Discussed the assignment with Akshay Bhagat , abhagat1@andrew.cmu.edu and Sambuddha Sarkar , sambudds@andrew.cmu.edu

SLAM

ab1@andrew.cmu.edu - Amit Bansal

February 23 2017

1 1.(a) Pose Estimate

$$M = \begin{bmatrix} x_t + [d_t * cos(\theta)] \\ y_t + [d_t * sin(\theta)] \\ [\theta_t + \alpha] \end{bmatrix}$$

$$I = \begin{bmatrix} [d_t * cos(\theta)] \\ [d_t * sin(\theta)] \\ [\alpha] \end{bmatrix}$$

$p_{t+1} = M = p_t + I$

2 1.(b) Predicted Uncertainity of Robot

Since, the uncertainity is in robot matrix, we get the jacobian to convert it in World frame. The jacobian is given by B

Also, we need to propagate the error from t-1 so we have the jacobian G for that.

To get the Jacobian B, we differentiate the error in robo's frame with respect to error in x , error in y and error in theta. Hence we get the B matrix

To get G, we differentiate the pose covariance with respect to x,y and theta.

$$B = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$G = \begin{bmatrix} 1 & 0 & -d_t * sin(\theta) \\ 0 & 1 & d_t * cos(\theta) \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = \begin{bmatrix} {\sigma_x}^2 & 0 & 0 \\ 0 & {\sigma_y}^2 & 0 \\ 0 & 0 & {\sigma_\theta}^2 \end{bmatrix}$$

R = B*S*B^T

predicted($\sum_{t+1}$) = $G * \sum_t * G^T + B * S * B^T$

3 1.(c) Estimated position of L

To get the estimated position of lx and ly , we take the current position of the robot and add the measurement and bearing to it. Here theta is the robot's angle ,
beta is the bearing measurement and tau is the range.

$$L = \begin{bmatrix} x_t + ((\tau + error_\tau) * cos(\theta + \beta + error_\beta)) \\ y_t + ((\tau + error_\tau) * sin(\theta + \beta + error_\beta)) \end{bmatrix}$$

$$L = \begin{bmatrix} l_x \\ l_y \end{bmatrix}$$

4 1.(d) Estimated Range and Bearing

From the state vector we get the following. $\mathrm{l}_x, l_y, x_t, y_t$

$\mathrm{xdiff} = \mathrm{l}_x - x_t$
$\mathrm{ydiff} = \mathrm{l}_y - yt$
$\tau = (xdiff^2 + ydiff^2)^{0.5}$
$\beta = wrapTo2Pi(atan2(ydiff/xdiff) - \theta_t))$

5 1.(e) $\mathbf{H}_p$

This is the jacobian that we get on differentiating beta and tau with respect to x, y and θ

$\beta = wrapTo2Pi(atan2(ydiff/xdiff) - \theta_t))$

$\tau = (xdiff^2 + ydiff^2)^{0.5}$

$\mathrm{xdiff} = \mathrm{l}_x - x_t$
$\mathrm{ydiff} = \mathrm{l}_y - yt$
$\tau = (xdiff^2 + ydiff^2)^{0.5}$
$\beta = wrapTo2Pi(atan2(ydiff/xdiff) - \theta_t))$

$$\frac{\partial \beta, \tau}{\partial x, y, \theta} =$$

$$\begin{bmatrix} \frac{\partial \beta}{\partial x_t} & \frac{\partial \beta}{\partial y_t} & \frac{\partial \beta}{\partial \theta_t} \\ \frac{\partial \tau}{\partial x_t} & \frac{\partial \tau}{\partial y_t} & \frac{\partial \tau}{\partial \theta_t} \end{bmatrix}$$

$$H_P =$$

$$\begin{bmatrix} (l_y - y_t)/\tau^2 & (x_t - l_x)/\tau^2 & -1 \\ (x_t - l_x)/\tau & (y_t - ly)/\tau & 0 \end{bmatrix}$$

6 1.(f) $\mathbf{H}_l$

This is the jacobian that we get on differentiating beta and tau with respect to $l_x and ly$

