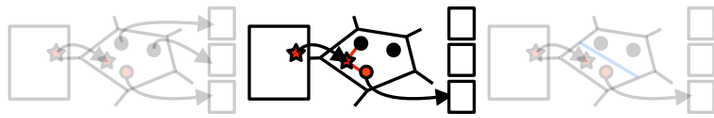


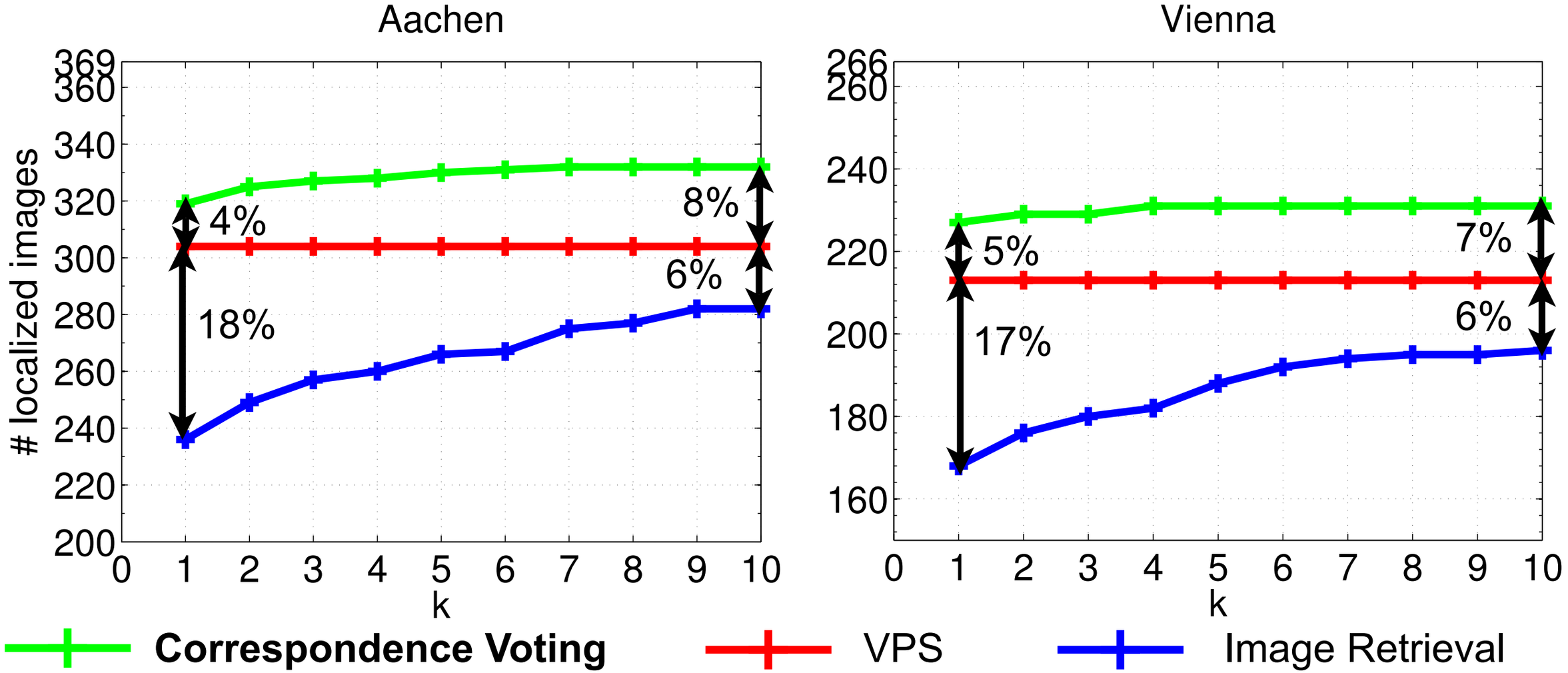
✓ More effective than VPS

✗ Full descriptors required for voting

[Sattler et al., BMVC'12]

Results





✓ More effective than VPS

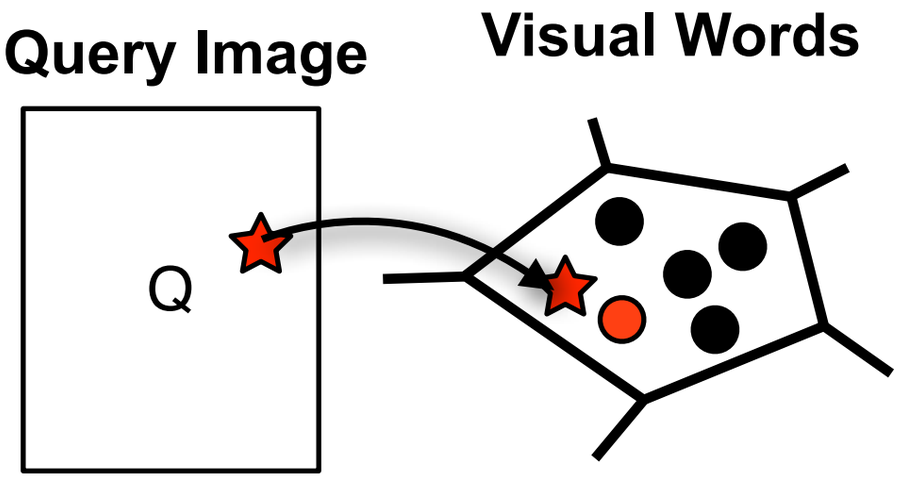
✗ Full descriptors required for voting

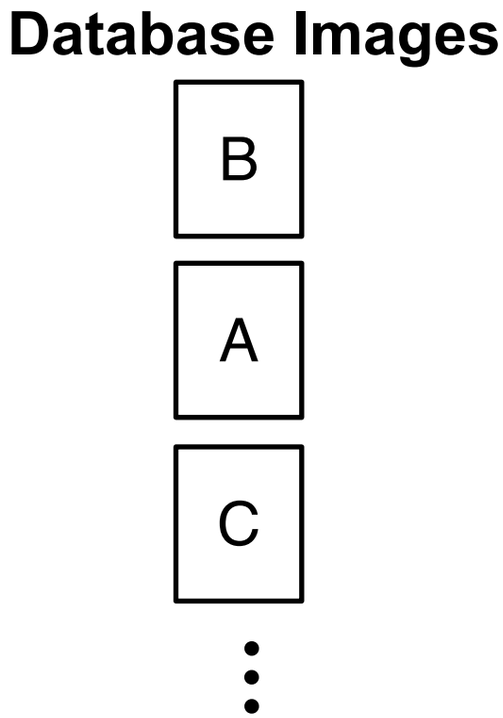
✗ Ratio test in dense descriptor space of 3D model

[Sattler et al., BMVC'12]

