Panorama + Practice Questions

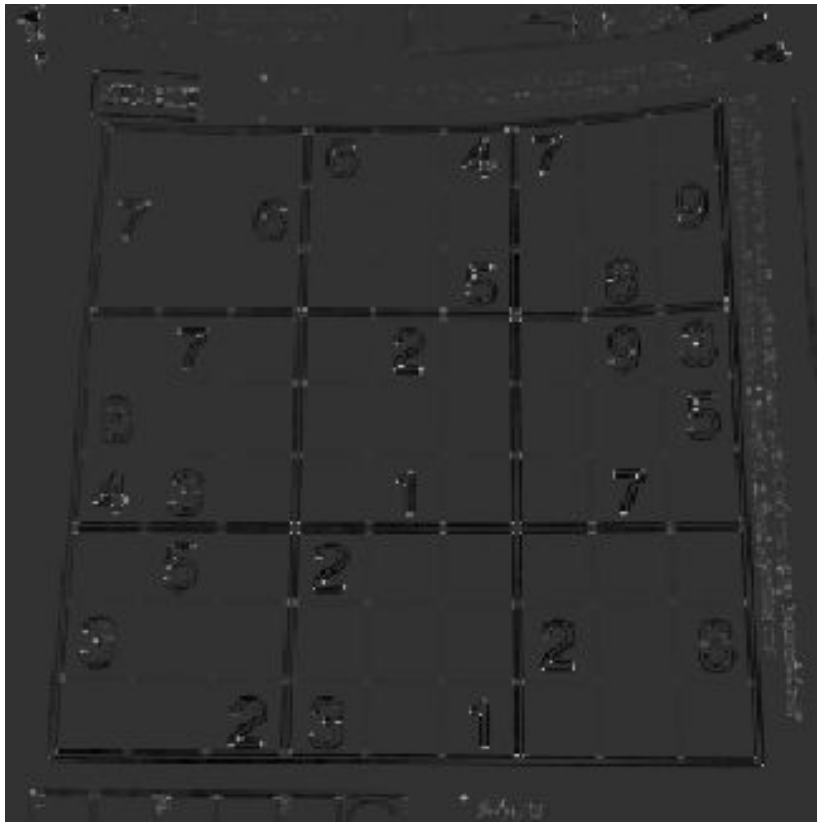
CSE 455: Computer Vision | Jun Wang

April 25th, 2025

Panorama Pipeline (A2 Q1)