$\beta = wrapTo2Pi(atan2(ydiff/xdiff) - \theta_t))]$

$\tau = (xdiff^2 + ydiff^2)^{0.5}$

$$\frac{\partial \beta, \tau}{\partial x, y} =$$

$$\begin{bmatrix} \frac{\partial \beta}{\partial l_x} & \frac{\partial \beta}{\partial l_y} \\ \frac{\partial \tau}{\partial l_x} & \frac{\partial \tau}{\partial l_y} \end{bmatrix}$$

$$H_l = \begin{bmatrix} (y_t - l_y)/\tau^2 & (l_x - x_t)/\tau^2 \\ (l_x - x_t)/\tau & (ly - y_t)/\tau \end{bmatrix}$$

We dont need to calculate the Jacobian Hl with respect to other landmarks apart from itself because the postion of landmarks are independent of each other. Since the position of one landmark does not effect the postion of the other landmark , they are independent to each other. Since they are independent to each, in the sense that they dont affect each other ,the variance of one with respect to other shall be zero at any instance. Hence, we dont need to calculate the Jacobian.

7 2(a)

There are 6 landmarks being observed by the robot

8 2(b)

The figure is listed below.

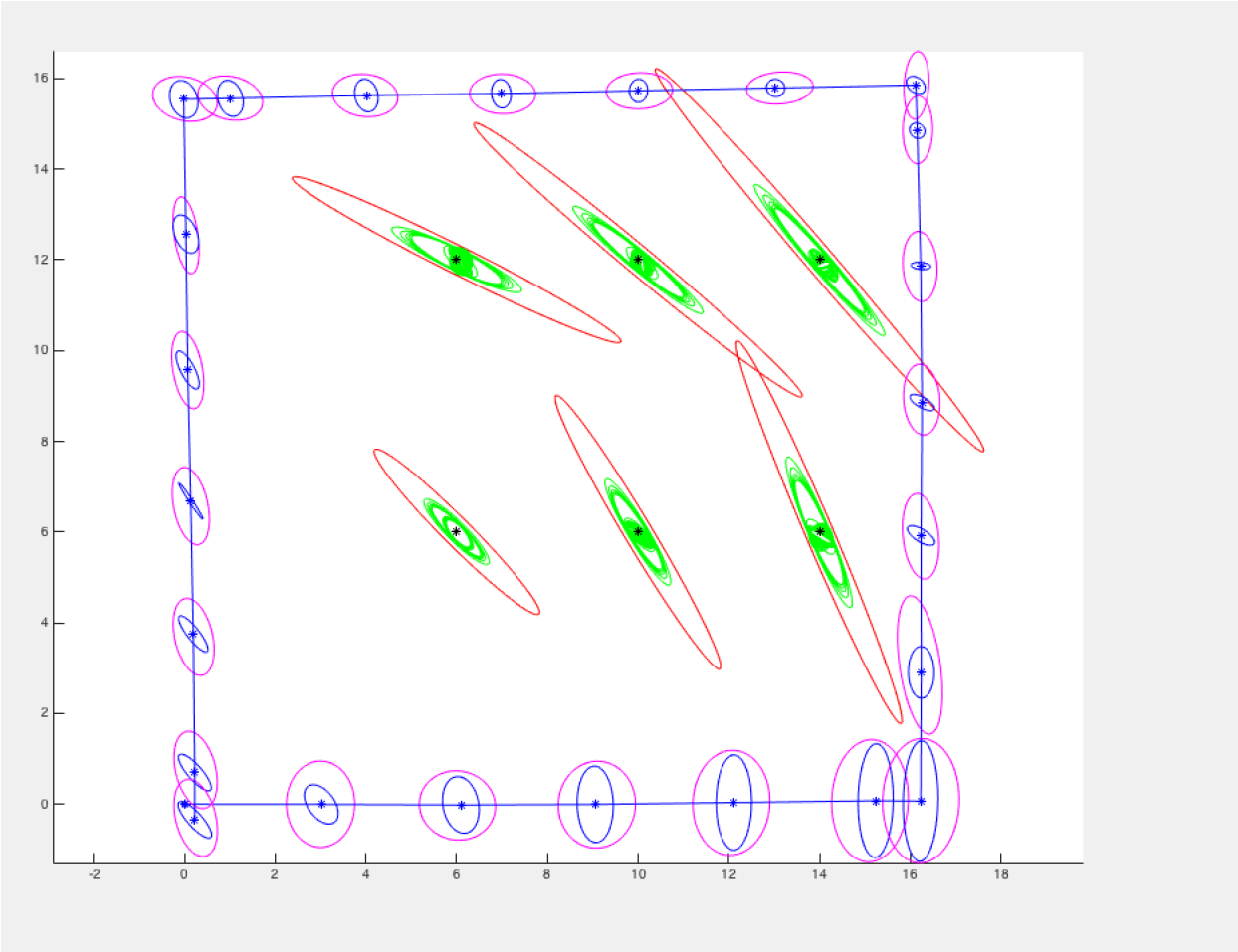

Figure 1: Robot trajectory

9 2(c)

