Keypoint Application: Panorama

Xiaojuan Wang

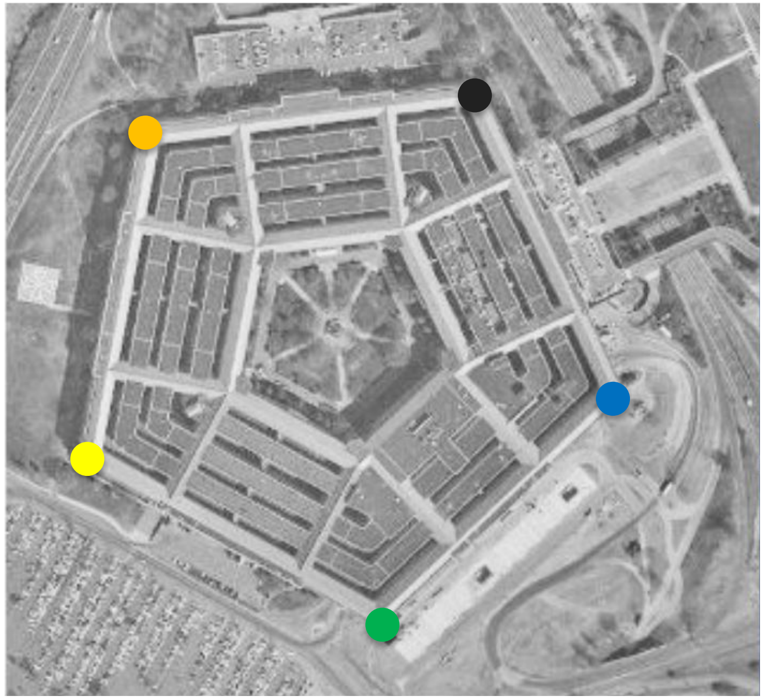
Outline

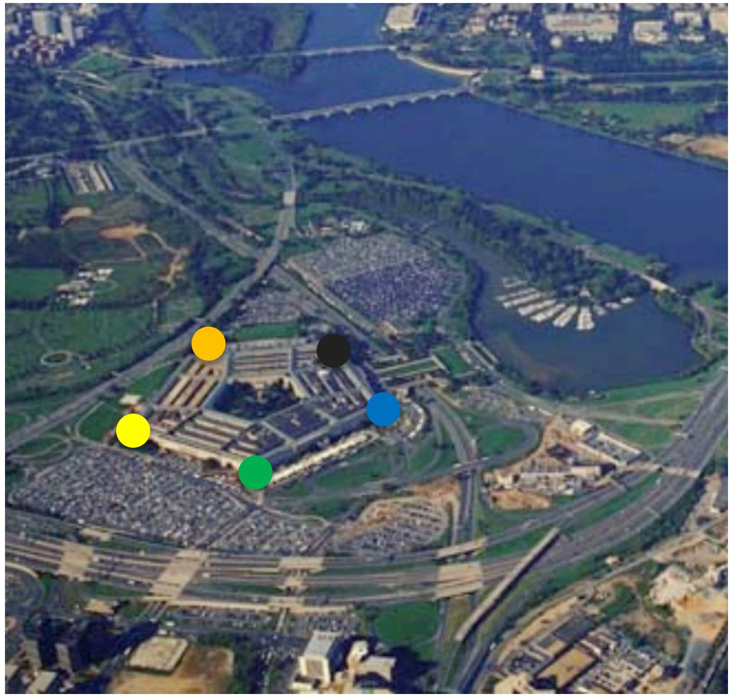
- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation

Outline

- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation

What are keypoints?

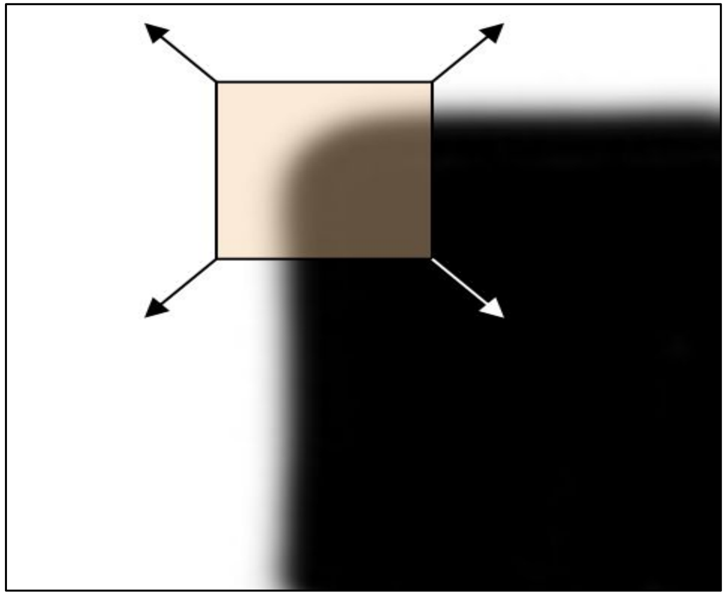


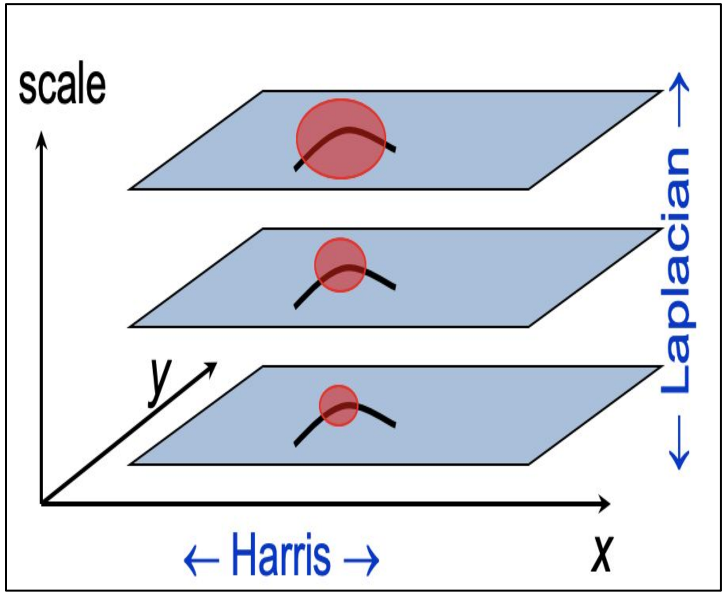


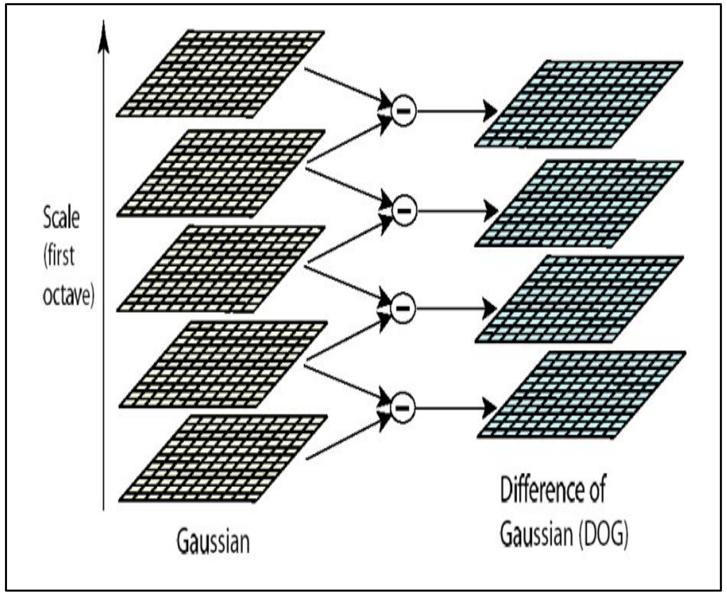
Reliable, unique points in images which can be used to find corresponding regions in different images of the same scene

Finding keypoints

SCALE INVARIANT





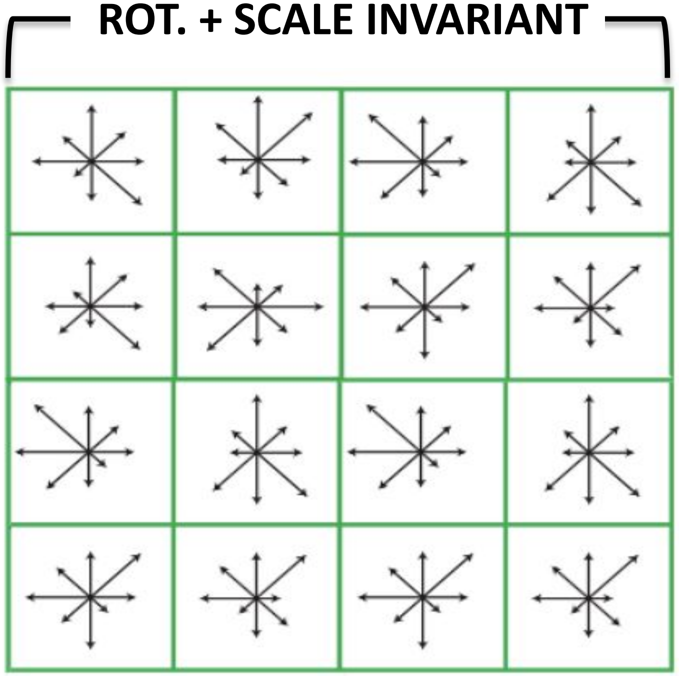


Harris Corner Detector
Use gradient Eigenvalues to find corners at a certain scale

Harris-Laplacian
Find keypoints using Harris and scale using Laplacian filter

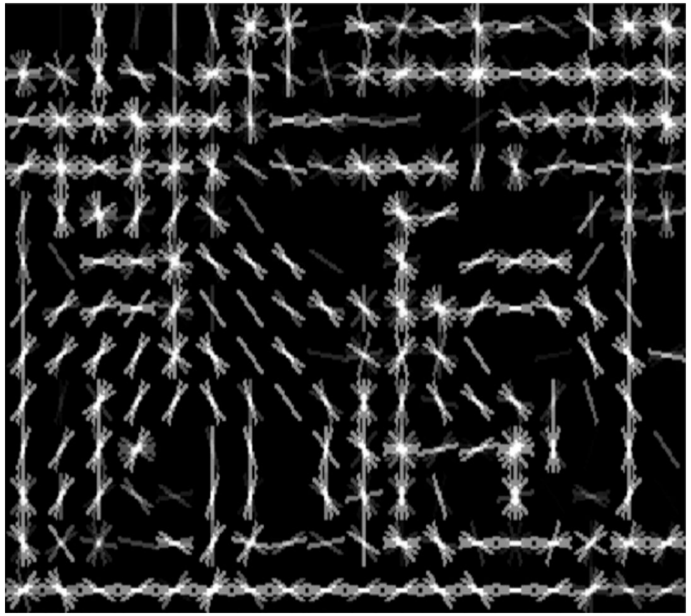
DoG
Use DoG filters to find keypoints across space and scale