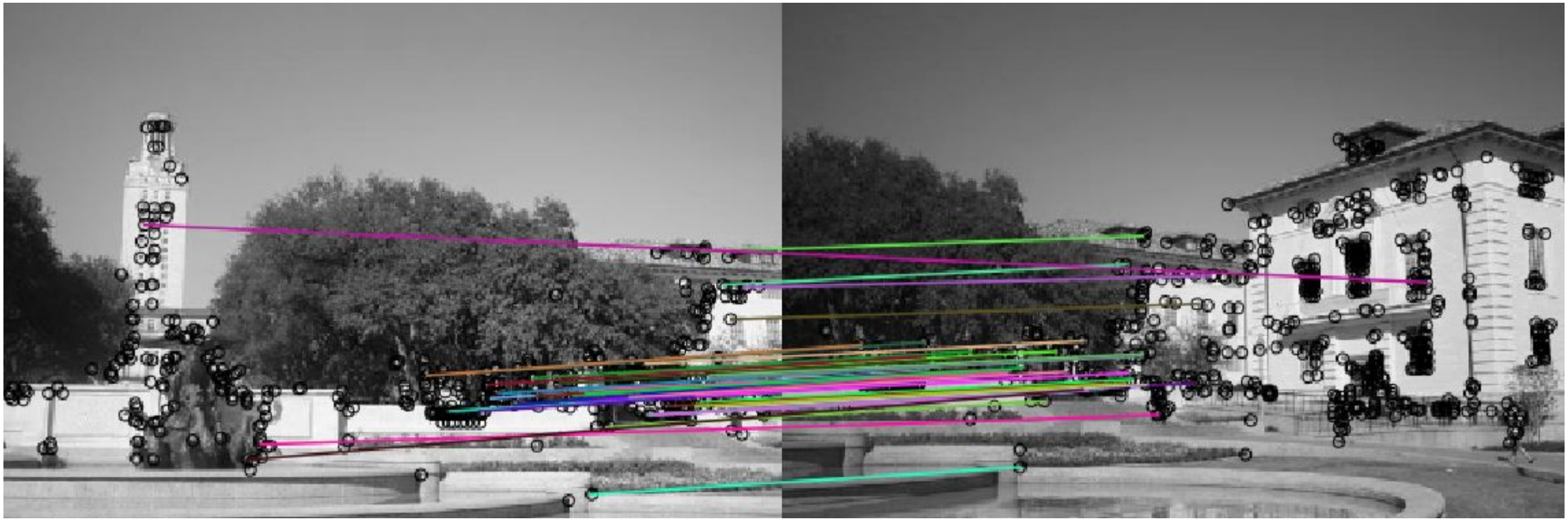
1. Find keypoints

- Harris corner detector
- Two possible solutions (vectorized/iterative)
- Check Sobel axes (sobel_v, sobel_h)
- Follow the formula. Don’t overthink!



2. Build and match feature descriptors

- simple_detector(): Don’t forget to flatten into a 1D array after normalization.
- match_descriptors(): Only accept the match if the ratio (closest / second closest) is strictly less than threshold.

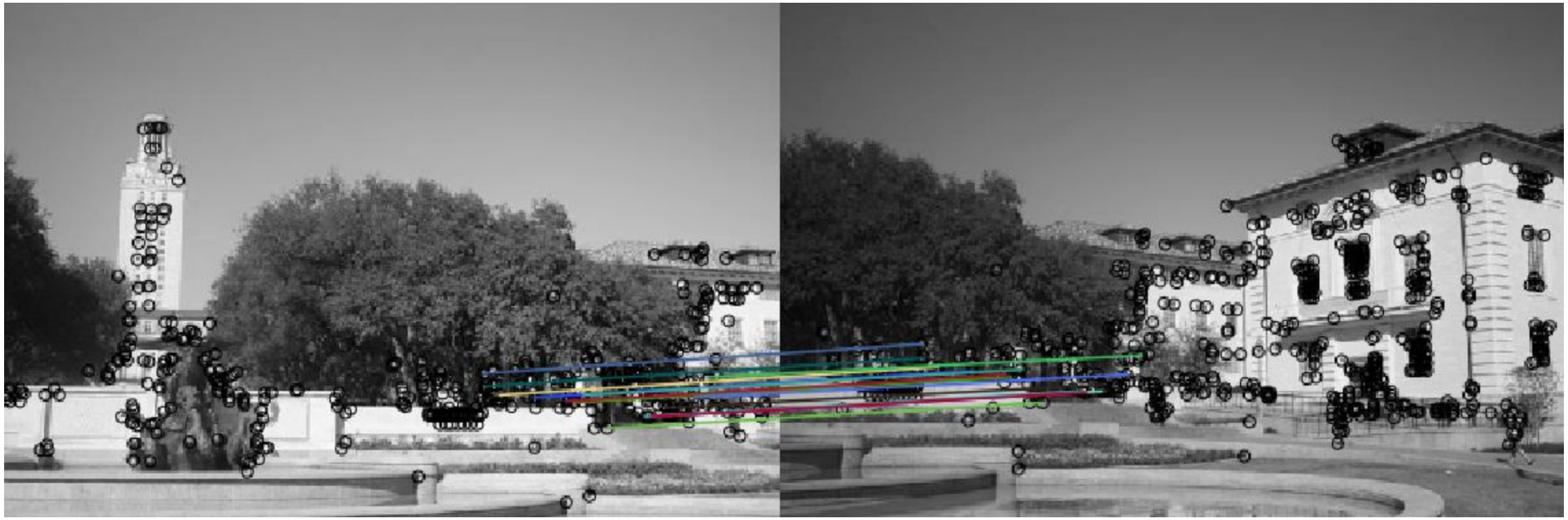


3. Estimate transformation

- Use least-squares method to find the affine transformation matrix that maps points in one image to another.
- Again, don’t overthink. Your code should be very short.
- The panorama does not have to look great!

