16-833 Homework 2

Changsheng Shen
changshs@andrew.cmu.edu

Mar. 6, 2020

1. Related Math and Equations

(a): From geometric relationship we can get:

$$\mathbf{p}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + d_t \cdot cos\theta_t \\ y_t + d_t \cdot sin\theta_t \\ \theta_t + \alpha_t \end{bmatrix}$$

(b):

$$\mathbf{p}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + d_t \cdot cos\theta_t + e_x \cdot cos\theta_t - e_y \cdot sin\theta_t \\ y_t + d_t \cdot sin\theta_t + e_x \cdot sin\theta_t + e_y \cdot cos\theta_t \\ \theta_t + \alpha_t + e_\alpha \end{bmatrix}$$

Here let's denote:

$\mathbf{p}_{t+1} = f(\mathbf{p}_t, \mathbf{u}_t, \mathbf{w}_t)$, where $\mathbf{u}_t = \begin{bmatrix} d_t \\ \alpha_t \end{bmatrix}$ is the control input, and $\mathbf{w}_t = \begin{bmatrix} e_x \\ e_y \\ e_\alpha \end{bmatrix}$ is the propagation noise.

The covariance matrix of the error in control input (propagation noise): $Q = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\alpha^2 \end{bmatrix}$

To propagate the uncertainty (as covariance matrix) from time t to time $t+1$, let's first compute the Jacobian matrices:

$$F_t = \frac{\partial f}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & -d_t \cdot sin\theta_t \\ 0 & 1 & d_t \cdot cos\theta_t \\ 0 & 0 & 1 \end{bmatrix}$$

$$L_t = \frac{\partial f}{\partial \mathbf{w}} = \begin{bmatrix} cos\theta_t & -sin\theta_t & 0 \\ sin\theta_t & cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then the predicted uncertainty of the robot at time $t+1$ is the Gaussian distribution $\mathcal{N}(0, \Sigma_{t+1})$, where:

$$\Sigma_{t+1} = F_t \Sigma_t F_t^T + L_t Q L_t^T$$

(c):

$$\mathbf{l} = \begin{bmatrix} l_x \\ l_y \end{bmatrix} = \begin{bmatrix} x_t + (r_t + n_r) \cdot cos(\theta_t + \beta_t + n_\beta) \\ y_t + (r_t + n_r) \cdot sin(\theta_t + \beta_t + n_\beta) \end{bmatrix}$$

Or we can represent the estimation of landmark postion as follows:

$$\mathbf{l} = \begin{bmatrix} l_x \\ l_y \end{bmatrix} = \begin{bmatrix} x_t + r_t \cdot cos(\theta_t + \beta_t) \\ y_t + r_t \cdot sin(\theta_t + \beta_t) \end{bmatrix}$$

Denote $\mathbf{l} = g(\mathbf{p}_t, \mathbf{z}_t)$, where $\mathbf{z}_t = \begin{bmatrix} \beta_t \\ r_t \end{bmatrix}$ is the laser sensor measurement.

Denote the measurement covariance as $R = \begin{bmatrix} \sigma_\beta^2 & 0 \\ 0 & \sigma_r^2 \end{bmatrix}$

To project the uncertainty (as covariance matrix) of robot pose and measurement onto the uncertainty of landmark location, let's first compute the Jacobian matrices:

$$G_t = \frac{\partial g}{\partial \mathbf{p}} = \begin{bmatrix} 1 & 0 & -r_t \cdot sin(\theta_t + \beta_t) \\ 0 & 1 & r \cdot cos(\theta_t + \beta_t) \end{bmatrix}$$

$$M_t = \frac{\partial g}{\partial \mathbf{z}} = \begin{bmatrix} -r_t \cdot sin(\theta_t + \beta_t) & cos(\theta_t + \beta_t) \\ r_t \cdot cos(\theta_t + \beta_t) & sin(\theta_t + \beta_t) \end{bmatrix}$$

Then the uncertainty of landmark location can be represented as the covariance matrix Σ_l:

$$\Sigma_l = G_t \Sigma_t G_t^T + M_t R M_t^T$$

(recall that Σ_t is the covariance matrix of robot pose at time t).