Hamming Voting



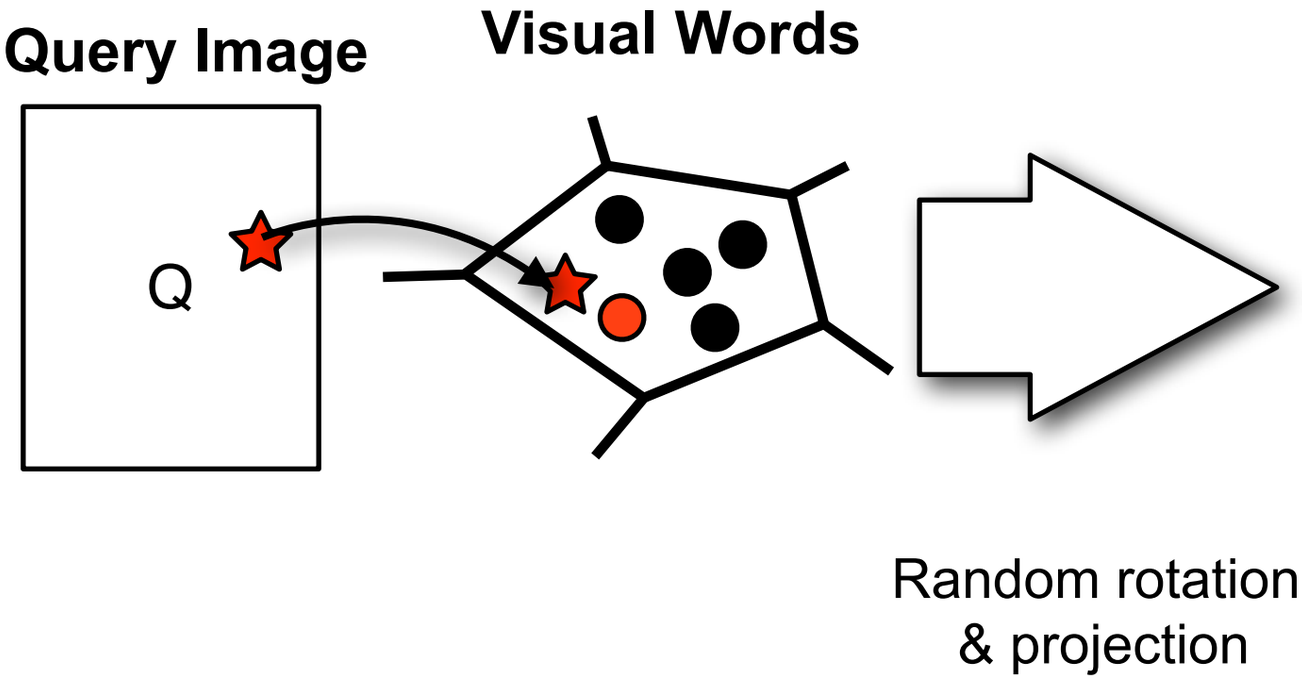


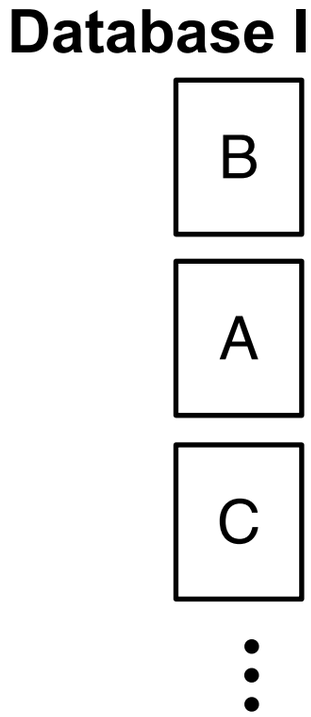


[Sattler et al., BMVC'12]

Hamming Voting



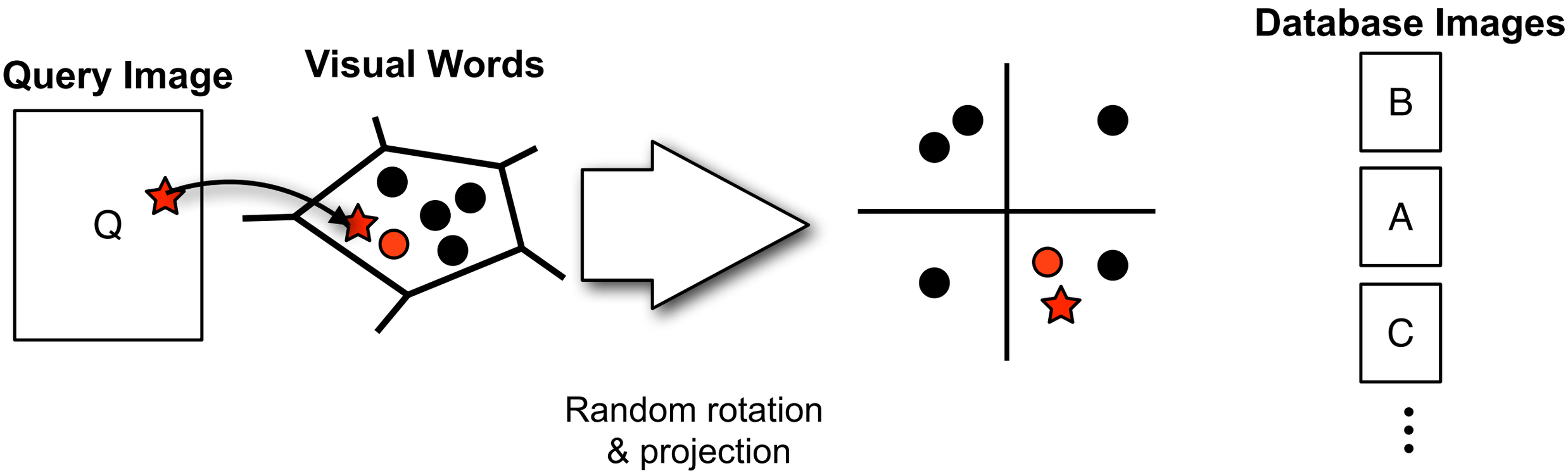




[Sattler et al., BMVC'12]