Describing keypoints



ROT. + SCALE INVARIANT

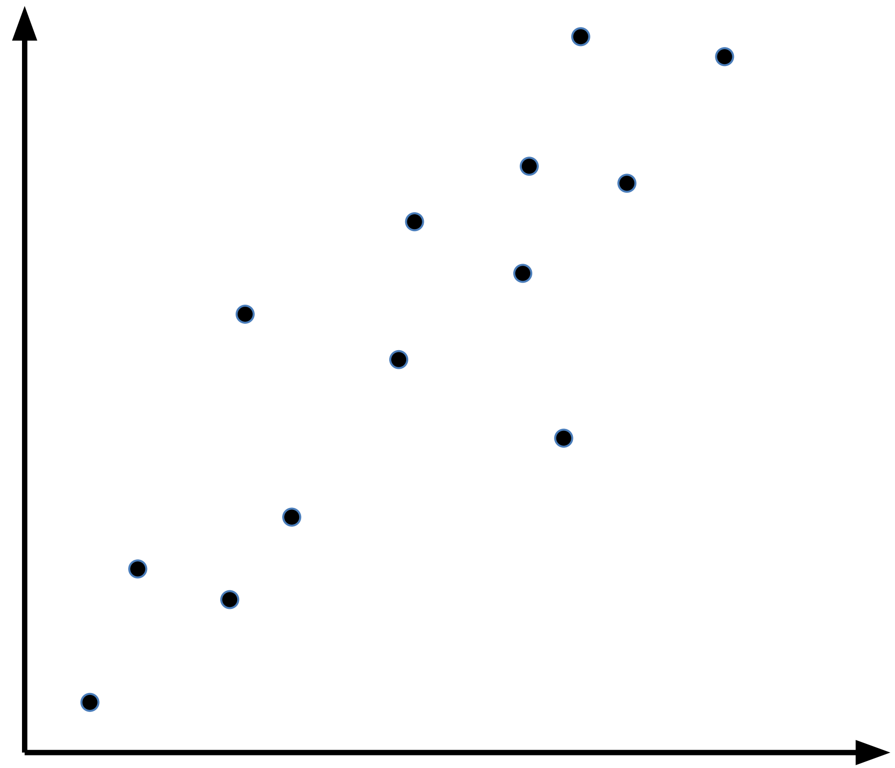
SIFT Descriptor
Keypoints as histogram of normalize gradient orientation



HoG
Region (or image) as histograms of local gradients

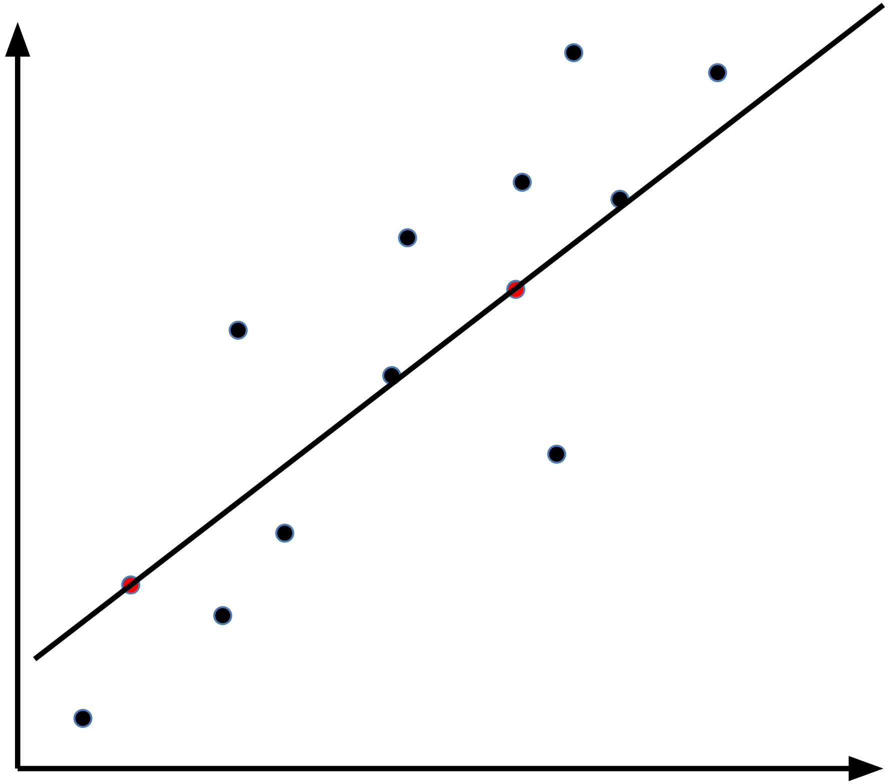
RANSAC – algorithm for model fitting

- Repeat n times:
  - Sample and form hypothesis
  - Find number of inliers
  - If max_inliers, save model
- Recompute model on inliers



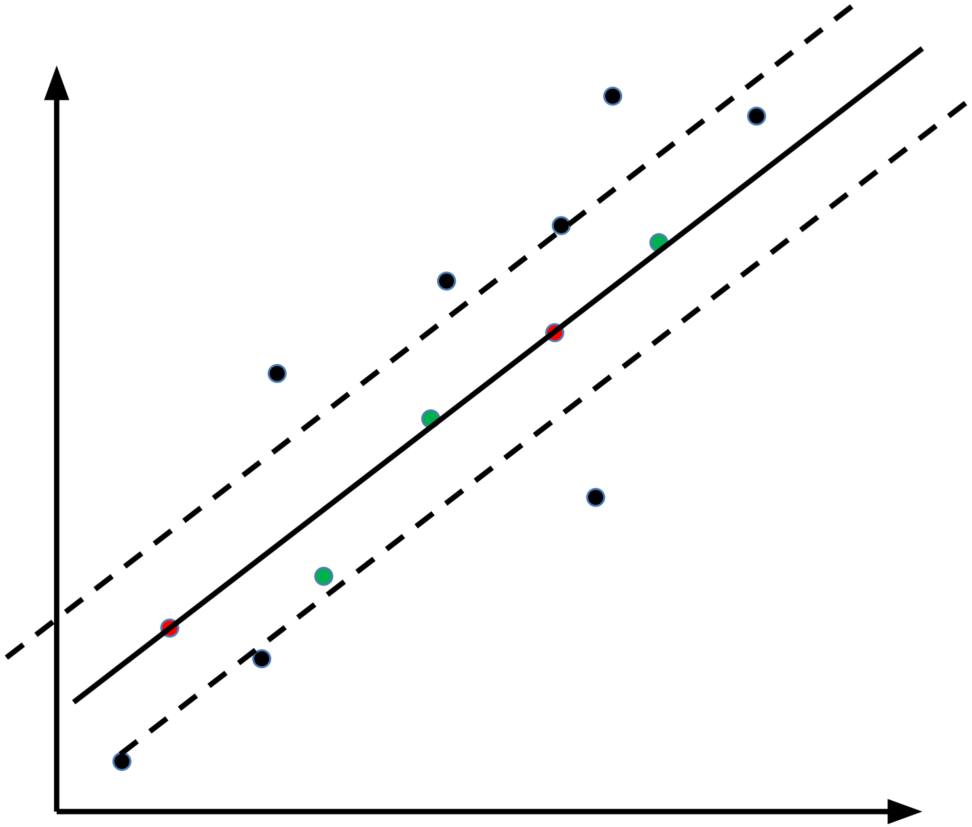
RANSAC – algorithm for model fitting

- Repeat n times:
  - **Sample and form hypothesis**
  - Find number of inliers
  - If max_inliers, save model
- Recompute model on inliers



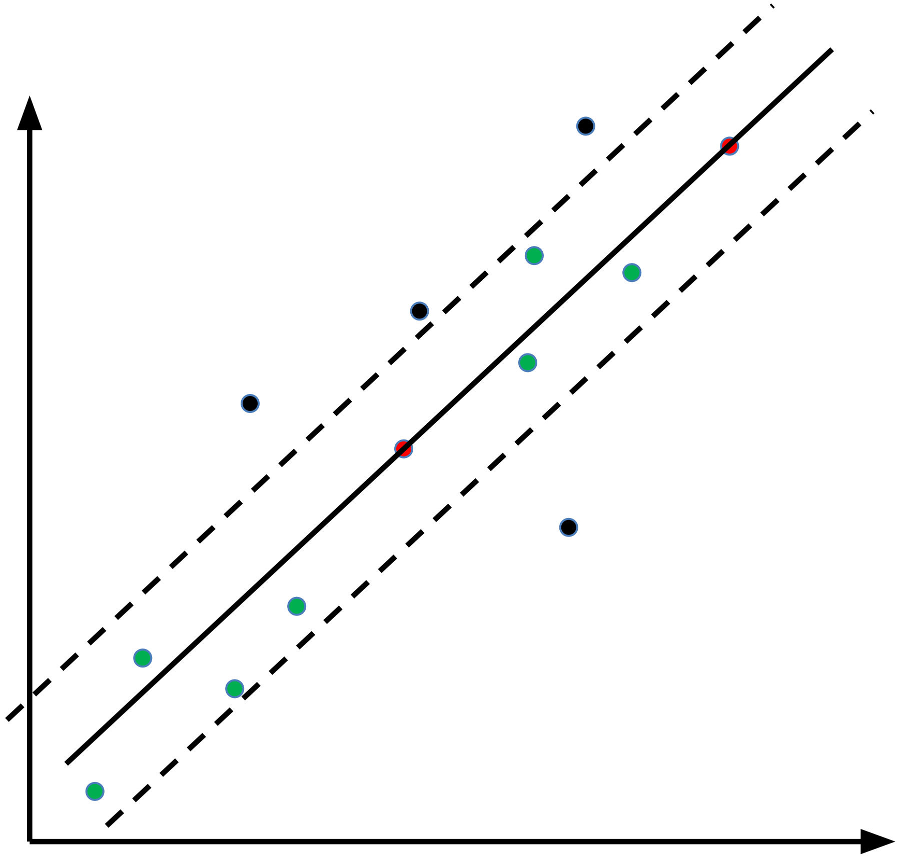
RANSAC – algorithm for model fitting

- Repeat n times:
  - Sample and form hypothesis
  - **Find number of inliers**
  - If max_inliers, save model
- Recompute model on inliers



