EN3160 Assignment 2 on Fitting and Alignment

Ranga Rodrigo

September 23, 2025

1. In this question, using the knowledge on blob detection, i.e., using Laplacian of Gaussians and scale-space extrema detection, we will detect and draw circles in the sunflower field image. Use the sunflower field image provided:
im = cv.imread('images/the_berry_farms_sunflower_field.jpeg', cv.IMREAD_REDUCED_COLOR_4)
Report the parameters of the largest circles. Report the range of σ values used. You may not use the cv2.HoughCircles function.

2. In this question, we will fit a line and, subsequently, a circle to a set of noisy points that conform to a line and a circle. The code snippet in Listing 1 shows the code to generate this noisy point set X amounting to a circle and a line. We can characterize a line with three parameters a, b, d where $[a, b]^\intercal$ is the unit normal to the line adn d is the perpendicular distance from the origin (see the slide on total least squares line fitting). We can parameterize a circle using the center $[x, y]^\intercal$ and the radius r. Given a set of inliers (points that we know belong to a model, e.g., a circle) we can estimate the best fitting model (best circle) using an optimizer like scipy.minimize (see the documentation). To learn a little of the problem setting in optimization, read the section 1.1 of Boyd and Vandenberghe. We find the inliers within the framework of RANSAC. Consider the noisy pointset X.

   (a) Estimate the line using the RNASAC algorithm (must be coded on your own). Ensure that you apply the constraint $\|[a, b]^\intercal\| = 1$. Carefully select the error (the normal distance to the estimated line) and the number of points that must be in the consensus.

   (b) Subtract the consensus of the best line and estimate the circle that fits the remnant using RANSAC. Carefully select the threshold of error (radial error) and the number of points that must be in the consensus.

   (c) Show in the same plot, the point set, the line estimated from the sample leading to the best estimate, the circle estimated from the sample leading to the best estimate, this sample of three points, the best fit line, line inliers, the the best-fit circle and circle inliers. See Figure 1 for an example.

   (d) What will happen if we fit the circle first?

3. Figure 2 shows an architectural image[1] with a flag [2] superimposed. This is done by clicking four points on a planar surface in the architectural image, computing a homography that maps the flag image to this plane, and warping the flag, and blending on to the architectural image. Carry this out for a couple of image pairs of you own choice. You *may* explain the (non-technical) rationale of your choice.

4. In this questions, we will stitch the two Graffiti image[3] img1.ppm onto img5.ppm.

   (a) Compute and match SIFT features between the two images.

   (b) Compute the homography using your own code within RANSAC and compare with the homography given in the dataset.

   (c) Stitch img1.ppm onto img5.ppm.

Listing 1: Circle Estimation

```
# Generation of a Noisy Point Set Conforming to a Line and a Circle
import numpy as np
```

[1] https://www.robots.ox.ac.uk/ vgg/data/mview/
[2] https://en.wikipedia.org/wiki/Flag_of_the_United_Kingdom
[3] https://www.robots.ox.ac.uk/ vgg/data/affine/