Hamming Voting

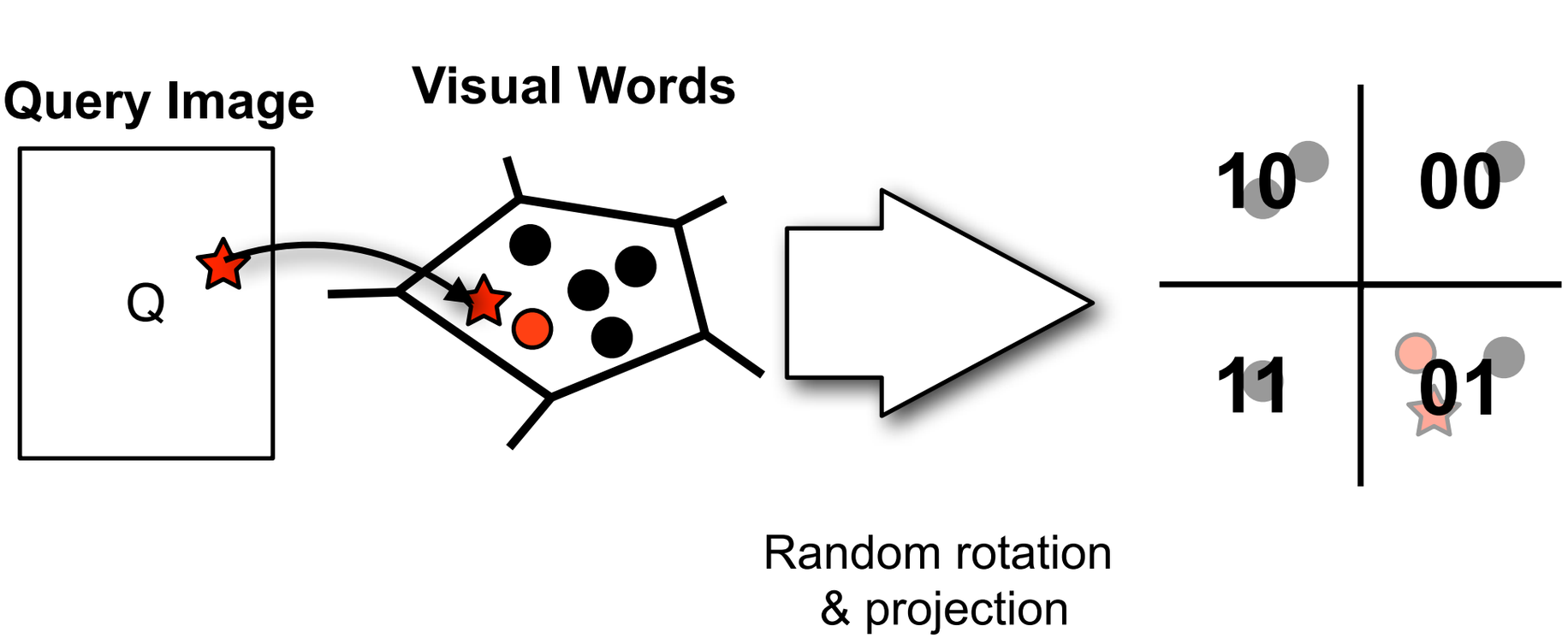




[Sattler et al., BMVC'12]

Hamming Voting

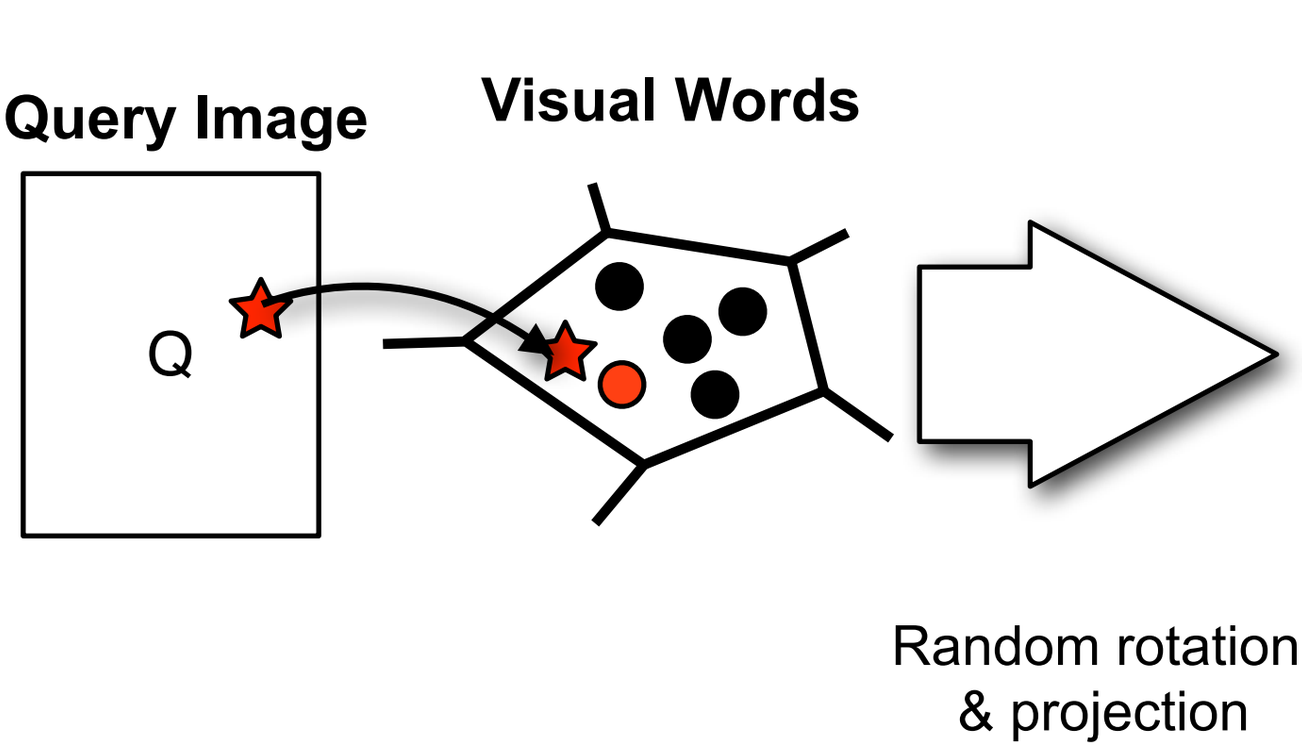




[Sattler et al., BMVC'12]

Hamming Voting

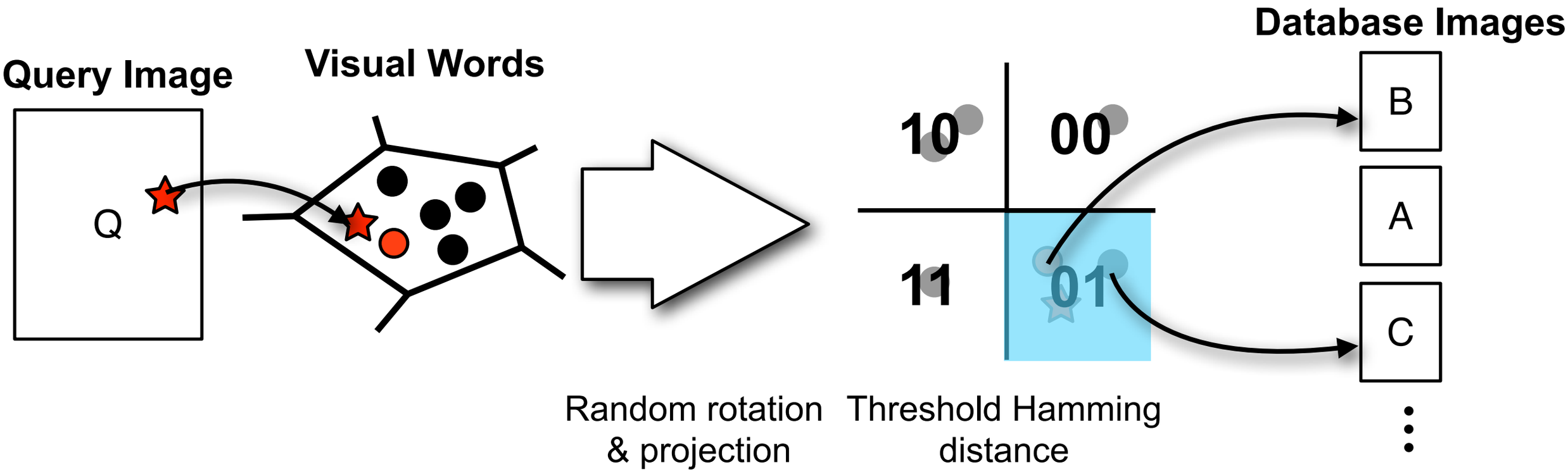




[Sattler et al., BMVC'12]