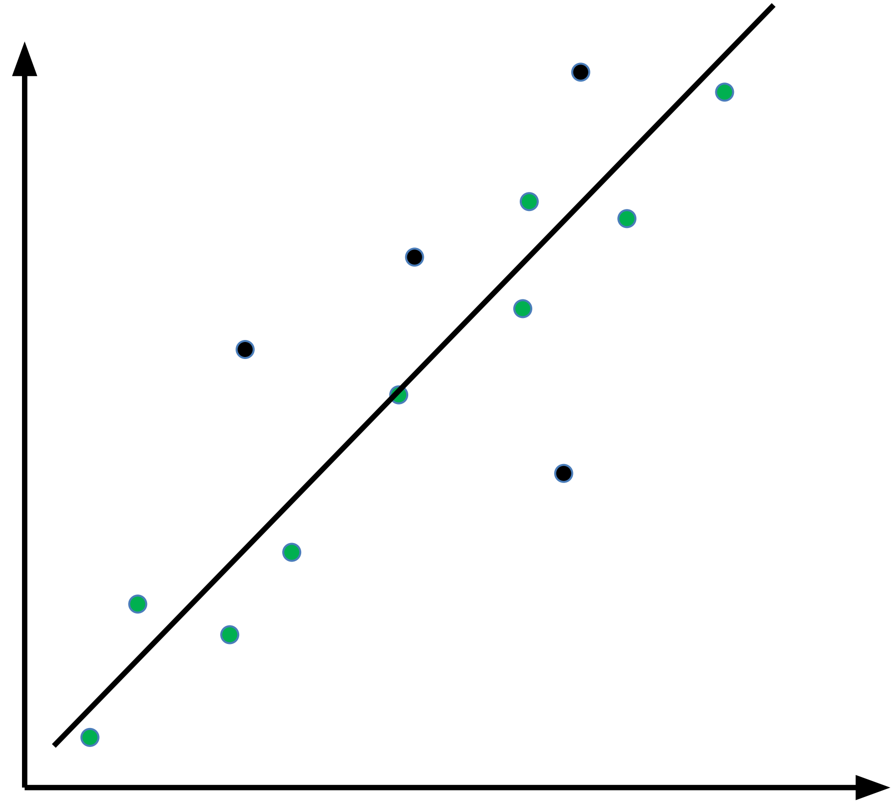
RANSAC – algorithm for model fitting

- Repeat n times:
  - Sample and form hypothesis
  - Find number of inliers
  - **If max_inliers, save model**
- Recompute model on inliers



RANSAC – algorithm for model fitting

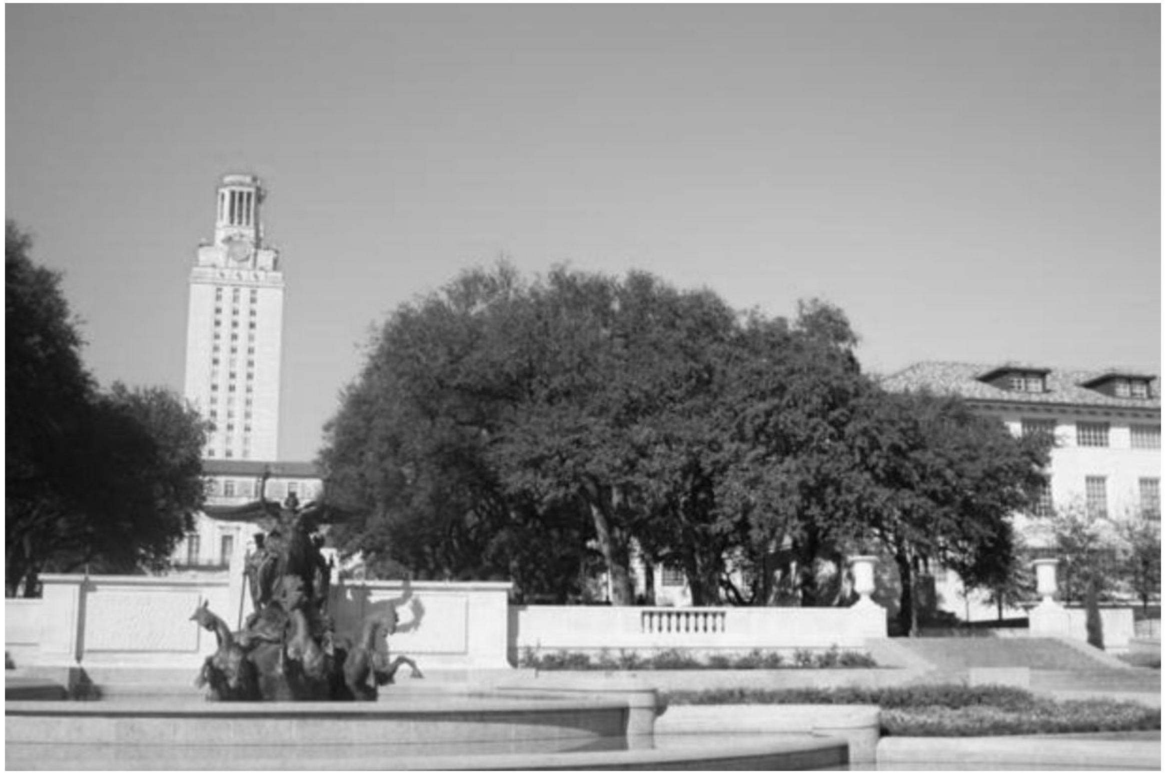
- Repeat n times:
    - Sample and form hypothesis
    - Find number of inliers
    - If max_inliers, save model
- **Recompute model on inliers**