As the Robot moves its uncertainty increases , however when it finds a landmark , it makes a prediction on the estimated position of the landmark and uses the observations to decrease its covariance.
Hence , the covariance of landmark and the robot's pose are both dependent on each other. The predicted position of the landmark is subtracted from the observed position of the landmark to find out how truthful is the controls of the robot.
Also the magneta colored curve gives us the robot pose uncertainty and the blue is the updated robot pose uncertainty after it has observed the landmarks.

Now if you see in the figure(Robot trajectory), the ellipse corresponding to the landmark decreases in size with time ,as the robot progresses on its trajectory.
The initial covariance given by red color is huge , however as the robot moves, it becomes sure about the landmark position and thus the green trajectories decrease with time. Thus, the predicted covariance, decrease with the observations made.

The robot pose covariance decreases as the robot observes the landmark . Whereas the landmark covariance decreases as the robot matches the expected and observed landmark positions in the state .

10 2(d)

Yes, all the ground truth positions of the landmarks are within the covariance ellipses. They are represented by black dot.

This tell us that the mean and the variance around it always included the actual ground truth position of landmark and it never actually missed the landmark completely.
From the statistical point of view it means, the choice of variances of the beta and tau are greater than or equal to the actual variance in the landmark measurement.

The mahalanobis and eucleadian distance is given below : Mahalanobis distance =
0.2255
0.5631
0.4491
0.6455
0.6475
0.6855

Eucleadian distance =

0.0494455319832707
0.103205713684402
0.0848708949389333
0.123092685855020
0.123499370068053
0.152960499067411

The conclusion that can be drawn from the data is that Mahalanobis distance gives us the distance weighted by the covariance. So if one of the axis of the covariance ellipse is more in one direction, the weight of the error would be less. That's why the inverse of the covariance is multiplied. Mahalanobis

distance is more sound analytically.

Hence, there is striking difference between Eucleadian and Mahalanobis distance , as eucleadian distance has no weight and takes the sum of the square of the difference of the coordinates.

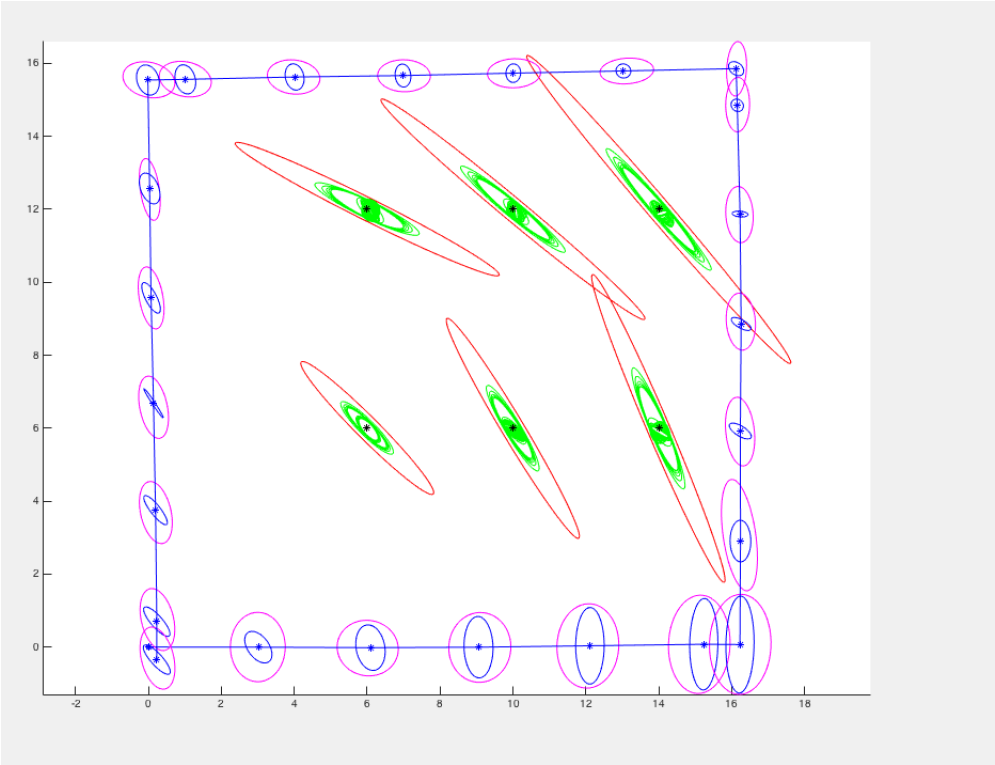

Figure 2: Robot trajectory with Ground Truth Landmark represented with black dot

11 3(a)