Hamming Voting

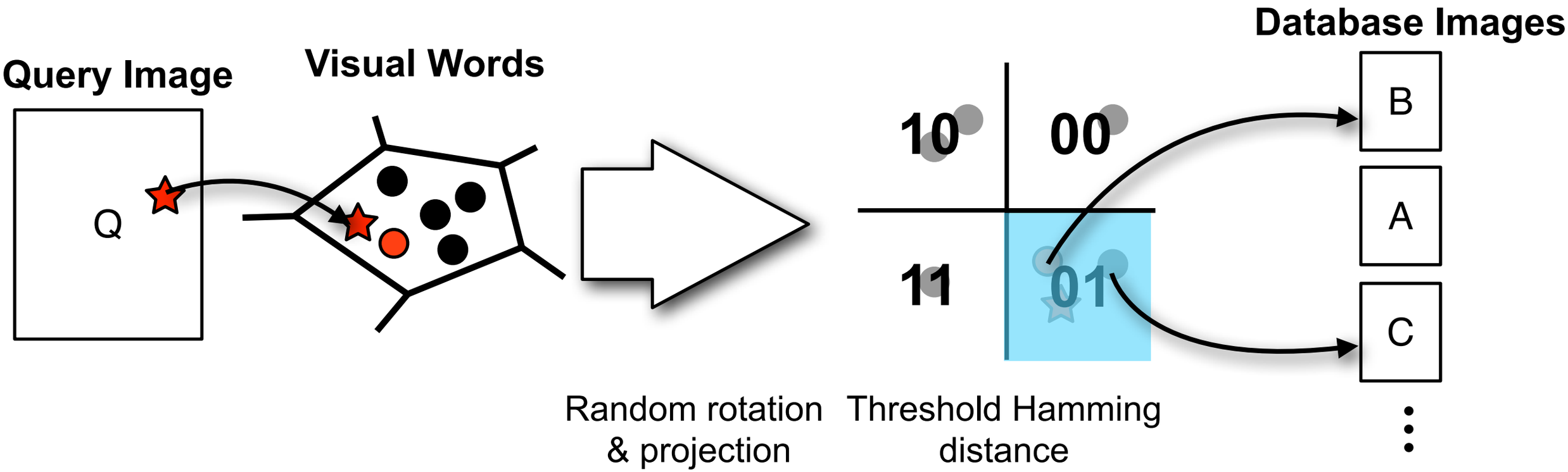




[Sattler et al., BMVC'12]

Hamming Voting



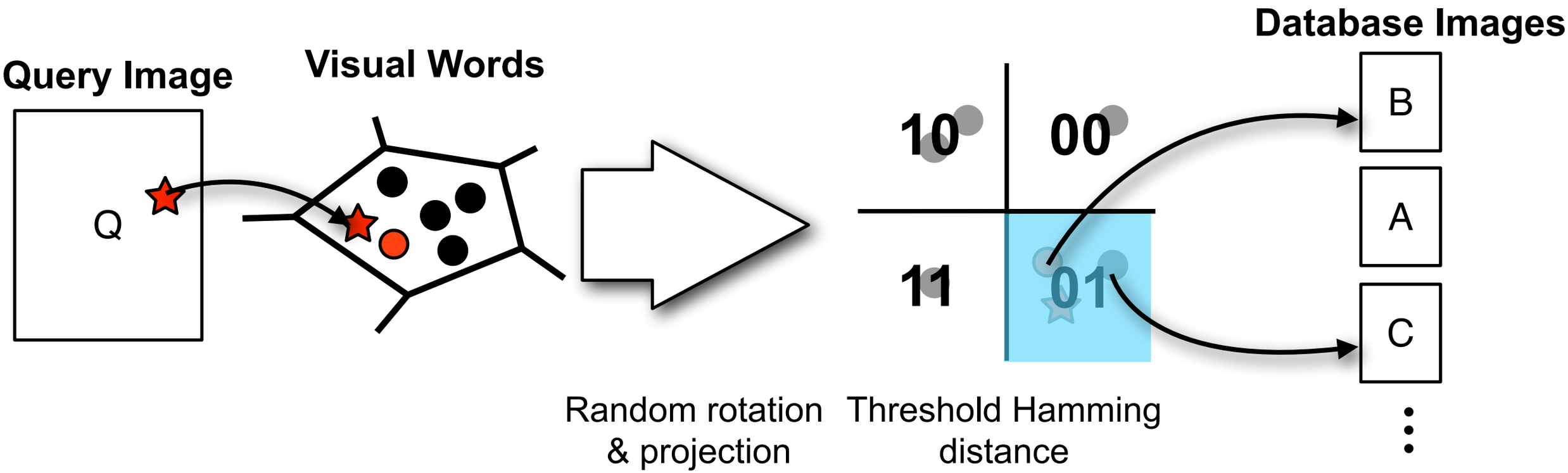


- **Hamming Embedding**: Convert SIFT to 64 bit binary descriptors [Jégou et al.,ECCV'08]

[Sattler et al., BMVC'12]

Hamming Voting





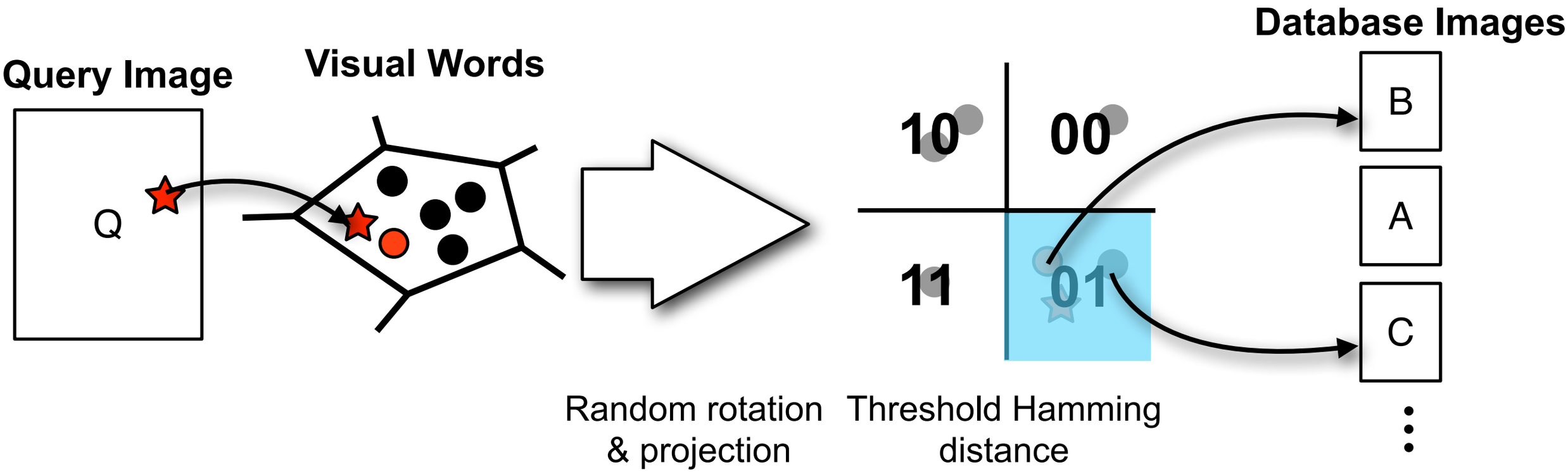
- **Hamming Embedding**: Convert SIFT to 64 bit binary descriptors [Jégou et al.,ECCV'08]

  ✓ 16x less memory & orders of magnitude faster descriptor comparisons

[Sattler et al., BMVC'12]

