Assignment 2

Andrew ID: sajaved

Name: Syed Ashar Javed

Q1

1.a) $$\bar{\mu}_{t+1} = \mu_{t+1} = g(\overbrace{x_t, y_t, \theta_t}^{\mu_t}, \overbrace{d_{t+1}, \alpha_{t+1}}^{u_{t+1}})$$

$$= \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} + \begin{pmatrix} d_{t+1}\cos\theta_t \\ d_{t+1}\sin\theta_t \\ \alpha_{t+1} \end{pmatrix} = \mu_t + \begin{pmatrix} d_{t+1}\cos\theta_t \\ d_{t+1}\sin\theta_t \\ \alpha_{t+1} \end{pmatrix}$$

1.b) $$\bar{\Sigma}_{t+1} = G_{t+1}\Sigma_t G_{t+1}^T + U_{t+1} R_{t+1} U_{t+1}^T$$

where,

$$G_{t+1} = \frac{\partial g(\mu_t, u_{t+1})}{\partial \mu_t} = \begin{pmatrix} 1 & 0 & -d_{t+1}\sin\theta_t \\ 0 & 1 & d_{t+1}\cos\theta_t \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{t+1} = \mathrm{diag}(\sigma_x^2, \sigma_y^2, \sigma_\alpha^2)$$

$$\Sigma_t \Rightarrow \mathcal{N}(0, \Sigma_t)$$

$$U_{t+1} = (R_{global\_to\_local})^{-1}$$

where,

$R_{global\_to\_local}$ is a transformation matrix from global coordinates to local robot coordinates

$$R_{global\_to\_local} = \frac{\partial X_{local}}{\partial X_{global}} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$X_{local} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} X_{global}$$

(a) Solution for 1.a and 1.b

1.c) $$\begin{pmatrix} l_x \\ l_y \end{pmatrix} = \begin{pmatrix} x_t \\ y_t \end{pmatrix} + \begin{pmatrix} (r_t + n_r)\cos(\beta_t + n_\beta + \theta_t) \\ (r_t + n_r)\sin(\beta_t + n_\beta + \theta_t) \end{pmatrix}$$

1.d) $$z_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \mu_t^\theta \end{pmatrix}$$

where,

$$q = \delta^T \delta$$

$$\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \overbrace{\bar{\mu}_j^x}^{l_x} - \bar{\mu}_t^x \\ \underbrace{\bar{\mu}_j^y}_{l_y} - \bar{\mu}_t^y \end{pmatrix}$$

The predicted measurement is ~~the difference~~ computed by adding the relative measurement to the estimated robot location (noise terms omitted)

$$\begin{pmatrix} l_x \\ l_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_j^x \\ \bar{\mu}_j^y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_t^x \\ \bar{\mu}_t^y \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\beta_t + \bar{\mu}_t^\theta) \\ r_t^i \sin(\beta_t + \bar{\mu}_t^\theta) \end{pmatrix}$$

1.e) $$l_x = x_t + r_t \cos(\beta_t + \theta_t)$$
$$l_y = y_t + r_t \sin(\beta_t + \theta_t)$$

$$r_t = \sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}$$

$$\beta_t = \tan^{-1}\left(\frac{l_y - y_t}{l_x - x_t}\right) - \theta_t$$

$$H_p = \begin{pmatrix} \partial\beta_t/\partial x_t & \partial\beta_t/\partial y_t & \partial\beta_t/\partial\theta_t \\ \partial r_t/\partial x_t & \partial r_t/\partial y_t & \partial r_t/\partial\theta_t \end{pmatrix}$$

(a) Solution for 1.c, 1.d and 1.e

1.f) $$H_l = \begin{pmatrix} \partial\beta_t/\partial l_x & \partial\beta_t/\partial l_y \\ \partial r_t/\partial l_x & \partial r_t/\partial l_y \end{pmatrix}$$

where the expression for β, r and l_x, l_y are given in the previous part

(a) Solution for 1.f. We do not calculate the measurement Jacobian with respect to landmarks other than itself because we assume that the landmarks are independent of others and the cross covariance terms will not appear in the Jacobian.