4. Robust estimation

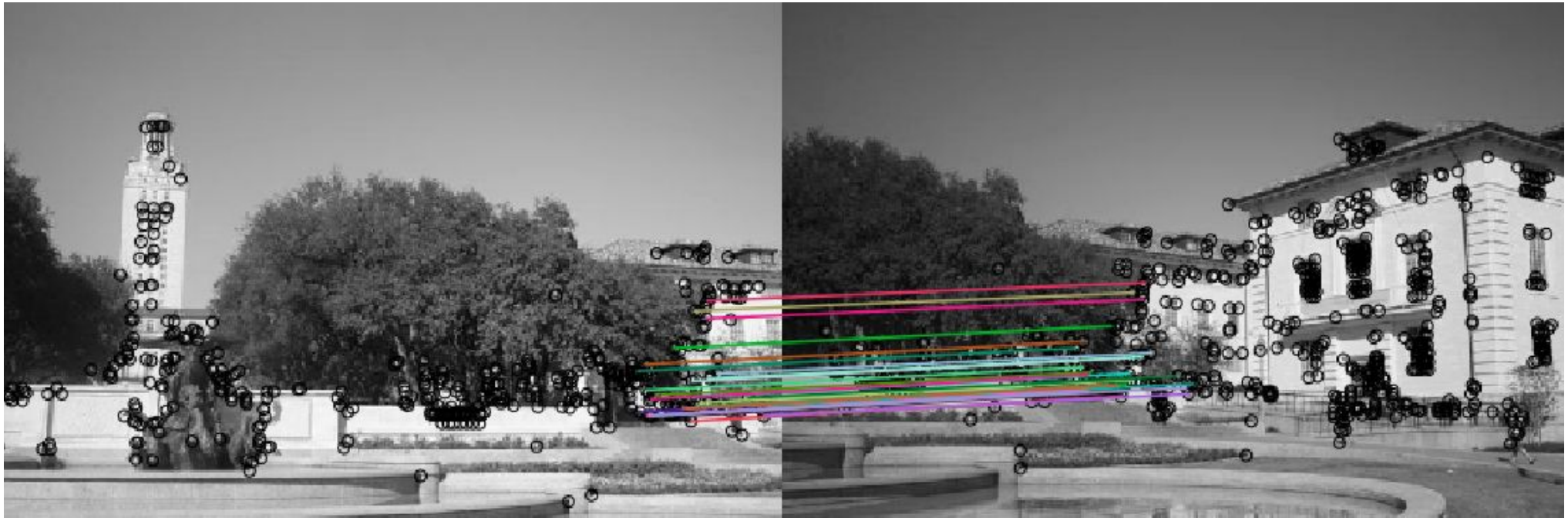
- Apply RANSAC to refine the affine transformation by filtering outliers.
- Don’t forget to keep the transformation with the most inliers.
- Don’t forget to recompute the final affine matrix using all inliers.



5. Panorama stitching

6. HOG descriptor

- Implement a different descriptor to generate a different stitched panorama.
- Check the angle calculation.
- Check the bin assignment. User integer division!
- Normalize the final feature vector.



Practice Questions

Exam Format (Past Quarters)

- True/False (~10 questions)
- Multiple choice (~10 questions)
- Short answers (~5 questions, each with multiple sub-questions)
- Extra credit (~10 points)

Practice exam (from the previous quarter):

https://courses.cs.washington.edu/courses/cse455/25wi/assignments/

We will release another practice exam before the final.