Hamming Voting



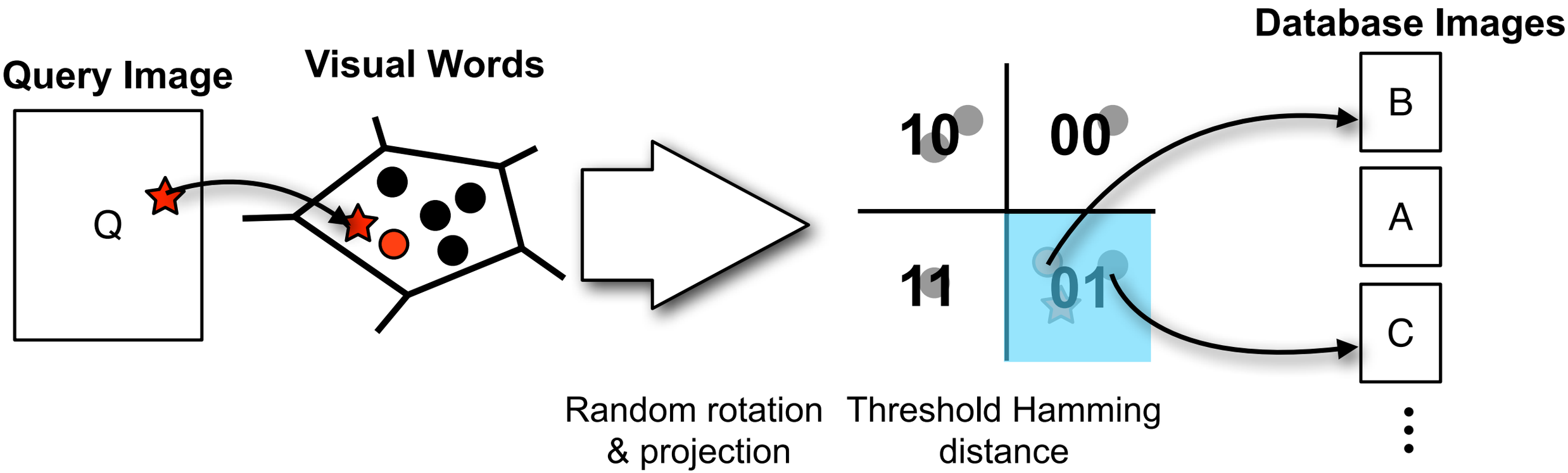


- **Hamming Embedding**: Convert SIFT to 64 bit binary descriptors [Jégou et al.,ECCV'08]
  - ✓ 16x less memory & orders of magnitude faster descriptor comparisons
  - ✓ Threshold on absolute distance instead of ratio test

[Sattler et al., BMVC'12]

Hamming Voting

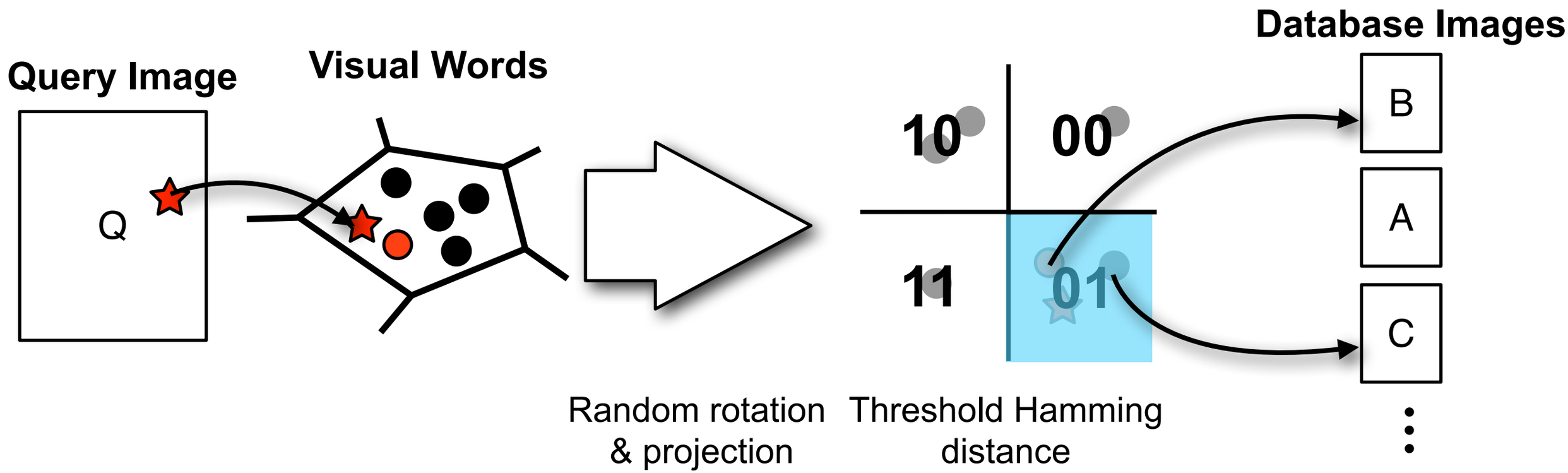




- **Hamming Embedding**: Convert SIFT to 64 bit binary descriptors [Jégou et al.,ECCV'08]
  - ✓ 16x less memory & orders of magnitude faster descriptor comparisons
  - ✓ Threshold on absolute distance instead of ratio test
  - ✓ Efficient to compute (~30ms for 10k features)

[Sattler et al., BMVC'12]