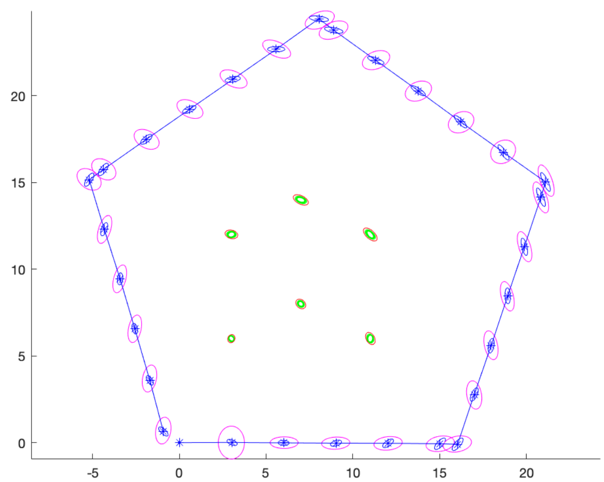
Q2

Part a

The no of landmarks for the given data is 6.

Part b

The code given in files has been modified and the results with the given noise variable values is shown below:



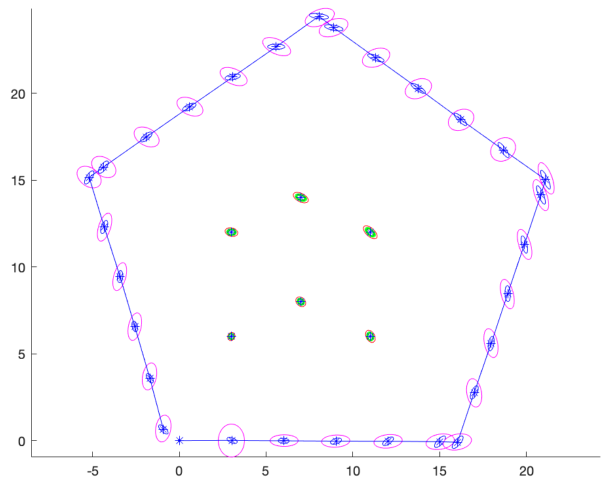
(a) Results with default parameters

Part c

The EKF algorithm increases the variance of the poses and landmarks in the prediction step and decreases the variance in the update step. This is shown in the ellipses from the figure for both the pose and the landmarks. The reduction in variance and the mean estimate of the state vector is dependent on the Kalman gain which denotes the weightage the model gives to the measurement relative to the previous estimate. If the covariances of the estimated values are high, the Kalman gain puts a high weight on the measurement. Hence, in the prediction step, the estimates of the mean and covariance are refined on the basis of the Kalman gain which leads to better trajectories and map.

Part d

The ground truth position of the landmarks are plotted in the figure below and they do lie within its corresponding ellipse. This means that the true location is within the estimated covariance ellipse obtained form the EKF model.



(a) Results with default parameters

The Euclidean distance of the 6 landmarks are as follows:
0.0023 0.0043 0.0023 0.0026 0.0018 0.0042

The Mahalanobis distance of the 6 landmarks are as follows:
0.0596 0.0779 0.0538 0.0691 0.0325 0.1120

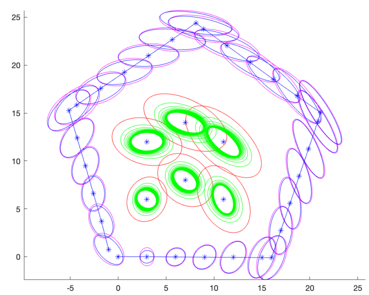
Since Mahalanobis distance is similar to Euclidean distance scaled by the inverse covariance matrix, it can measure distances between multivariate data points by using the inverse covariance matrix. While Euclidean distance only measures the geometric distance between two points, the Mahalanobis distance can relate the compactness of a set of points from its mean value. The numbers from the data given show that although some landmarks have a high Euclidean distance, they can be relatively centered around the mean value and have a lower Mahalanobis distance.