(d):

Denote the estimated (predicted) measurement as $\hat{\mathbf{z_t}} = \begin{bmatrix} \hat{\beta}_t \\ \hat{r}_t \end{bmatrix} = h(\mathbf{p}_t, \mathbf{l})$

$$h(\mathbf{p}_t, \mathbf{l}) = \hat{\mathbf{z_t}} = \begin{bmatrix} \hat{\beta}_t \\ \hat{r}_t \end{bmatrix} = \begin{bmatrix} warpToPi(atan2(l_y - y_t, l_x - x_t) - \theta_t) \\ \sqrt{(l_y - y)^2 + (l_x - x)^2} \end{bmatrix}$$

The covariance matrix S_t of the predicted measurement $\hat{\mathbf{z_t}}$ is given by:

$$S_t = H_p \Sigma_t H_p^T + R$$

where R is defined in **1.(c)** and H_p is the Jacobian matrix $H_p = \dfrac{\partial h}{\partial \mathbf{p}}$ (see below in **1.(e)**)

(e):

For simplicity, let's denote $dy_t = l_y - y_t$, $dx_t = l_x - x_t$

$$H_p = \frac{\partial h}{\partial \mathbf{p}} = \begin{bmatrix} \dfrac{dy_t}{dx_t^2 + dy_t^2} & \dfrac{-\,dx_t}{dx_t^2 + dy_t^2} & -1 \\ \dfrac{-\,dx_t}{\sqrt{dx_t^2 + dy_t^2}} & \dfrac{-\,dy_t}{\sqrt{dx_t^2 + dy_t^2}} & 0 \end{bmatrix}$$

(f):

For simplicity, let's denote $dy_t = l_y - y_t$, $dx_t = l_x - x_t$

$$H_l = \frac{\partial h}{\partial \mathbf{l}} = \begin{bmatrix} \dfrac{-\,dy_t}{dx_t^2 + dy_t^2} & \dfrac{dx_t}{dx_t^2 + dy_t^2} \\ \dfrac{dx_t}{\sqrt{dx_t^2 + dy_t^2}} & \dfrac{dy_t}{\sqrt{dx_t^2 + dy_t^2}} \end{bmatrix}$$

We don't need to calculate the measurement Jacobian with respect to other landmarks except for itself based on the assumption that all the landmarks' location are independent from each other, and the measurement for each landmark is i.i.d.

2. Implementation and Evaluation

(a):

There are 6 fixed landmarks.

(b):

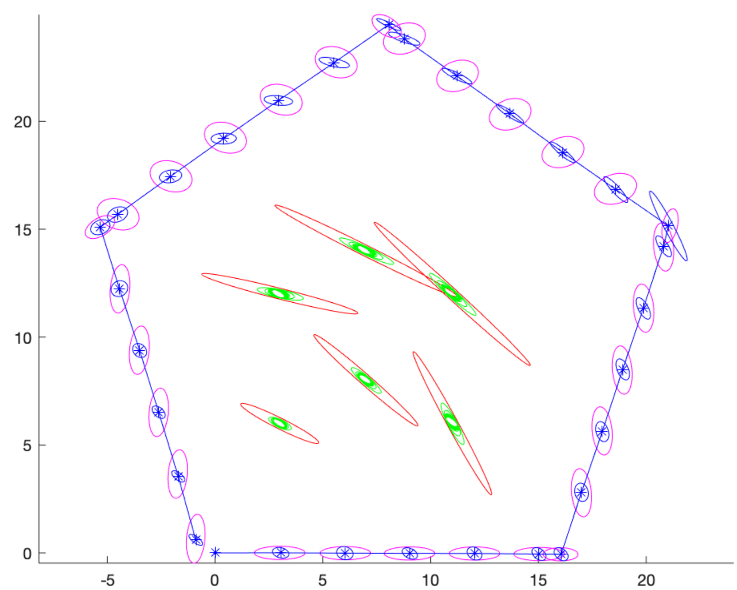


Figure 1: Result with default parameter setting

(c):

From the figure, we can observe that:

In terms of robot trajectory, the blue ellipses are (almost) always smaller than the corresponding magenta ellipses, in both dimensions.

