Department of Electronic and Telecommunication Engineering
University of Moratuwa

Assignment II

190610E - Sumanasekara W.K.G.G.

This report is submitted as a partial fulfillment of module EN2550

April 19, 2022

Note: All codes relevant to this assignment can be found in `https://github.com/GevinduGanganath/EN2550/tree/main/Assignment%202`

Question 1

In the first part of the assignment, a circle is estimated using the RANSAC algorithm. Objective is to write a code for RANSAC manually. Figure 1 shows the python code written by me.

```python
def RANSAC_Circle(data_points):
    thres = np.std(data_points)/16  # threshold for RANSAC
    num_iterations = np.log(1 - 0.95)/np.log(1 - (1 - 0.5)**3)
    iterations_done, max_inlier_count, best_model = 0, 0, None

    while iterations_done < num_iterations:
        iterations_done += 1
        np.random.shuffle(data_points)  # randomly selecting 3 data points
        sample_data = data_points[:3]
        xc,yc,radius,_ = cf.least_squares_circle((sample_data)) # estimating a circle with selected data points
        center = (xc, yc)
        error = np.abs(radius - np.sqrt(np.sum((center - data_points[3:])**2, axis=1))) # computing error of remaining data points
        inliers = error <= thres # camparing the error with threshold
        inlier_count = np.count_nonzero(inliers) # number of inliers
        if inlier_count > max_inlier_count: # selecting the best model
            max_inlier_count = inlier_count
            inlier_points = []
            for index, inlier in enumerate(inliers): # filtering the inlier points
                if inlier == True:
                    inlier_points.append(data_points[3:][index])
            inlier_points = np.array(inlier_points)
            best_model = (center, radius, data_points[:3], inlier_points)

    return best_model
```

Figure 1: Code snip of RANSAC algorithm

RANSAC parameters were set as follows:

- Threshold: Standard deviation of data points divided by 16 (selected by trial and error)
- Number of samples (s): 3
- Probability (p): 0.95
- Outlier ration (e): 0.5
- Number of iterations $= \frac{log(1-p)}{log(1-(1-e)^s)} \approx 22$

Data points are shuffled at each iteration and the first three data points were selected to estimate the circle. *least squares circle* function of the *circle fit* python library was used to compute the center and radius. Then the distance between the center and the remaining data points are calculated. By comparing those values with the threshold we can count the number of inliers. The iteration which gives the highest number of inliers was determined as the best model.

Figure 2 shows the results. Circle obtained by RANSAC algorithm (red) is giving better results than the inbuilt least square fitting function.

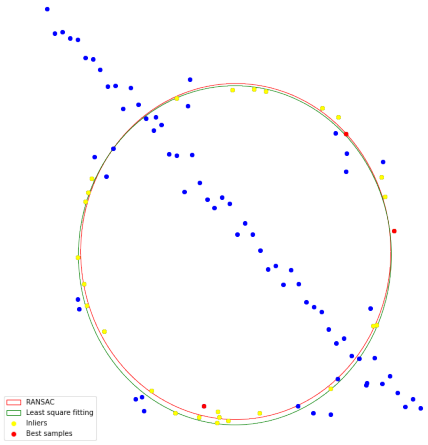

Figure 2: Results of RANSAC