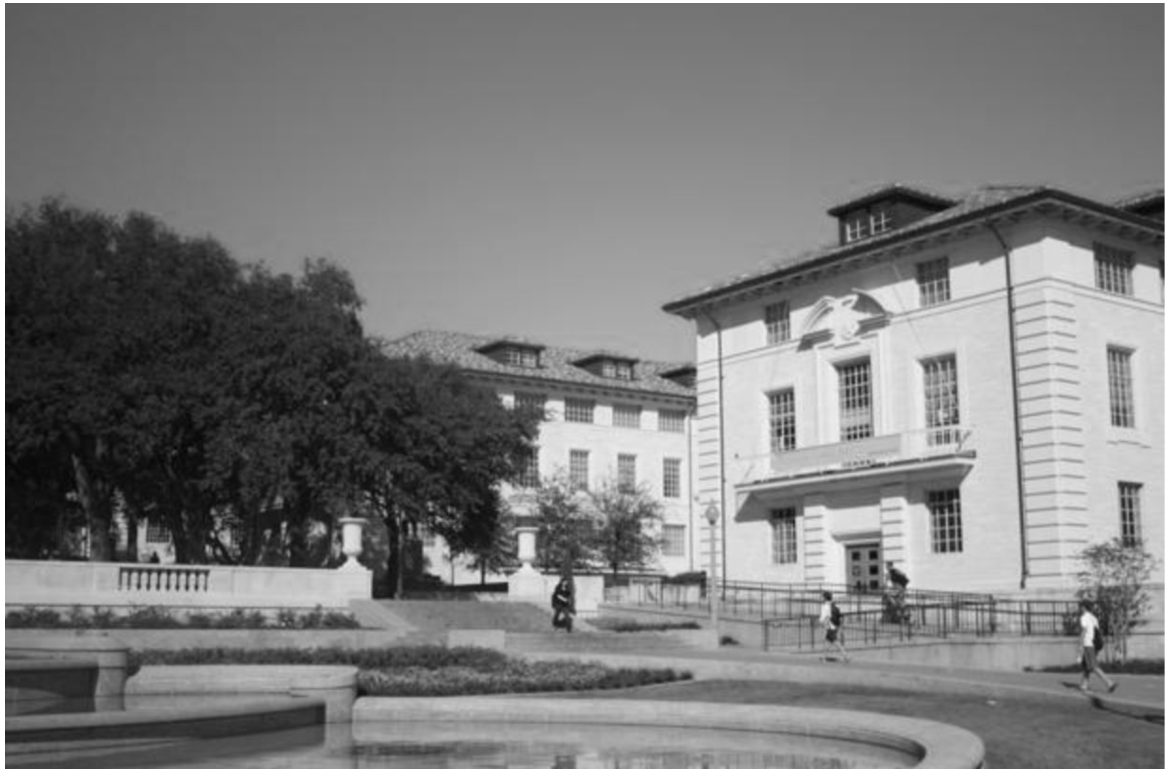


Outline

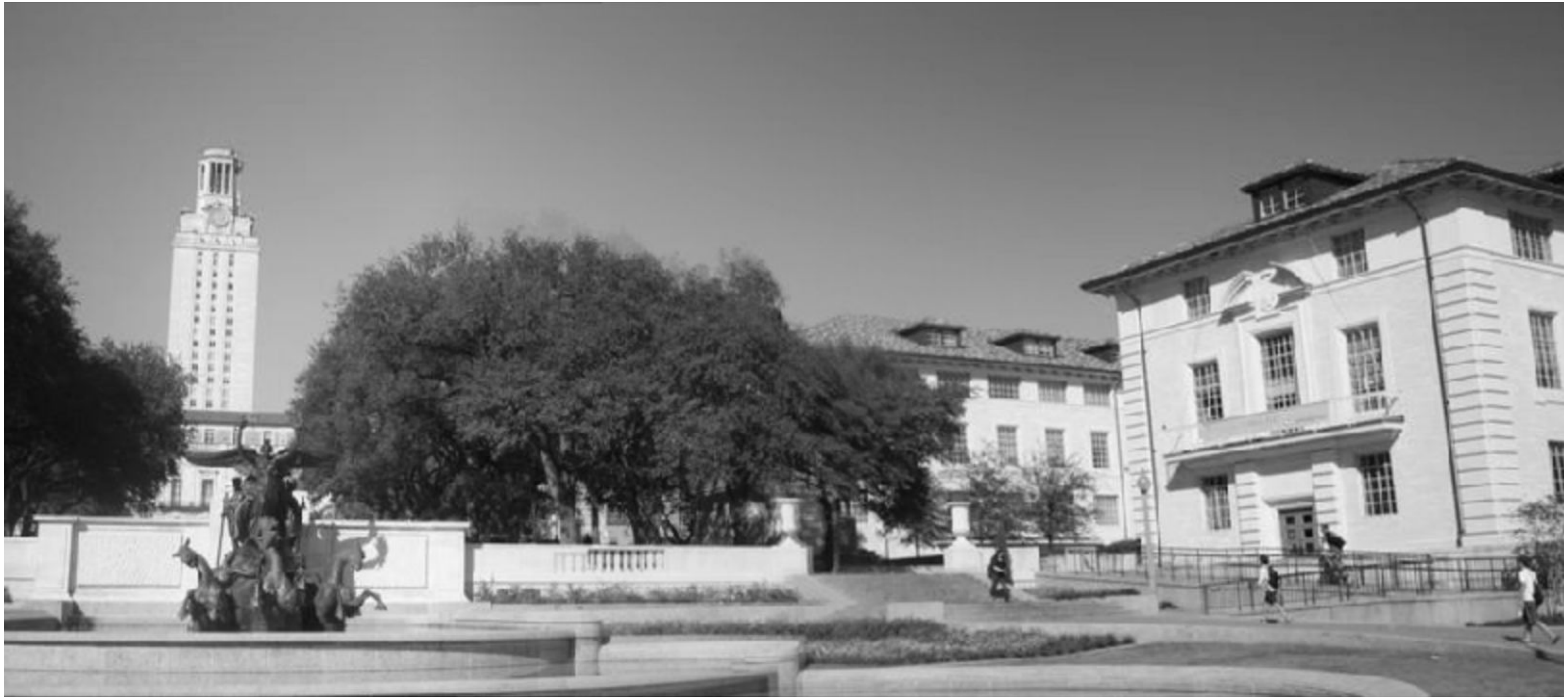
- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation

Panorama

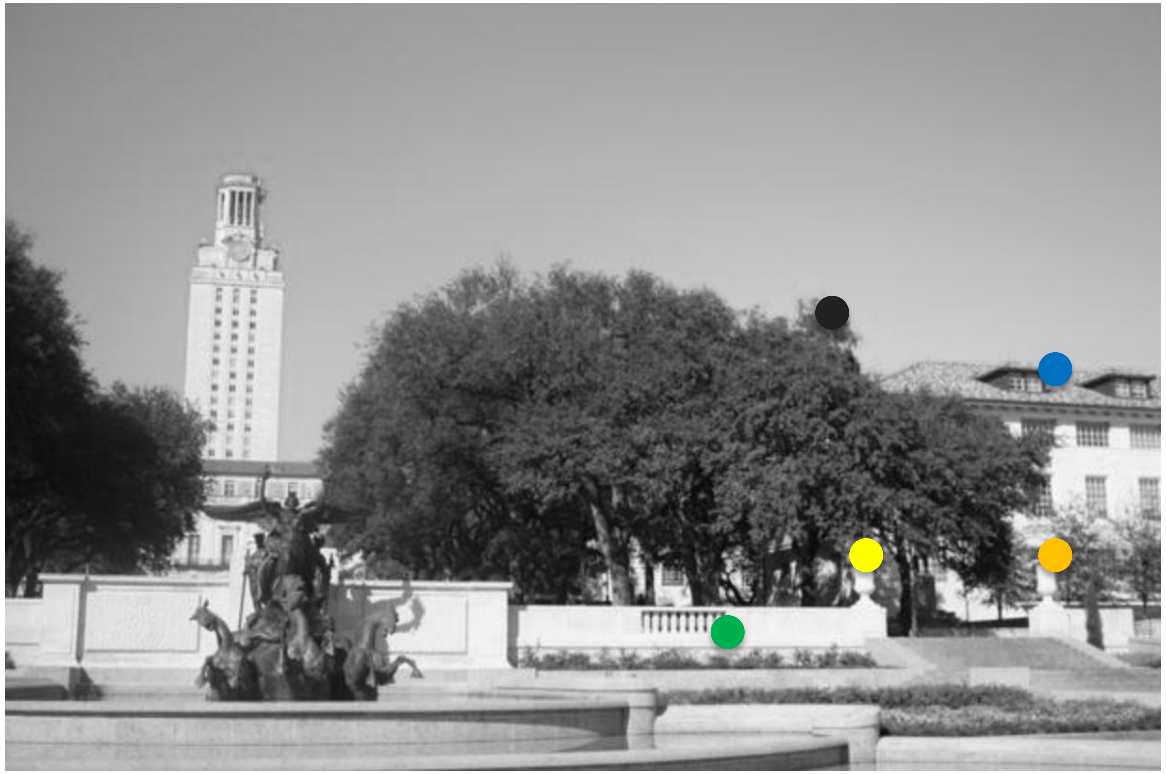


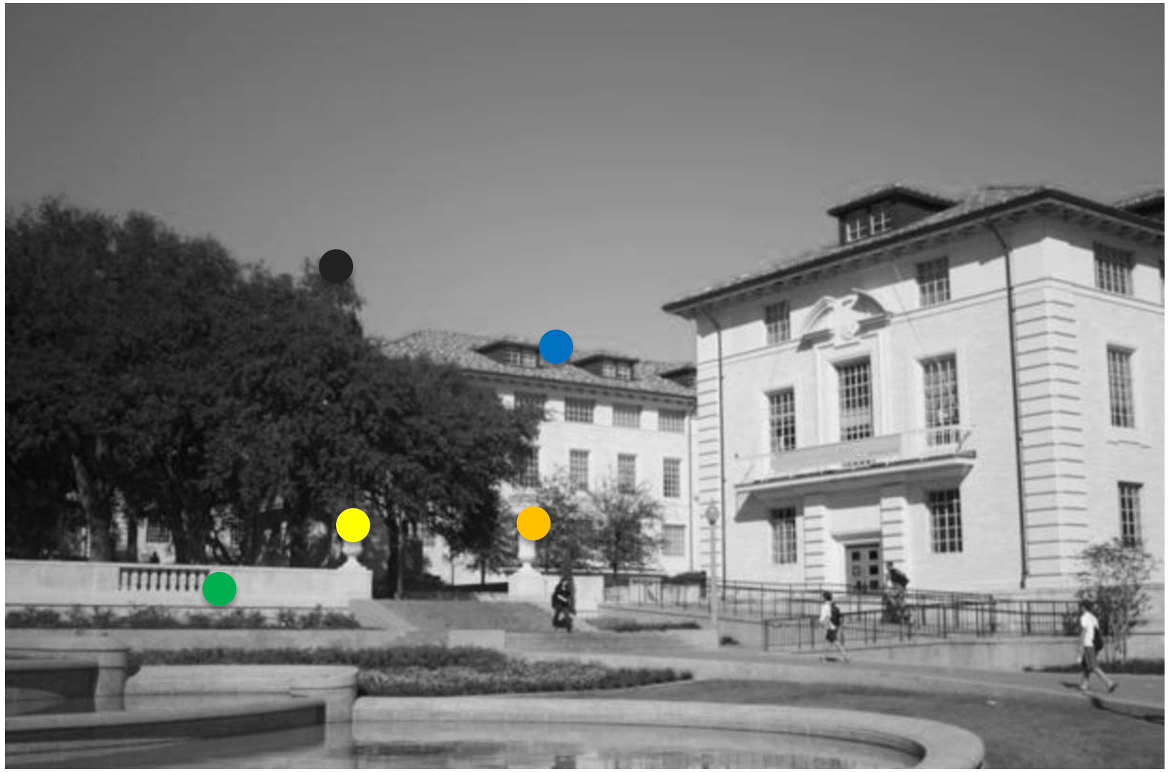


Panorama

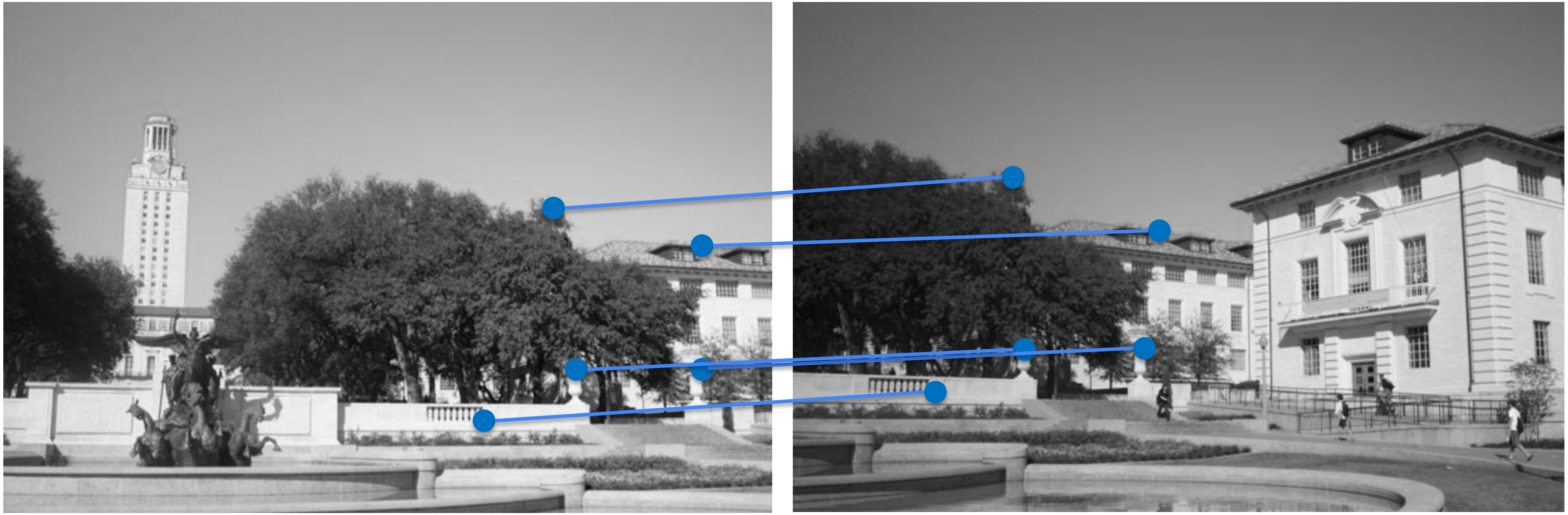


Key insight: leverage corresponding keypoints

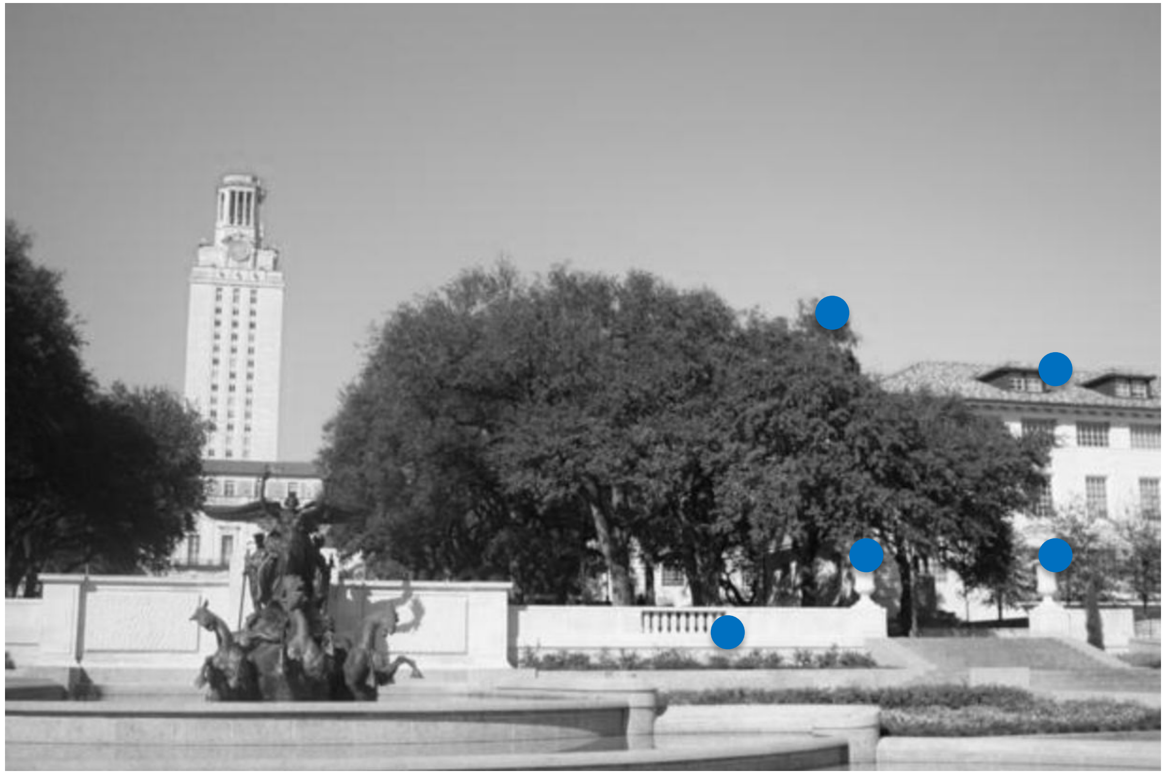


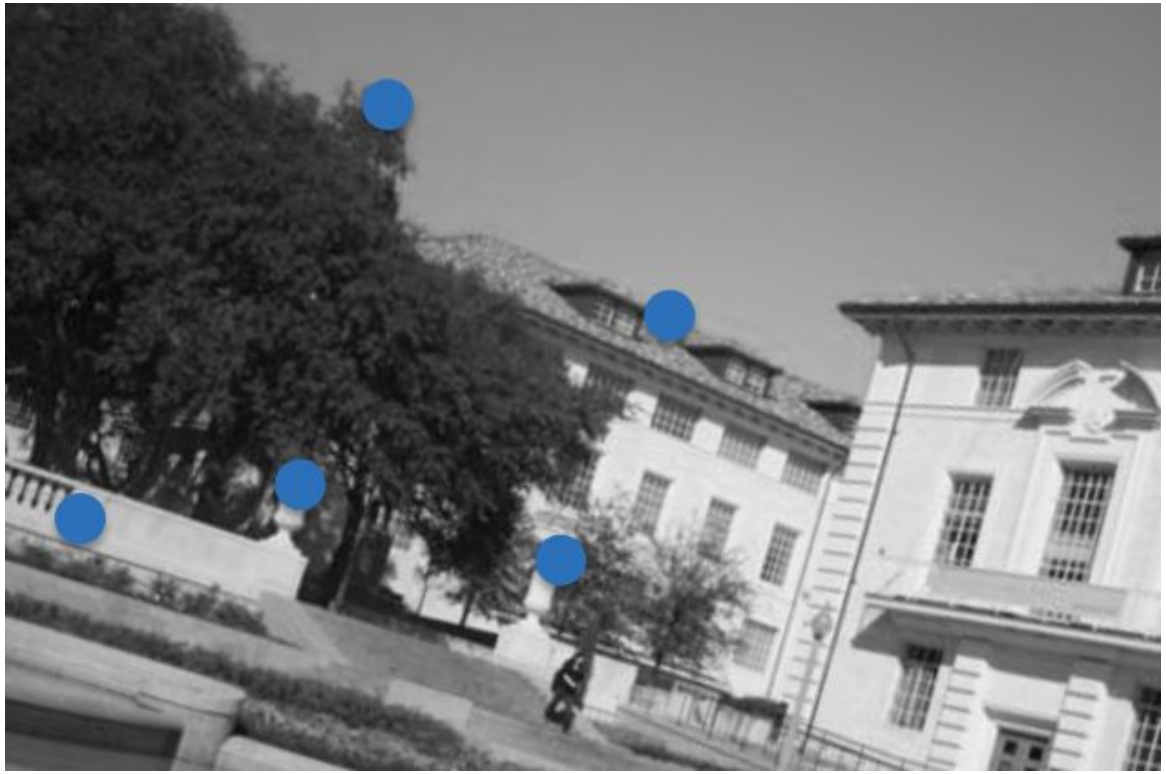


Problem 1: how to match keypoints?



Problem 2: how to fit images?

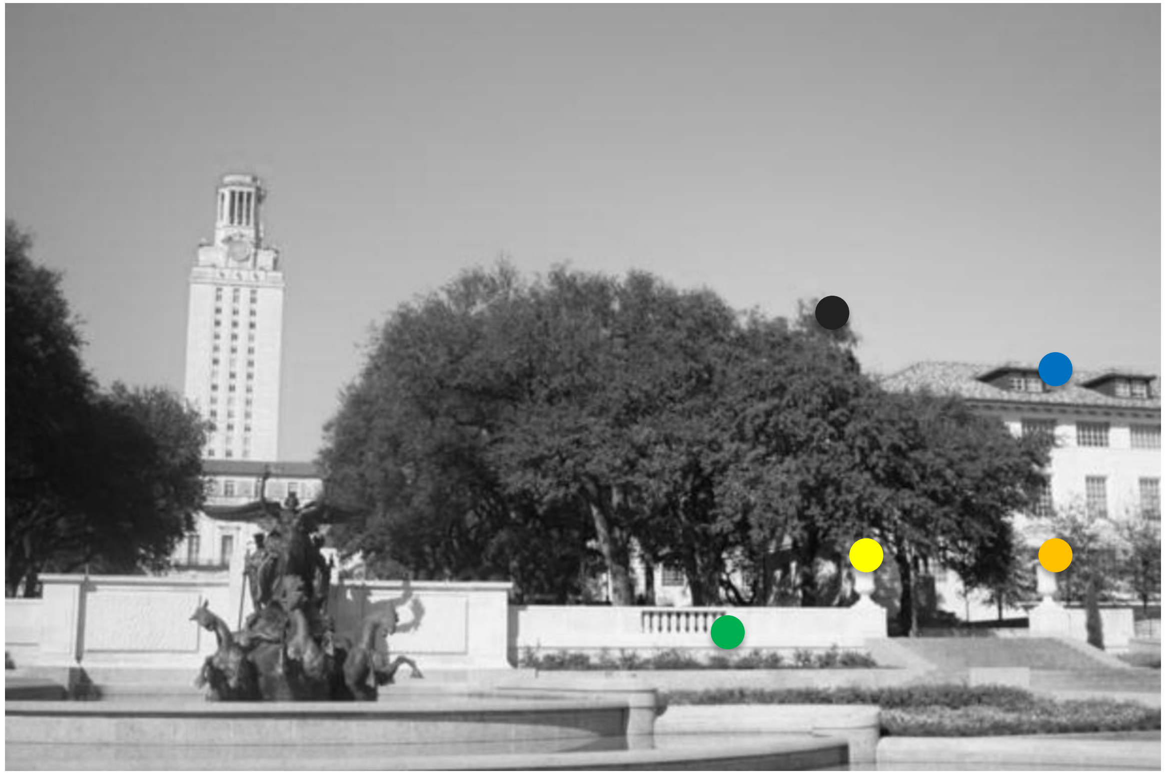


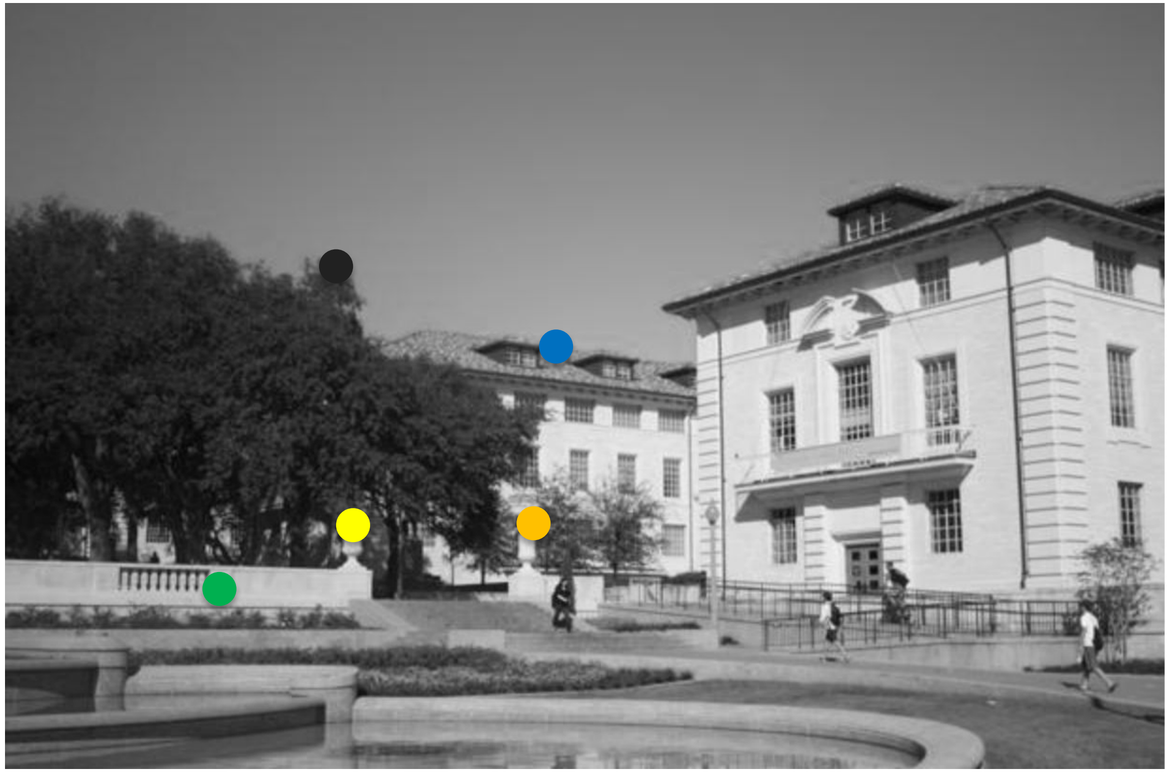


Outline

- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation

How to know if keypoints are “the same”?