Similarly, in terms of the "map" (landmark positions), the green ellipses are also smaller the corresponding red ellipses.

From that we can say that the iterative covariance update process of EKF-SLAM improves the estimation of both robot trajectory and map (landmark positions), by making the uncertainty smaller and smaller.

(d):

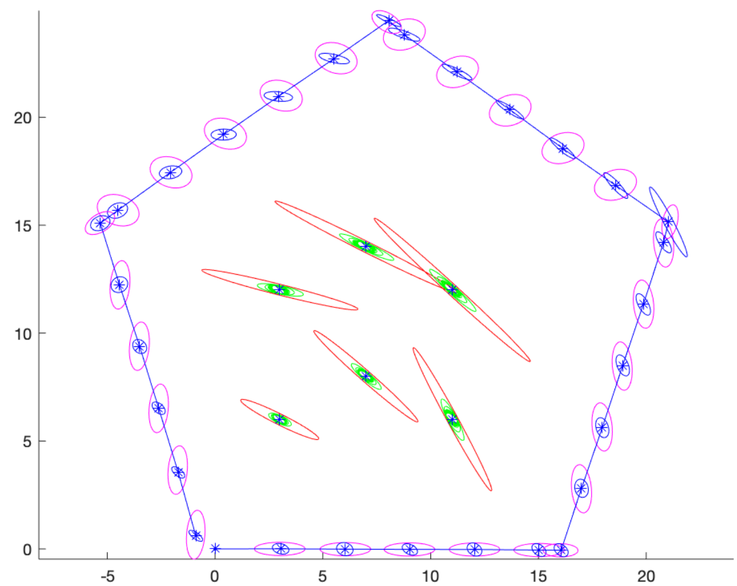


Figure 2: Result with landmark ground truth

From the figure above we can see that every ground truth landmark position is inside the smallest corresponding ellipse. This means the means of the ellipses are roughly correct (close to the ground truth), and the covariances correctly represent the uncertainty.

Landmark	Ground Truth	Euclidean	Mahalanobis
1	(3, 6)	0.046661	0.005009
2	(3, 12)	0.115305	0.013181
3	(7, 8)	0.090453	0.010289
4	(7, 14)	0.152463	0.019646
5	(11, 6)	0.115155	0.012092
6	(11, 12)	0.152411	0.019822

The euclidean distances represents the absolute amount of difference between the mean of the estimated position of each landmark, and the corresponding ground truth.

The Mahalanobis distances represents the relative differences, in terms of how many standard deviations away is the ground truth from the mean of the estimated distribution.

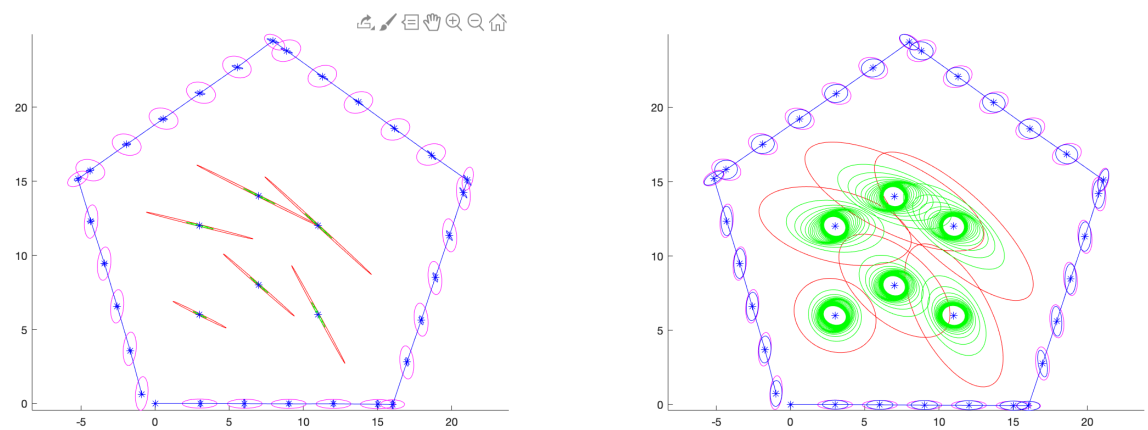
3. Discussion

(a):