The terms non zero in the begining are the covariance of x,y , theta and landmarks. Some terms initially zero in the covariance matrix become non-zero because as the robot progresses and observes the landmarks, the measurements made on the landmarks are dependent on the robot position. Also, the robot position are corrected based on the measurements.
The landmark covariance is affected by the robots position. The more accurate the robots position the less error in observed landmarks.

and if the error in landmarks is less , the robot's position would be updated more accurately.

The co-variances of the robot position is dependent on the co-variance of the landmark and vice-versa i.e. the landmark covariance is dependent on the robot covariance.

Hence , the update in robot's position is dependent on the landmarks accurate predication, and the accuracy of landmarks is dependent on the robot's correct positioning,
Therefor the terms initially zero become non-zero

12 3(b)

The following changes were observed on keeping the x, y and theta same and varying the beta and tau.
These are the inputs :

1. Figure 3 represents the robot trajectory

$\text{sig}_x = 0.25$;
$sig_y = 0.1$;
$sig_alpha = 0.1$;
$sig_beta = 0.2$;
$sig_r = 1.4$;

On increasing the beta and sigma the landmark ground truth stays within the green eclipses.

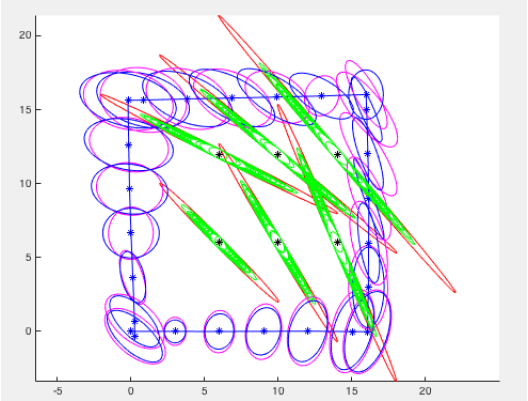

Figure 3: Robot trajectory

2. This corresponds to the robot trajectory in figure:4

$\text{sig}_x = 0.25$;

$sig_y = 0.1$;
$sig_alpha = 0.1$;
$sig_beta = 0.1$;
$sig_r = 0.5$;

Additionally, it can be seen that if you increase the beta and r to make them comparable to the uncertainty in x, y and theta, the uncertainty in robot's pose aligns itself to the uncertainty in landmark pose.

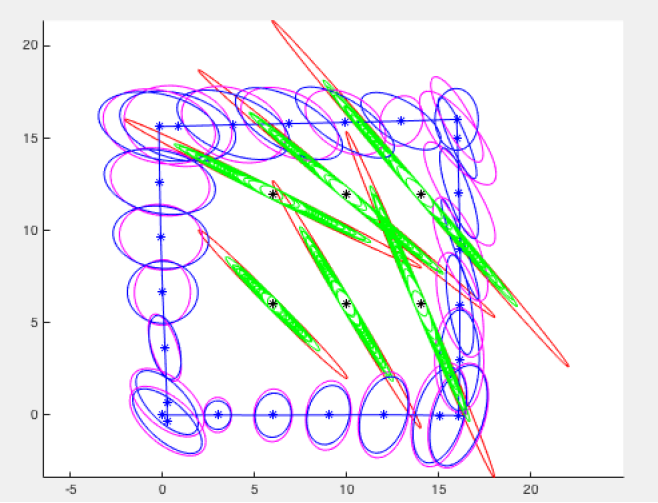

Figure 4: Robot trajectory

3. This refers to the robot trajectory in Figure :5

As we decrease the beta and r with respect to x,y and theta, the ground truth of the landmark starts to go outside the smallest green ellipse. One reason is because the real variance of the landmark is greater than what is assumed.