Hamming Voting

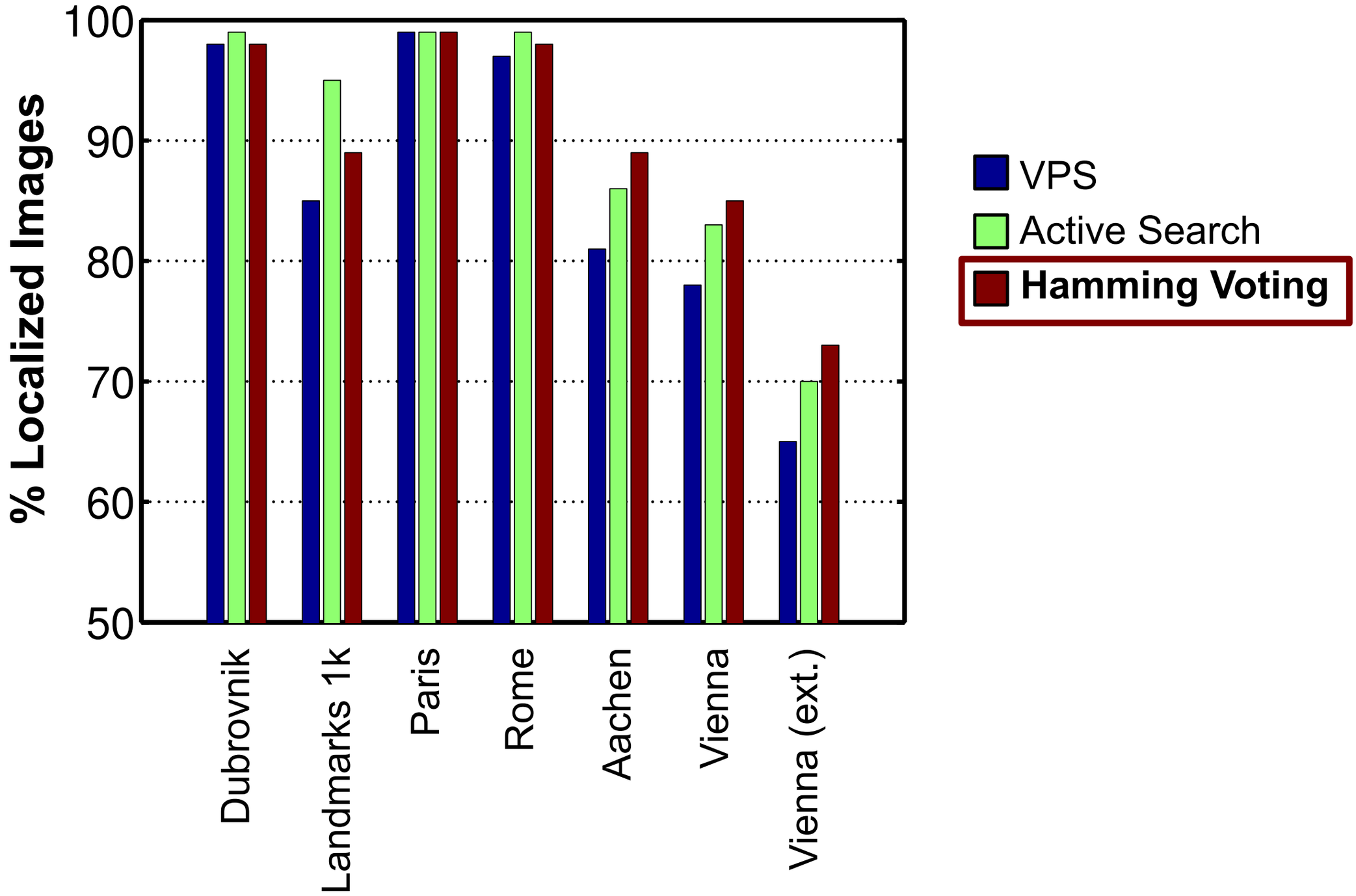


- **Hamming Embedding**: Convert SIFT to 64 bit binary descriptors [Jégou et al.,ECCV'08]
  - ✓ 16x less memory & orders of magnitude faster descriptor comparisons
  - ✓ Threshold on absolute distance instead of ratio test
  - ✓ Efficient to compute (~30ms for 10k features)
- **Hamming Voting**: Localizes nearly as many images as Correspondence Voting

[Sattler et al., BMVC'12]

Results

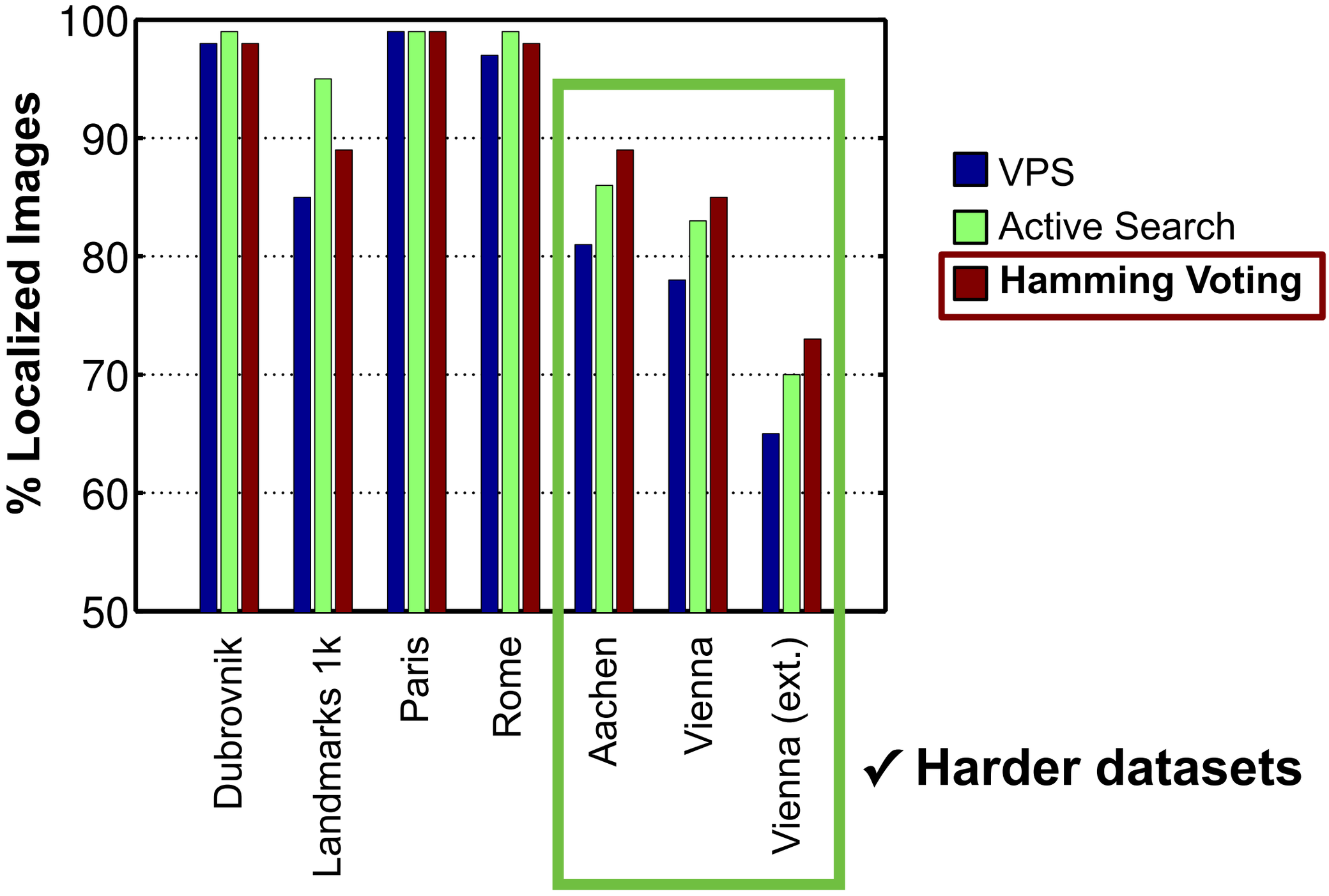




[Sattler, Thesis'14]