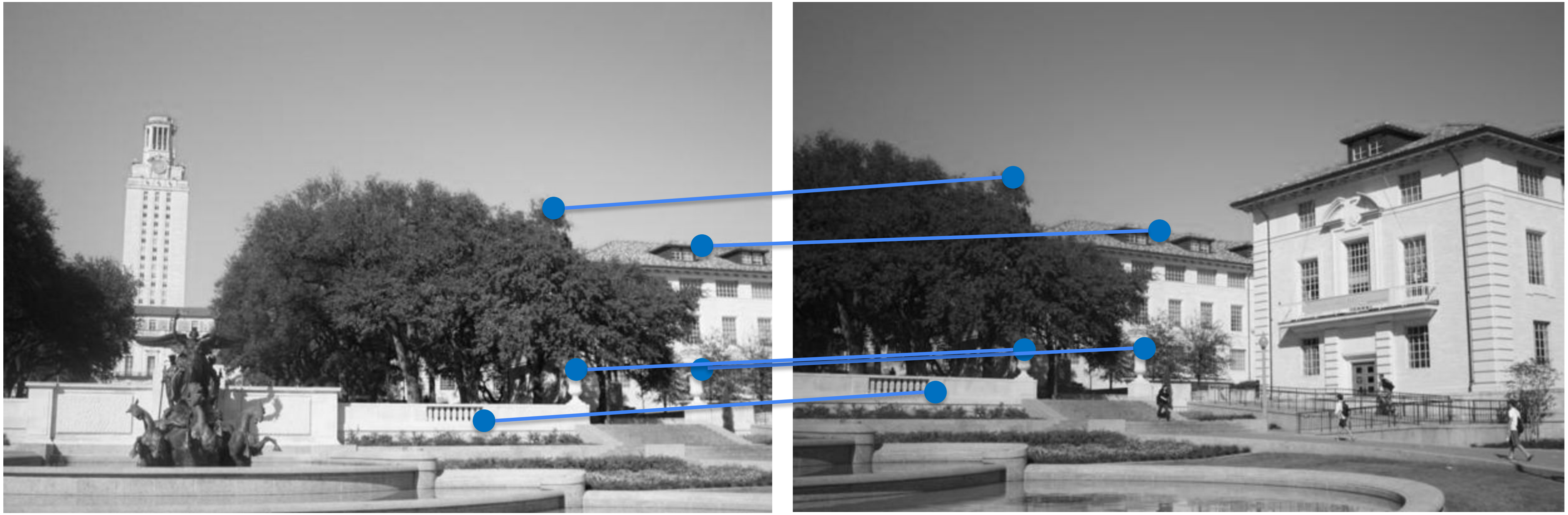
Use keypoint descriptors!

Matching algorithm

- For every keypoint in image_1:
  - **Compute the euclidean distance from every keypoint in image_2**
  - Sort keypoints by distance
  - If the first keypoint’s distance is significantly smaller than the second keypoint’s, it’s a match!
- Return all eligible matches for keypoints in image_1

Try this with one for-loop!