Corners generally have high gradients in a single direction.

- ◯ True
- ◯ False

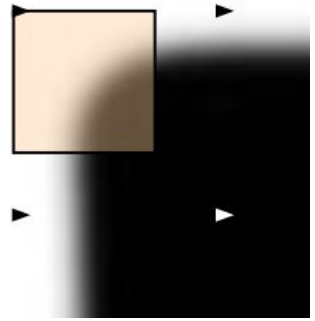
Harris corners are scale-invariant by default

- ◯ True
- ◯ False

The output size of an image after convolution is always the same as the input image.

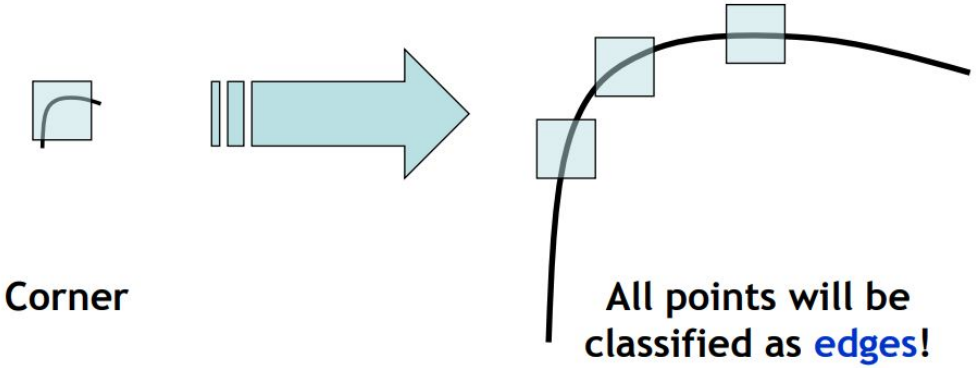
- ◯ True
- ◯ False

Corners generally have high gradients in a single direction.



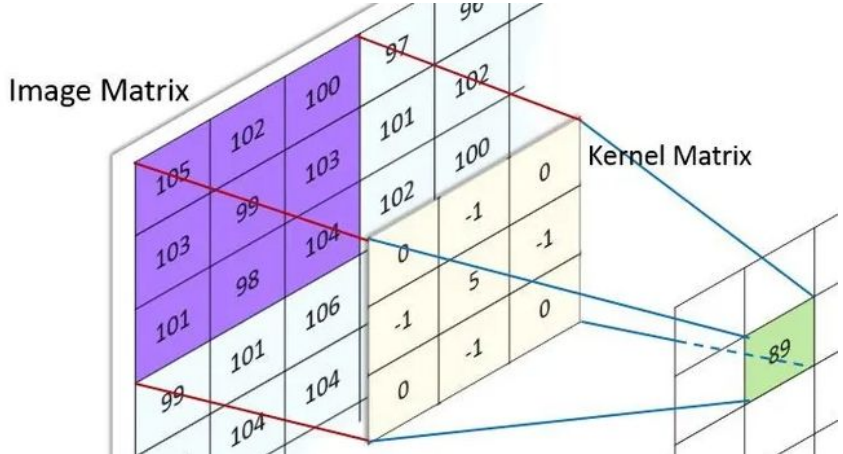
- ○ True
- √ **False**

Harris corners are scale-invariant by default



- ○ True
- √ **False**

The output size of an image after convolution is always the same as the input image.



- ○ True
- √ **False**

Convolutions (2 points). Which of the following affects the output size of convolution operations? (*Choose all that apply*):

- ☐ Kernel Size
- ☐ Padding
- ☐ Stride
- ☐ Activation function

Convolutions (2 points). Which of the following affects the output size of convolution operations? (*Choose all that apply*):

- ☐ Kernel Size
- ☐ Padding
- ☐ Stride
- ☐ Activation function

✅ Kernel size: A bigger kernel covers a larger region, reducing the output size.