Results



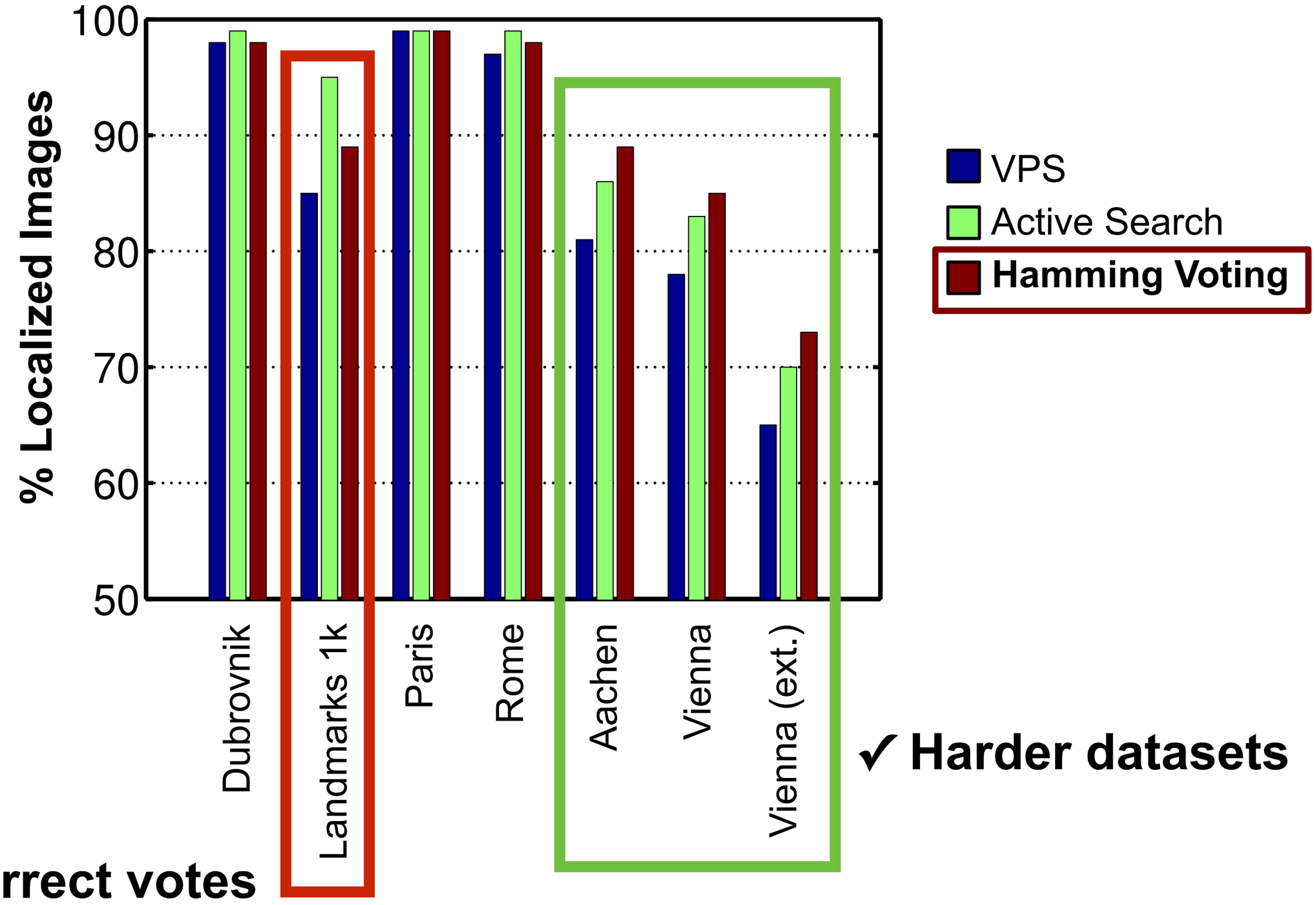


✓ **Harder datasets**

[Sattler, Thesis'14]

Results





✗ **Incorrect votes**

✓ **Harder datasets**

[Sattler, Thesis'14]

Overview

- Basics
- Localization using Prioritized Matching
- Scalable Image-Based Localization
- Conclusion / Open Challenges

Summary

- Direct Matching Approaches for Image-Based Localization:

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes
  - Gain in efficiency using prioritized matching

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes
  - Gain in efficiency using prioritized matching
  - 2D-to-3D matching more reliable than 3D-to-2D search

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes
  - Gain in efficiency using prioritized matching
  - 2D-to-3D matching more reliable than 3D-to-2D search
  - Combining both search directions strongly recommended!

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes
  - Gain in efficiency using prioritized matching
  - 2D-to-3D matching more reliable than 3D-to-2D search
  - Combining both search directions strongly recommended!
  - Exploit co-visibility information to gain efficiency

Summary

- Direct Matching Approaches for Image-Based Localization:
  - Establish 2D-3D matches by directly comparing descriptors
  - Extremely efficient even for relatively large scenes
  - Gain in efficiency using prioritized matching
    - 2D-to-3D matching more reliable than 3D-to-2D search
    - Combining both search directions strongly recommended!
    - Exploit co-visibility information to gain efficiency
- Achieve good localization accuracy

Summary

- Clear limits to scalability of direct matching schemes:

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space
- Possible solution: Allow more false matches

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space
- Possible solution: Allow more false matches
  - Adapt RANSAC's sampling strategy

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space
- Possible solution: Allow more false matches
  - Adapt RANSAC's sampling strategy
  - Alternative strategies independent of outlier ratios [Svärm et al., CVPR'14]

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space
- Possible solution: Allow more false matches
  - Adapt RANSAC's sampling strategy
  - Alternative strategies independent of outlier ratios [Svärm et al., CVPR'14]
  - Enables us to handle both ratio test & memory problems:

Summary

- Clear limits to scalability of direct matching schemes:
  - Memory consumption
  - Failing ratio test for larger models
  - Still valuable: Combine direct matching with pose priors to restrict search space
- Possible solution: Allow more false matches
  - Adapt RANSAC's sampling strategy
  - Alternative strategies independent of outlier ratios [Svärm et al., CVPR'14]
  - Enables us to handle both ratio test & memory problems:
    - Find multiple nearest neighbors using quantized descriptors

Robustness against Illumination Changes

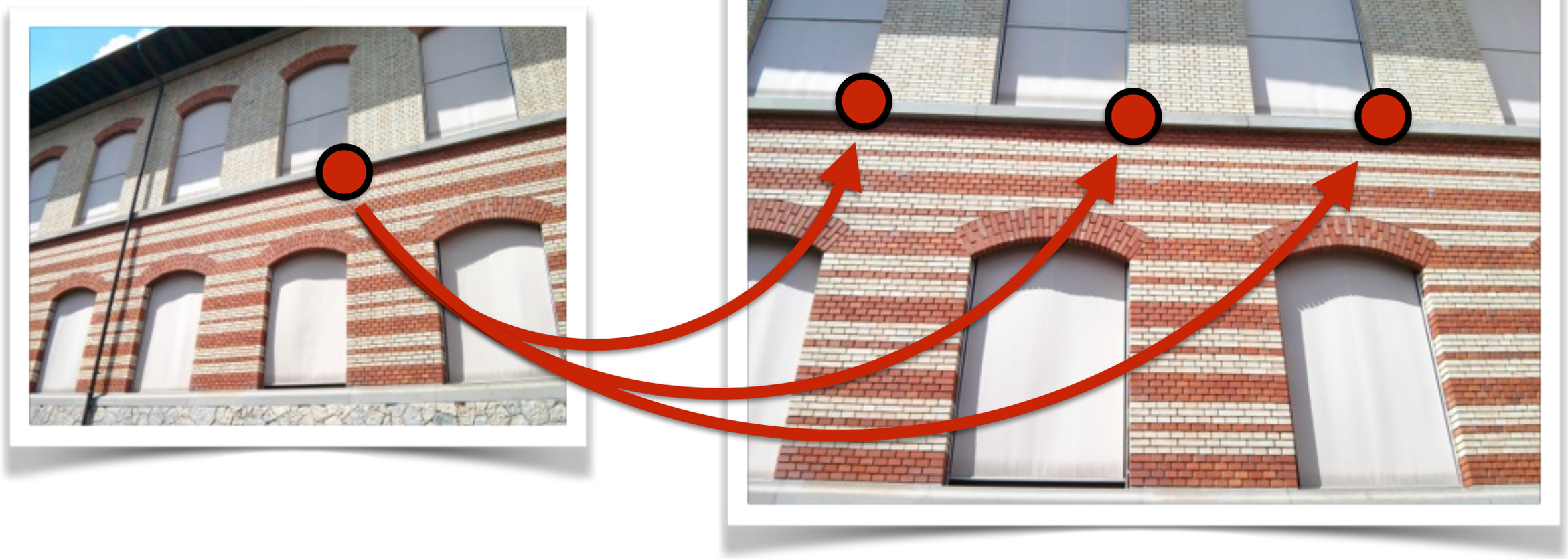




© Google

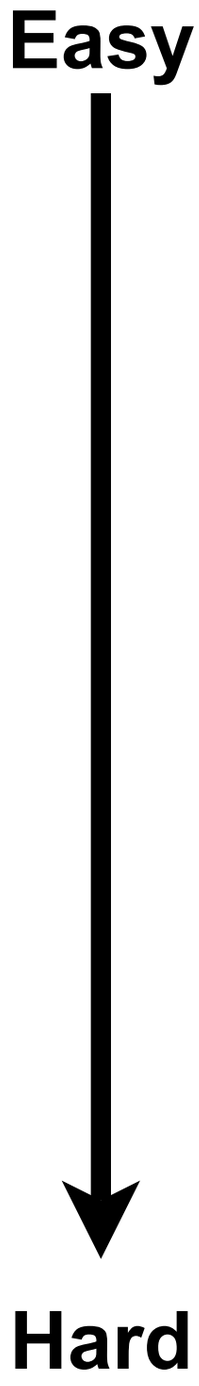
Good luck finding feature matches!