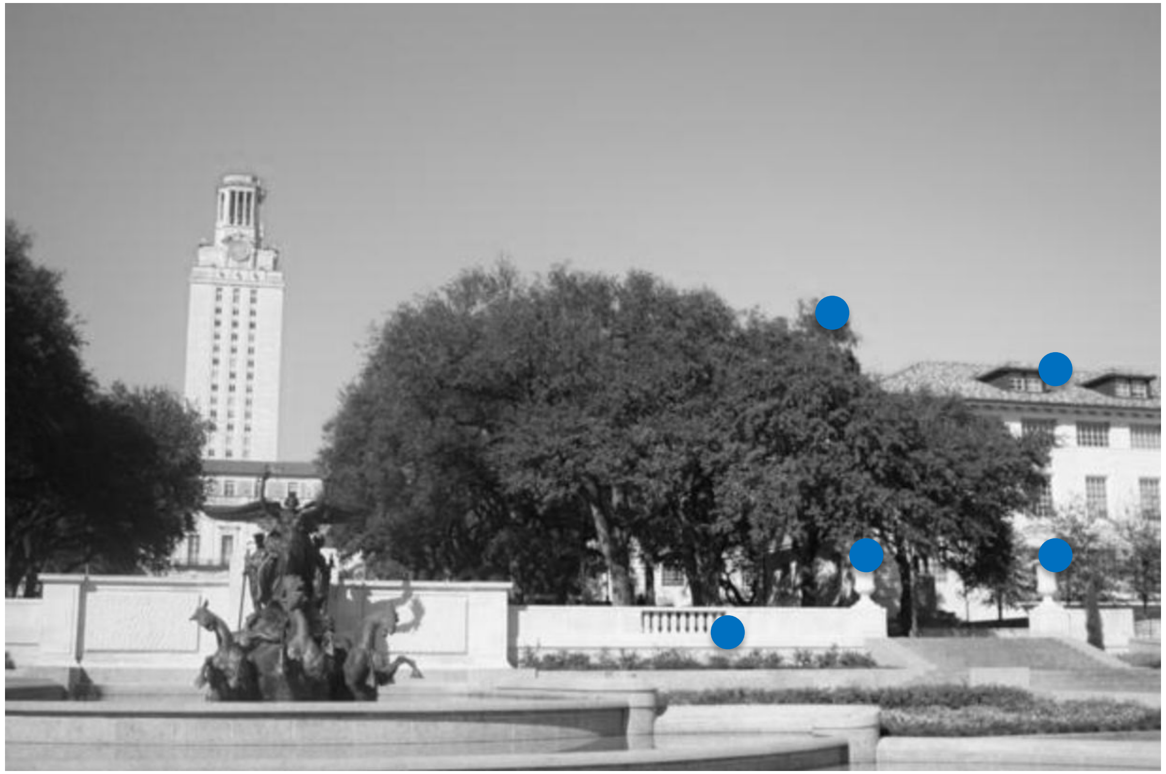
Matching result

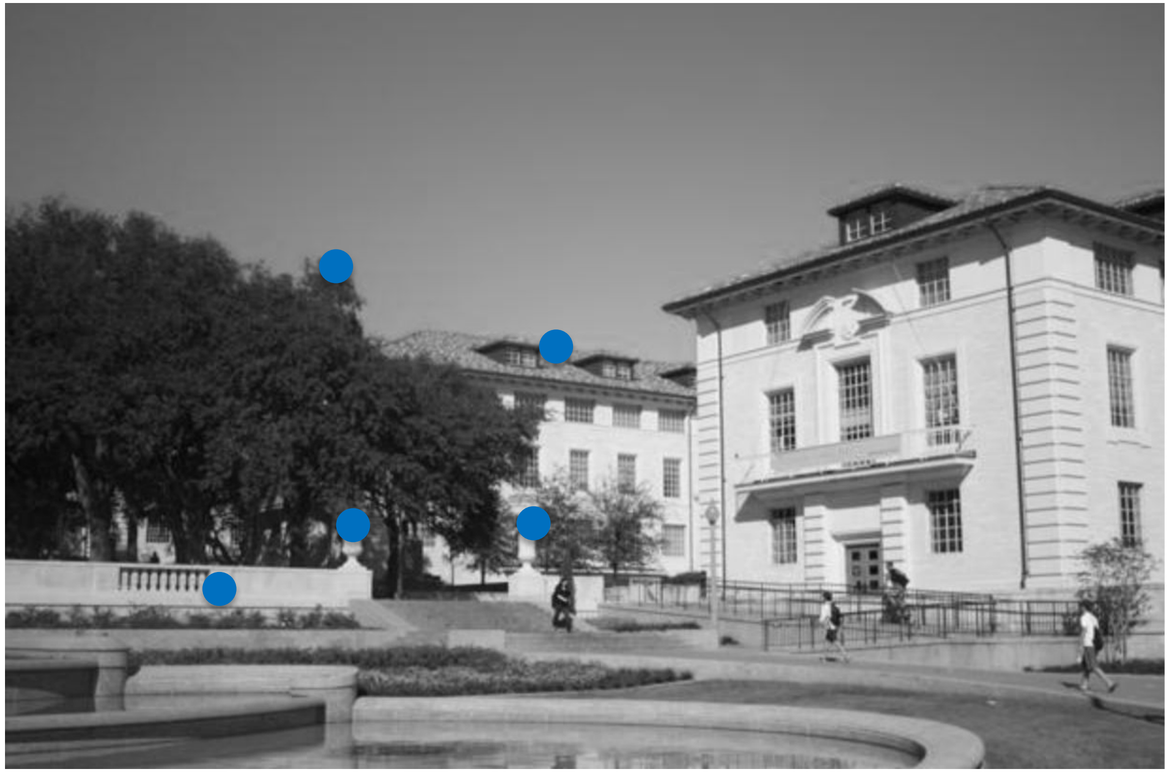


Outline

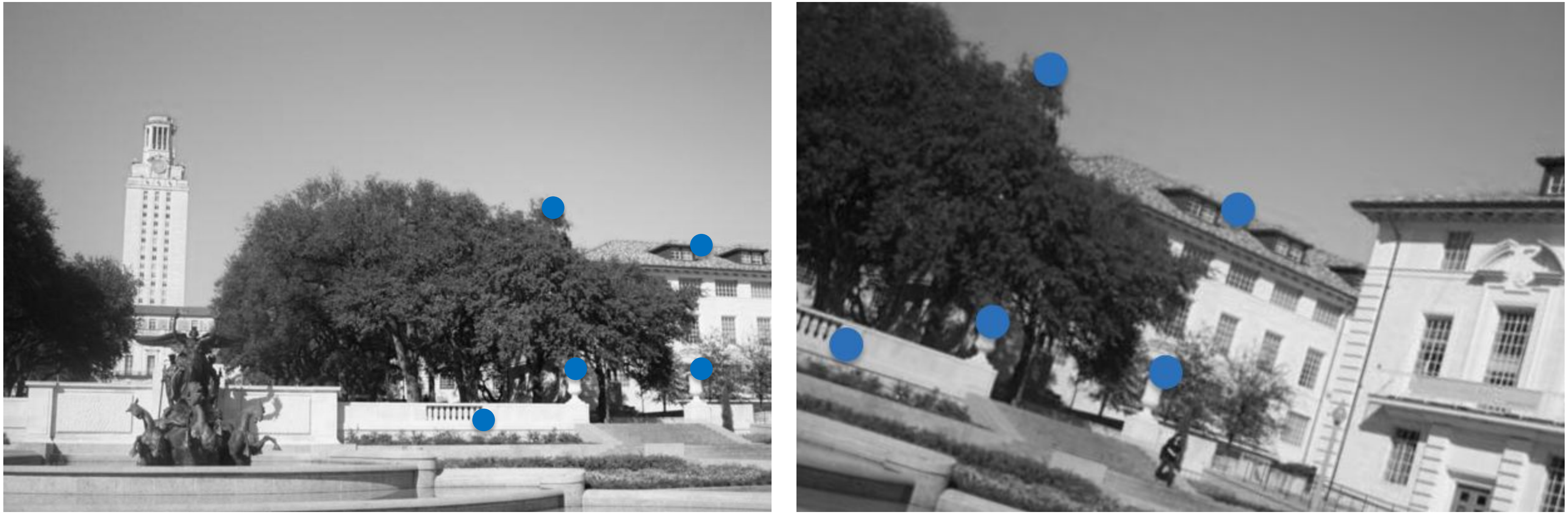
- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation

Easy case: pictures taken from same angle





Hard case: pictures taken from diff angles



Find transformation between matches

Given:

$p1$

		1
		1
		1
		1

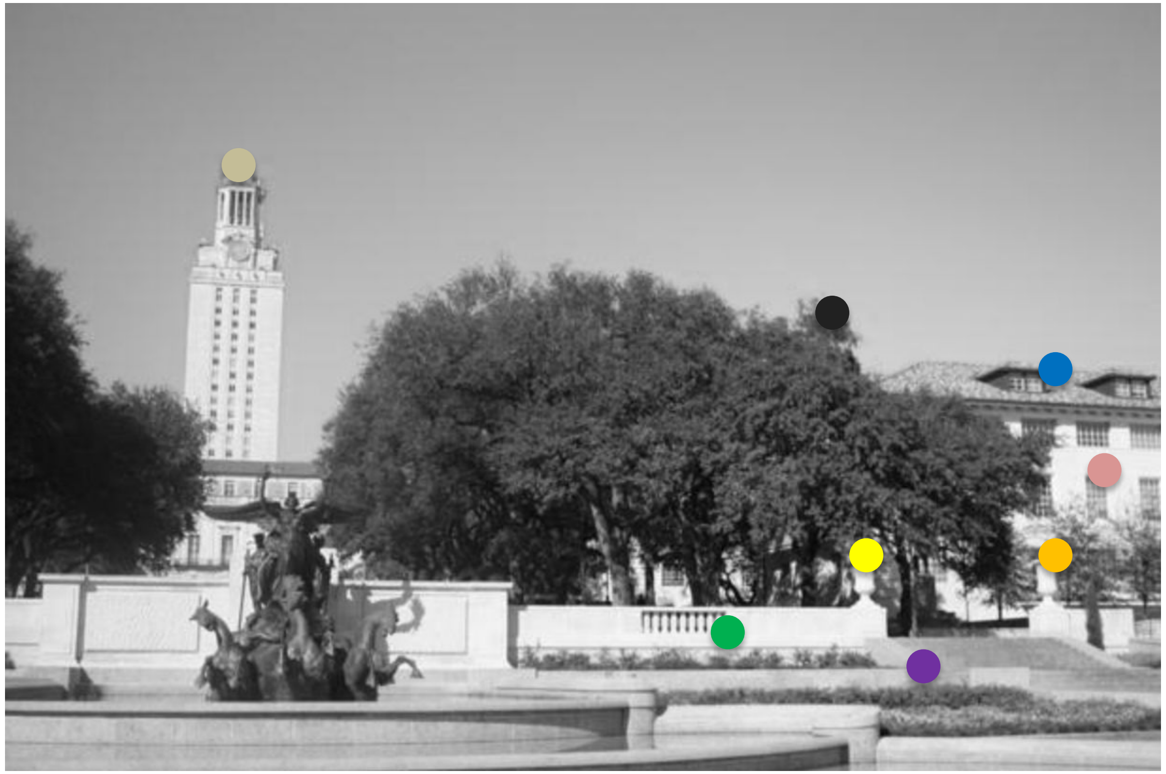
$p2$

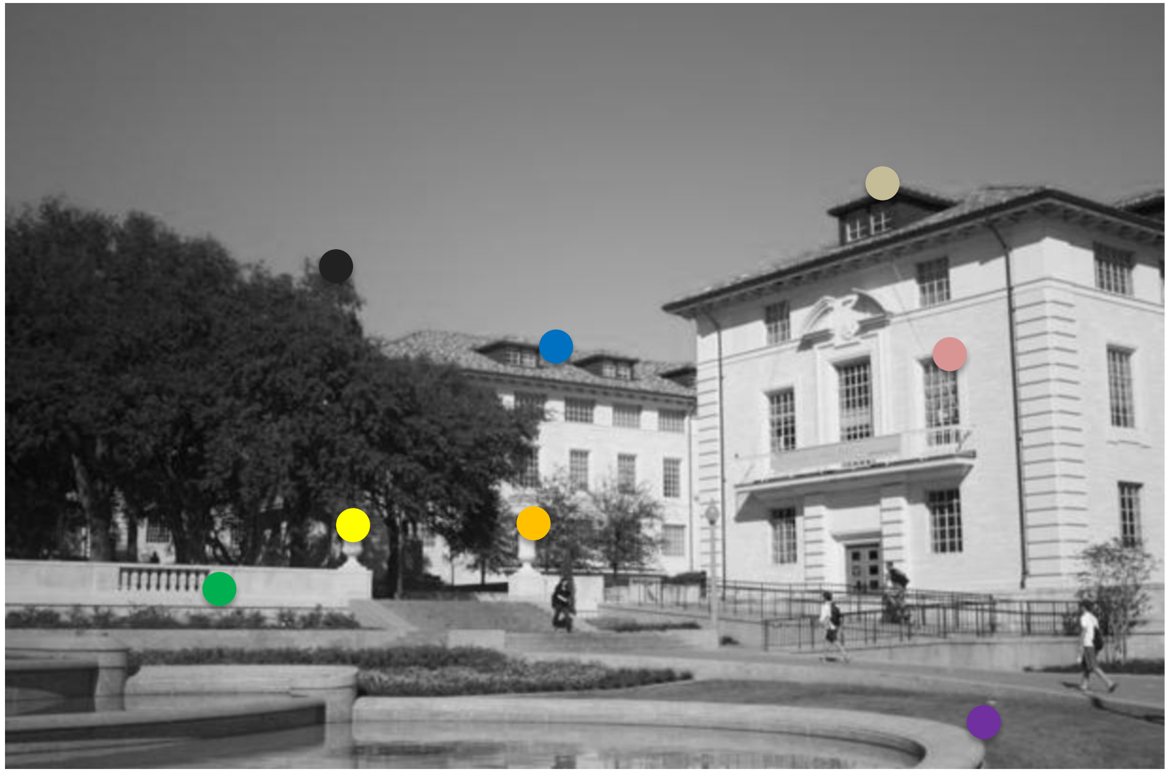
		1
		1
		1
		1

Find transformation matrix H such that:

$$p2 \cdot H = p1$$

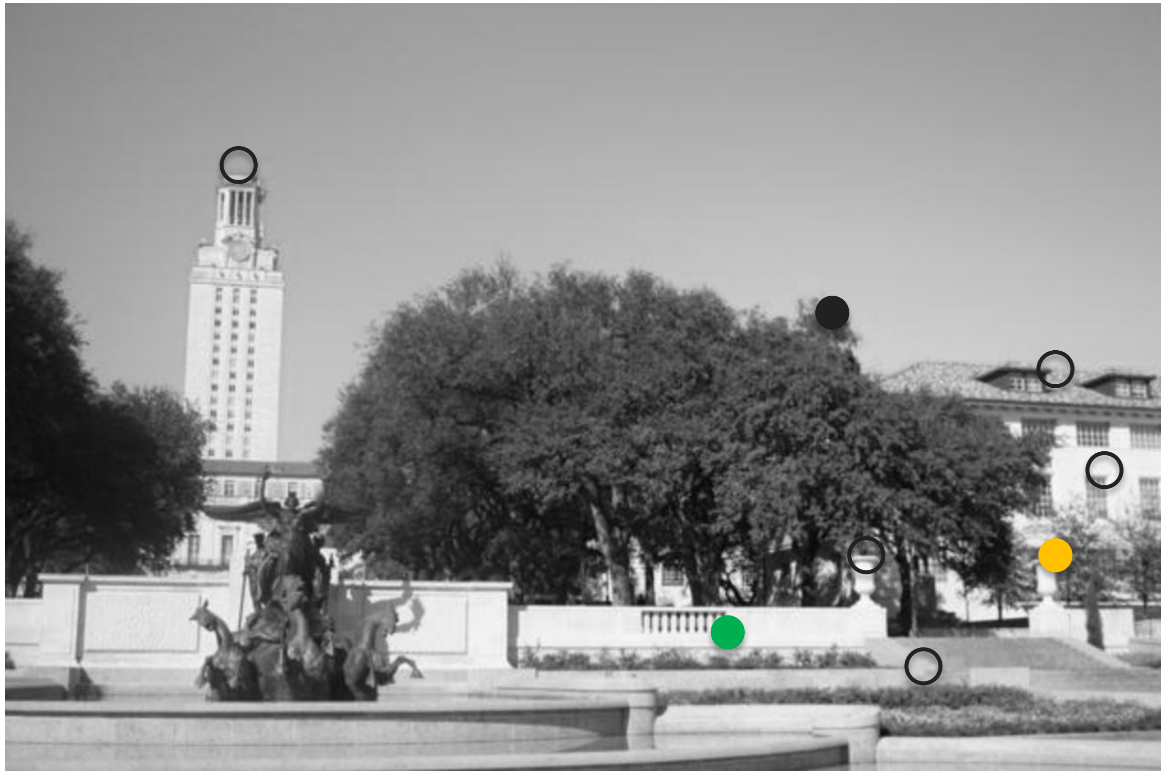
What if we have noisy matches?