✅ Padding: Adding padding increases input size, increasing the output size.

✅ Stride: A larger stride skips more positions, reducing the output size.

❌ Activation function: Only changes values, not spatial dimensions.

Convolutions (2 points). Which of the following affects the output size of convolution operations? (*Choose all that apply*):

- ☐ Kernel Size
- ☐ Padding
- ☐ Stride
- ☐ Activation function

$$\text{Output size} = \left\lfloor \frac{\text{Input size} + 2 \times \text{Padding} - \text{Kernel size}}{\text{Stride}} \right\rfloor + 1$$

Kernels (2 point). Of the four kernels below, which would be best for finding and enhancing horizontal edges and lines in an image? (*Choose the correct answer*)

a)

-1	-1	-1
-1	8	-1
-1	-1	-1

b)

-1	0	1
-2	0	2
-1	0	1

c)

0	-1	0
-2	6	-2
0	-1	0

d)

-1	-2	-1
0	0	0
1	2	1

Kernels (2 point). Of the four kernels below, which would be best for finding and enhancing horizontal edges and lines in an image? (*Choose the correct answer*)

a)

-1	-1	-1
-1	8	-1
-1	-1	-1

b)

-1	0	1
-2	0	2
-1	0	1

c)

0	-1	0
-2	6	-2
0	-1	0

d)

-1	-2	-1
0	0	0
1	2	1

A: Highlight edges for all directions.

B: Highlight vertical edges.

C: Sharpening filter.

D: Highlight horizontal edges.

(vertical intensity change → highlights horizontal edge)

RANSAC (2 points). RANSAC is a powerful method for a wide range of model fitting problems as it is easy to implement. We've seen how RANSAC can be applied to fitting lines. However, RANSAC can handle more complicated fitting problems as well, such as fitting circles.In this problem we will solve for the steps needed to fit circles with RANSAC. What is the minimum number of points we must sample in a seed group to compute an estimate for a uniquely defined circle? (*Choose the correct answer*)

The general equation for a circle with center (h, k) and radius r is:

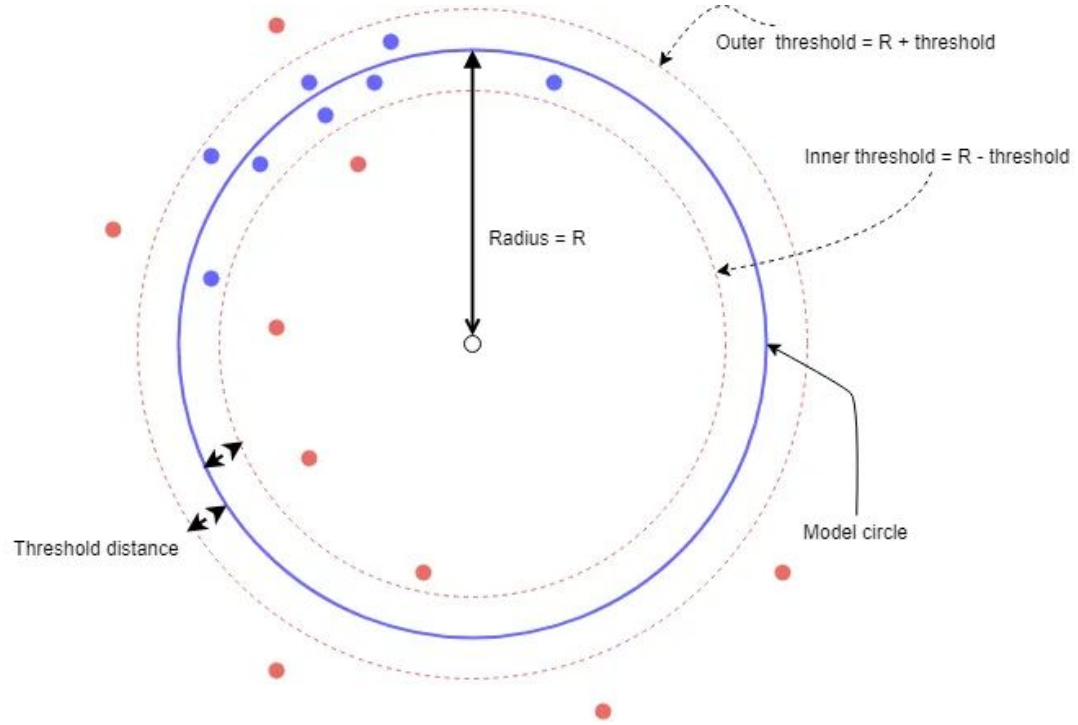
$$(x - h)^2 + (y - k)^2 = r^2$$

RANSAC (2 points). RANSAC is a powerful method for a wide range of model fitting problems as it is easy to implement. We've seen how RANSAC can be applied to fitting lines. However, RANSAC can handle more complicated fitting problems as well, such as fitting circles.In this problem we will solve for the steps needed to fit circles with RANSAC. What is the minimum number of points we must sample in a seed group to compute an estimate for a uniquely defined circle? (*Choose the correct answer*)

The general equation for a circle with center (h, k) and radius r is:

$$(x - h)^2 + (y - k)^2 = r^2$$

3 points because a circle has three degrees of freedom – center (h, k) and radius r.



SIFT is translation invariant, rotation invariant, and scale invariant.

But is it perspective invariant?

Answer here (participation!): https://forms.gle/kP7avRowLbB5CX9w5

No. SIFT assumes a locally planar model.

True perspective invariance for arbitrary 3D scenes from a single image is impossible without extra geometry.