Camera Pose Estimation Under Repetitions



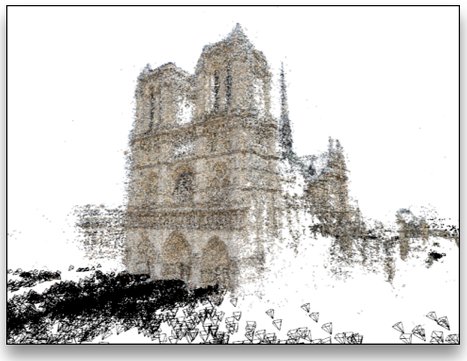
- RANSAC's run-time increases exponentially with outlier ratio
- Need alternative that is independent of outlier ratio
  - Very interesting approach: [Svärm et al., CVPR'14]
- Obtain multiple plausible poses?

What Situations Can Be Handled



What Situations Can Be Handled

Easy

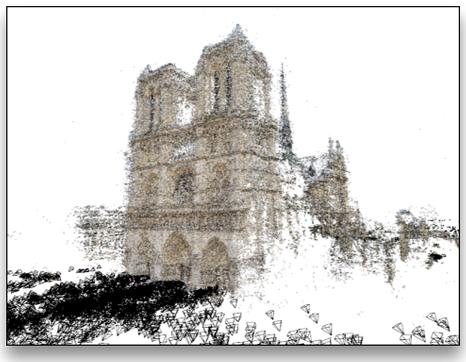


- Database & query images from same source, e.g., Flickr
- 97% - 100% localization rates
- *Challenges*: Run-time & memory consumption for large scale

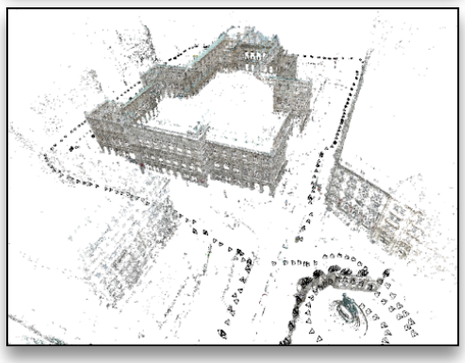
Hard

What Situations Can Be Handled

Easy



- Database & query images from same source, e.g., Flickr
- 97% - 100% localization rates
- *Challenges*: Run-time & memory consumption for large scale

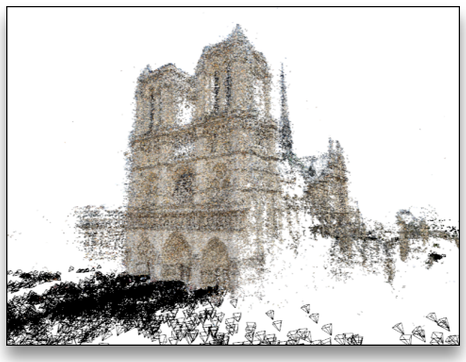


- Database & query images from different spatial distributions
- 70% - 90% localization rates
- *Challenges*: Deal with larger variety in viewpoints

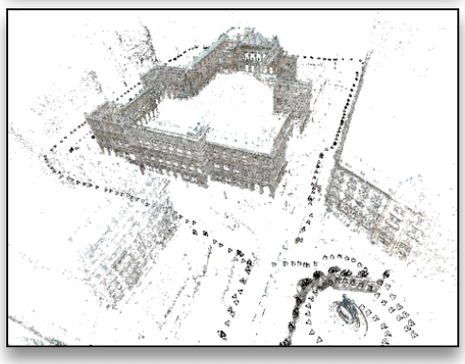
Hard

What Situations Can Be Handled

Easy



- Database & query images from same source, e.g., Flickr
- 97% - 100% localization rates
- *Challenges*: Run-time & memory consumption for large scale

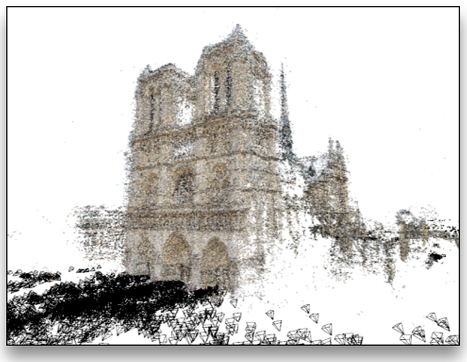


- Database & query images from different spatial distributions
- 70% - 90% localization rates
- *Challenges*: Deal with larger variety in viewpoints

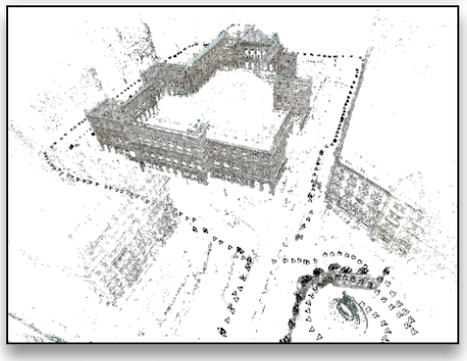
Hard

What Situations Can Be Handled

Easy



- Database & query images from same source, e.g., Flickr
- 97% - 100% localization rates
- *Challenges*: Run-time & memory consumption for large scale



- Database & query images from different spatial distributions
- 70% - 90% localization rates
- *Challenges*: Deal with larger variety in viewpoints

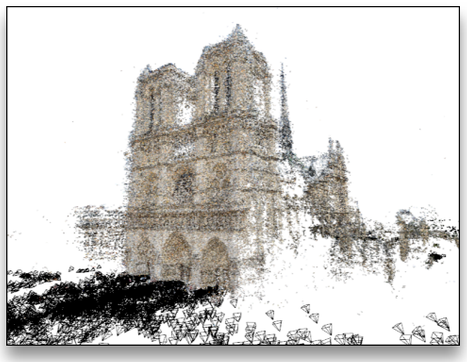


- Streetview imagery
- 50% - 65% localization rates
- *Challenges*: Repetitions, viewpoint variations, scale

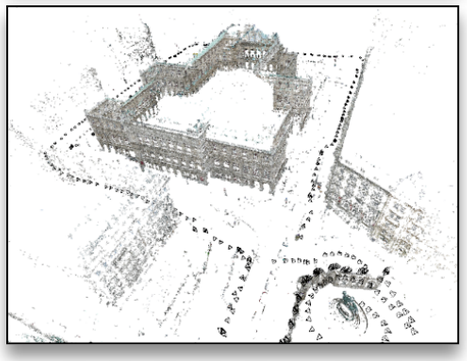
Hard

What Situations Can Be Handled

Easy



- Database & query images from same source, e.g., Flickr
- 97% - 100% localization rates
- *Challenges*: Run-time & memory consumption for large scale



- Database & query images from different spatial distributions
- 70% - 90% localization rates
- *Challenges*: Deal with larger variety in viewpoints



- Streetview imagery
- 50% - 65% localization rates
- *Challenges*: Repetitions, viewpoint variations, scale



- Indoor scenarios
- *Challenges*: Identical structures, small distance to scene

Hard

Questions?