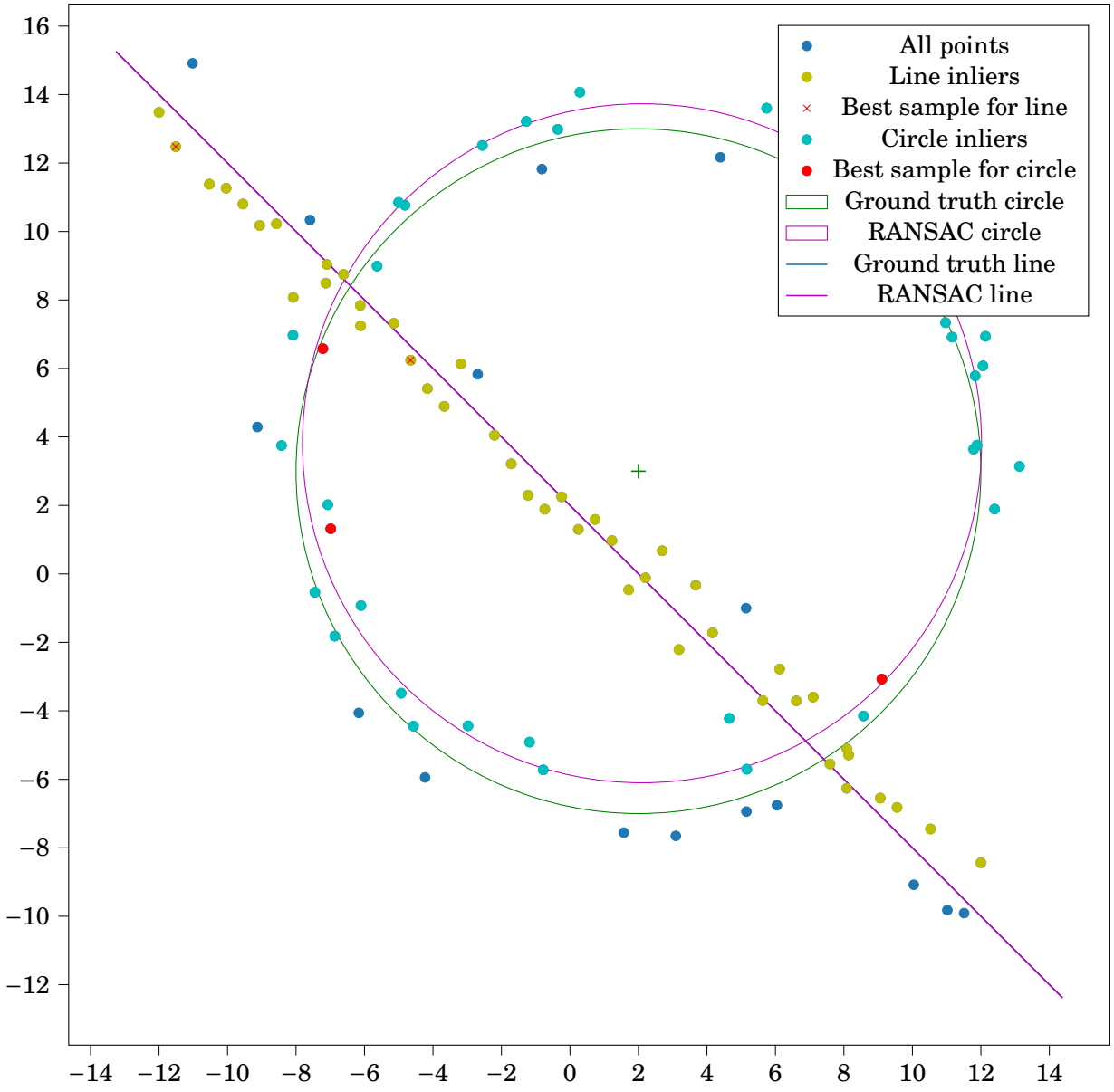

Figure 1: Circle fitting with RANSAC.

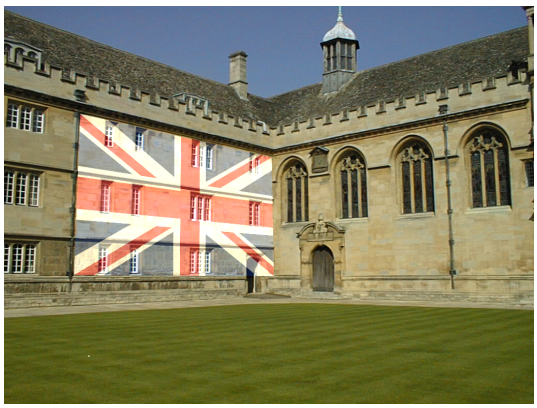

Figure 2: Wadham College image with the British flag superimposed.

```
from scipy.optimize import minimize
from scipy import linalg
import matplotlib.pyplot as plt
import tikzplotlib
# np.random.seed(0)
N = 100
half_n = N//2
```

```
r = 10
x0_gt, y0_gt = 2, 3 # Center
s = r/16
t = np.random.uniform(0, 2*np.pi, half_n)
n = s*np.random.randn(half_n)
x, y = x0_gt + (r + n)*np.cos(t), y0_gt + (r + n)*np.sin(t)
X_circ = np.hstack((x.reshape(half_n,1), y.reshape(half_n,1)))
s = 1.
m, b = -1, 2
x = np.linspace(-12, 12, half_n)
y = m*x + b + s*np.random.randn(half_n)
X_line = np.hstack((x.reshape(half_n,1), y.reshape(half_n,1)))
X = np.vstack((X_circ, X_line)) # All points

fig, ax = plt.subplots(1,1, figsize=(8,8))
ax.scatter(X_line[:,0],X_line[:,1], label='Line')
ax.scatter(X_circ[:,0],X_circ[:,1], label='Circle')
circle_gt = plt.Circle((x0_gt,y0_gt), r, color='g', fill=False, label='Ground truth circle')
ax.add_patch(circle_gt)
ax.plot((x0_gt), (y0_gt), '+', color='g')
x_min, x_max = ax.get_xlim()
x_ = np.array([x_min, x_max])
y_ = m*x_ + b
plt.plot(x_, y_, color='m', label='Ground truth line')
plt.legend()
```

GitHub Profile

You must include the link to your GitHub (or some other SVN) profile, so that I can see that you have worked on this assignment over a reasonable duration. Therefore, make commits regularly. However, I will use only the pdf for grading to save time.

Submission

Upload a report (four pages or less) named as your_index_a02.pdf. Include the index number and the name within the pdf as well. The report must include important parts of code, image results, and comparison of results. The interpretation of results and the discussion are important in the report. Extra-page penalty is two (2) marks per page.