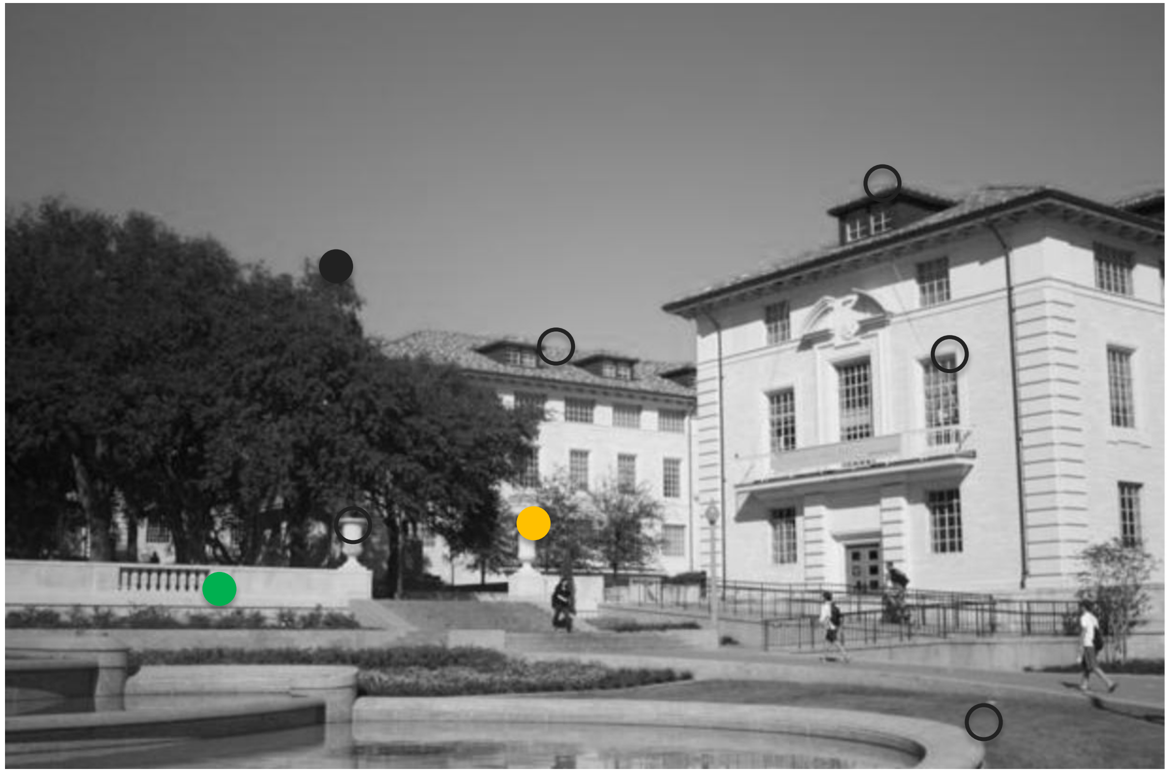




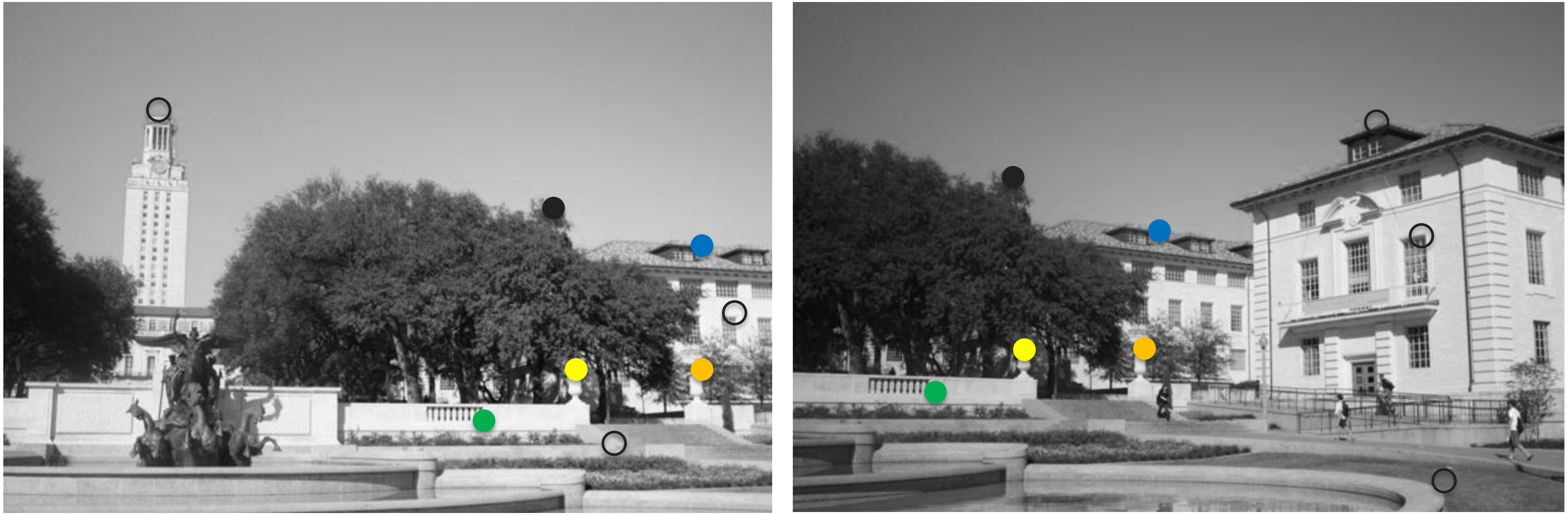
Refine transformation matrix with RANSAC!

Pick subset

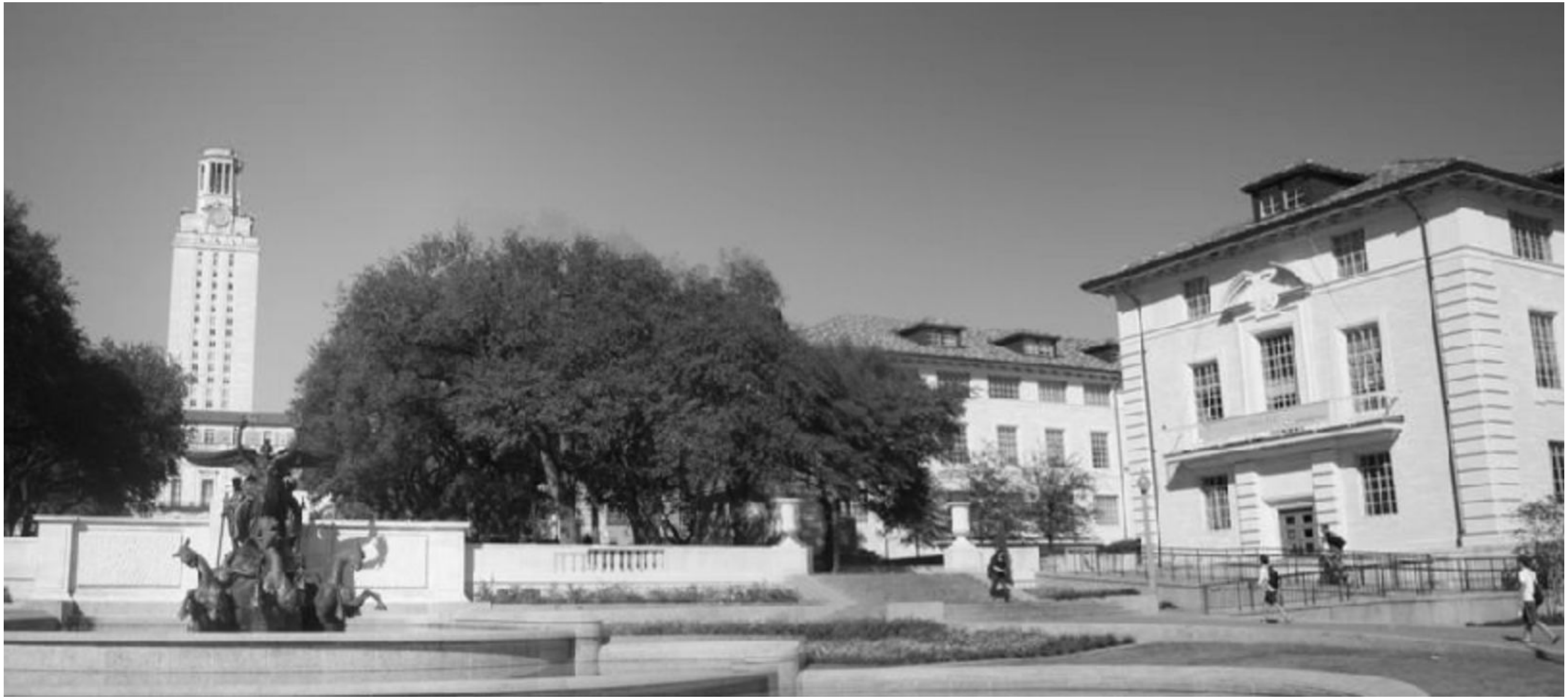




Fit affine matrix and find inliers



Recompute matrix with all inliers and stitch!



Outline

- Quick review of keypoints and RANSAC
- Panorama formulation
- Matching corresponding keypoints
- Stitching images together with affine transformation