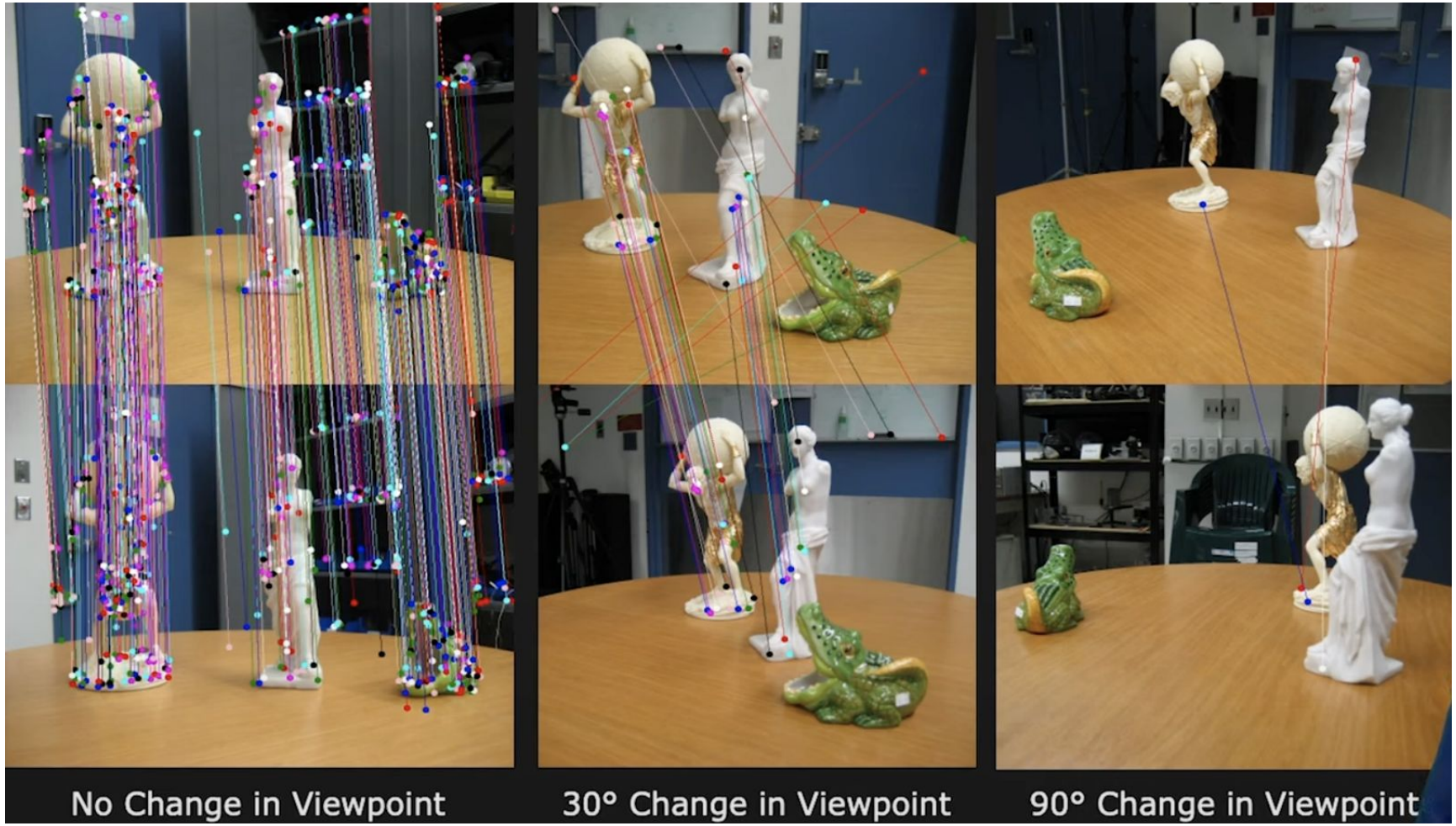


Image Credit: Shree Nayar

Hough Transform (5 points). Your aim is to detect lines in an image using Hough transform. Using Canny edge detector, you have found a set of 5 points that correspond to edges in the first figure below. Using the Hough Transform, transform each point into a line from x, y space to a, b space. Draw these 5 lines in the second image, and find the best values of a, b that define the line. Note that a line is defined as $y = ax + b$.