Q3

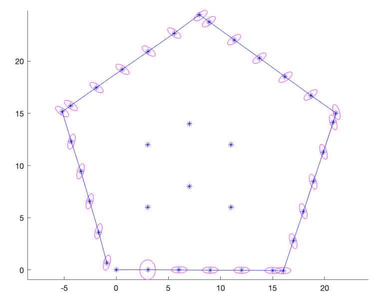
Part a

The initial landmark covariance matrix has zeros in the off-diagonal blocks within the landmark-landmark sub-matrix. The final covariance matrix has nonzero values for these cross covariances because the landmarks are potentially dependent on each other through confounding variables like the robot pose and hence will have a nonzero estimated value. While setting the initial covariance matrix, we assume the sub-matrix with cross-covariance between robot state and landmarks and the cross-covariance between landmarks is zero. In reality, the robot poses and landmarks are related to each other. Also, the landmarks are not independent of each other, but are conditionally independent, given variables like the robot pose and other variables.

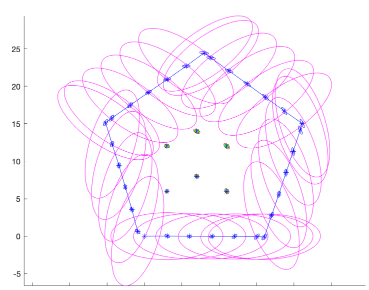
Part b



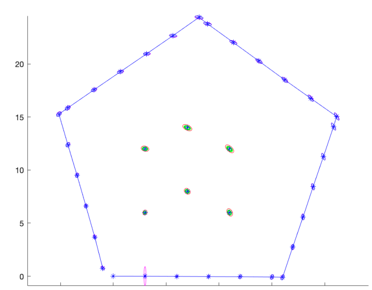
(a) $\sigma_\beta = 0.1$ $\sigma_r = 0.8$



(b) $\sigma_\beta = 0.001$ $\sigma_r = 0.008$



(a) $\sigma_x = 2.5$ $\sigma_y = 1.0$ $\sigma_\alpha = 1.0$



(b) $\sigma_x = 0.025$ $\sigma_y = 0.01$ $\sigma_\alpha = 0.01$

For the case where we change the measurement variable variances:

Having high values of the variances of β and r (top row; (a)), we get an increase in the covariance associated with both the poses and landmarks. We also get a slight change in the trajectory of the robot and the landmark positions. This is expected since the high measurements variances influence the Kalman gain and in turn the estimated values of both mean and variance of the state vector. The variance keeps on increasing with time and the estimated robot trajectory deviates from the actual path. Similarly, having very low values of the variances of β and r(top row; (b)), we see shrinkage of the covariance ellipses of both the poses and the landmarks which shows the uncertainties associated with the states is very low due to low measurement noise.

For the case where we change the measurement variable variances:

Having high values of the variances of σ_x, σ_y and σ_α (bottom row; (a)), we see that the covariance ellipses for the predicted pose of the robot grows large while that of the landmarks stays more or less the same. The control variances lead to large uncertainties in the robot pose. Similarly, having low values of the variances of σ_x, σ_y and σ_α (bottom row; (b)), we see that the predicted uncertainties shrink to low values and we get smaller covariance ellipses. In both the above cases, only prediction step variances get affected as we're changing only the control noise variables.

Part c

For the cases when the no of landmarks is increasing as the robot is moving into new environments, it makes sense to not estimate and update values for the landmarks which have gone out of view. The SLAM front-end can be used for robust landmark matching and data association so we only deal with landmarks in the view. Alternatively, to keep the runtime constant in the no of landmarks, we can design a heuristic which can be a function of the landmark covariance sub-matrix, which keeps the best N landmarks for estimation and update. This method will have a trade-off as choosing high uncertainty landmarks will improve the map by refining the landmark positions, but might degrade the Kalman gain value which in turn degrades the robot localization. Choosing the low uncertainty landmarks will have the opposite effect. Hence, a good heuristic must be designed for ranking of landmarks to be used. Another idea is to exploit the dependencies in the map itself to rank landmarks (eg, some landmarks have spatial dependencies and we can choose a few from multiple landmarks in the same location).