The zero terms in the initial landmark covariance matrix indicates that the position of each landmark is independent with other landmarks. However, as EKF iteratively updates the state vector and the corresponding covariance matrix, it projects all the related uncertainties onto each single component of the covariance matrix (with the calculated Kalman gain), including the process noise and the measurement noise. This makes every landmark position correlated to other landmarks positions, since at each time step we are observing all the landmarks from each uncertain position, and update our covariance matrix iteratively for every single landmark observation (with noise added).

When we initialize the full covariance matrix P, we made the assumption that the position of each landmark is independent with other landmarks. This is not necessarily correct because we use the initial measurement to calculate the initial covariance, where all the measurements are obtained from the same starting position. Therefore, there indeed exist some correlation between all the landmark positions (imagine the factor graph for this measurement, all landmarks are connected to the same pose).

(b):



(a) 0.1x std (Measurement noise)

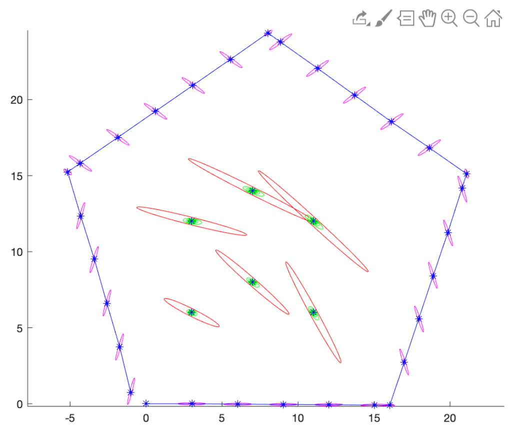
(b) 10x std (Measurement noise)

Figure 3: Default σ_x, σ_y and σ_α

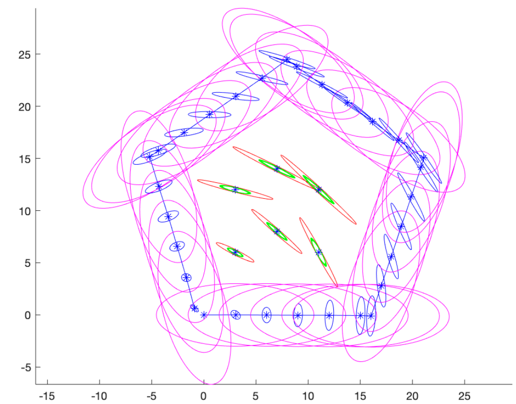
As we change the standard deviation of measurement noise, we observe that the uncertainty ellipses for the landmarks changes significantly, and the uncertainty on robot poses also changes a bit correspondingly.

When the measurement noise is high, it means that we do not trust our measurement very much (or say, the uncertainty in each measurement is high), which directly results in a high uncertainty in "measured" landmark positions. Since we updates our whole state vector based on all information we have, the increased noise will also result in increased uncertainties in robot poses, but not that significant.

Similar logic applies when we fix the measurement noise and change the process noise (see figure below).



(a) 0.1x std (Process noise)



(b) 10x std (Process noise)

Figure 4: Default σ_β and σ_r

(c): Potential solutions to this could be:

1. Maintain a fix-sized sliding time window, and only consider and update the landmarks within the time window.

2. Maintain a fix-sized queue to limit the maximum number of landmarks to be used. When the queue is full, newer landmarks enters and oldest landmarks will be throw away.

3. If landmarks are dense, we may extract keyframes and use keyframes only.

4. We may only use landmarks with low uncertainty, and abandon landmarks with high uncertainty.