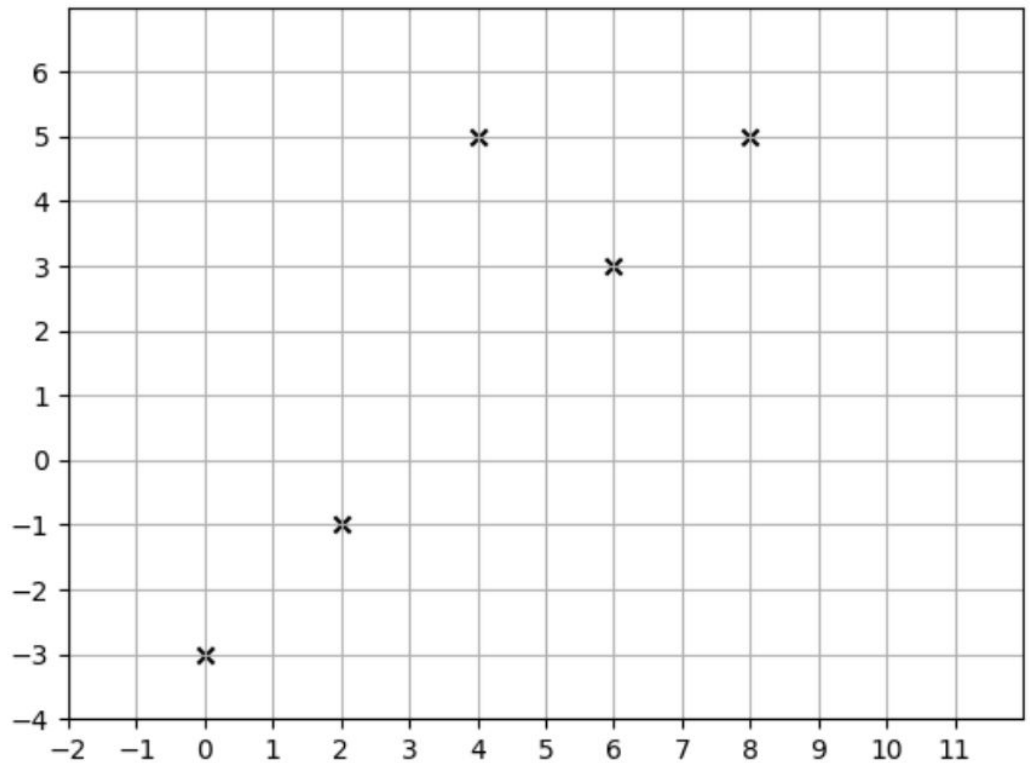


Figure 3: x (horizontal), y (vertical) space where we have the detected points

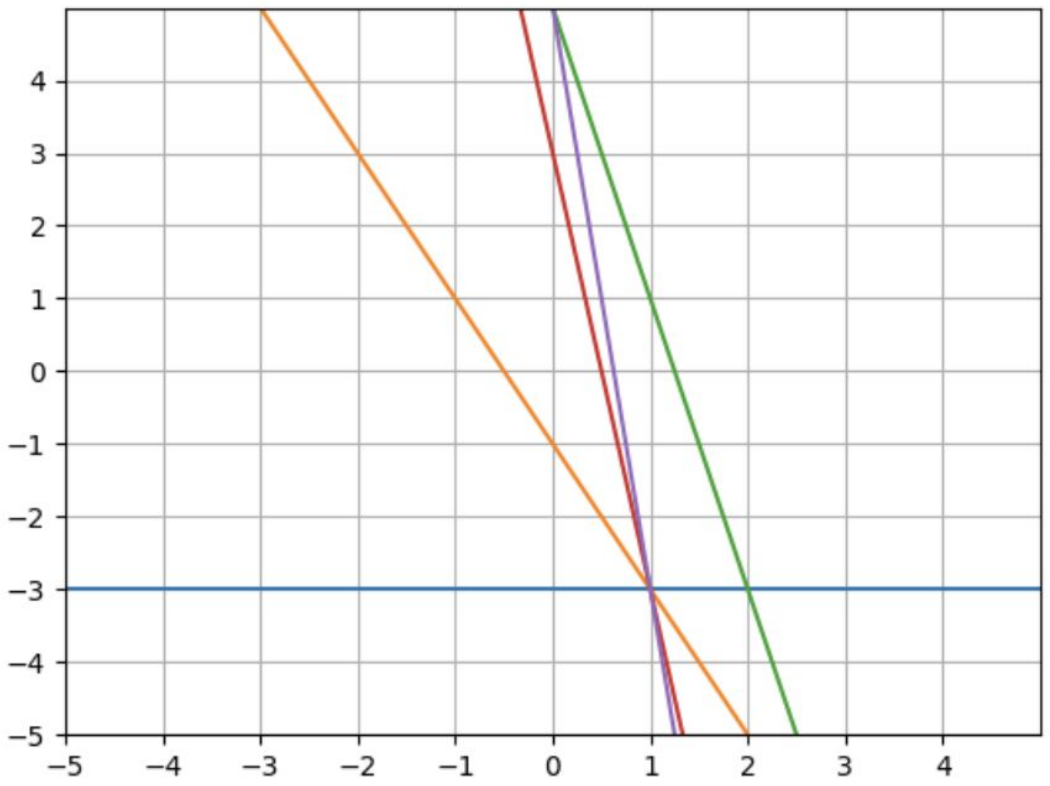


Figure 4: a, b space

Best: a = 1, b = -3

1. Rewrite y = ax + b into b = -ax + y
2. Each point in the (x, y) space defines a line in the (a, b) space.
3. Find the point with the most intersection.