The following changes were observed on keeping the beta, tau same and varying the x, y and theta

4. This refers to the robot trajectory in Figure :6

$sig_x = 0.65$;
$sig_y = 0.5$;
$sig_alpha = 0.5$;
$sig_beta = 0.01$;
$sig_r = 0.08$;

As the robot variance increases, The landmark variance increases as well

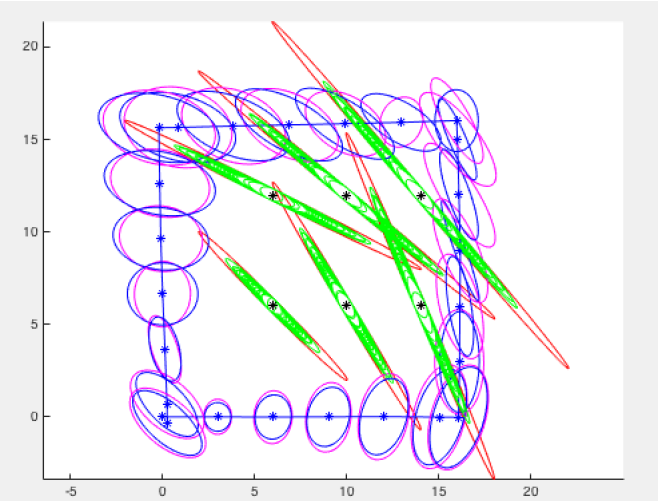

Figure 5: Robot trajectory

compared to previous cases

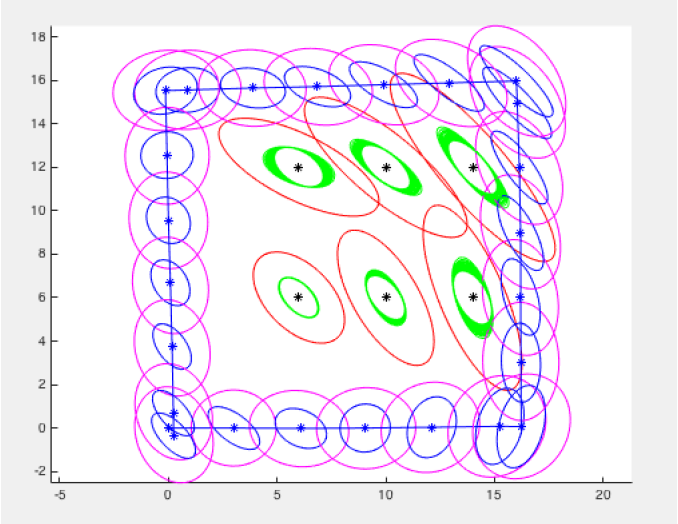

Figure 6: Robot trajectory

5. This refers to the robot trajectory in Figure :7

$\text{sig}_x = 0.05;$
$sig_y = 0.05;$

$sig_alpha = 0.05$;
$sig_beta = 0.01$;
$sig_r = 0.08$;

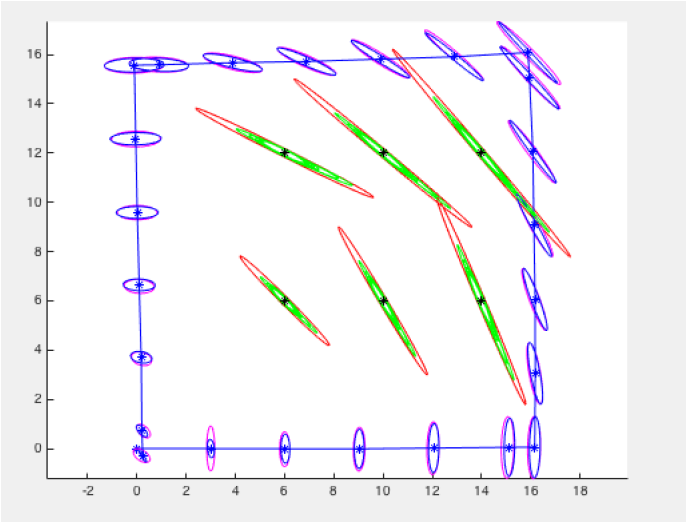

Figure 7: Robot trajectory

13 3(c)

The changes that can be made to the EKF slam to achieve constant computation time in each cycle are :

1) Batch : Define a constant number of landmarks to be considered in every iteration. So, drop the landmarks if they grow more than the specified number

2) Sliding window filter slam - The robot poses and the landmarks are slowly marginalized out over time, such that the state vector ceases to grow and the Slam becomes constant time.

3) Sub-mapping Methods/Map joining Slam : A map can be broken down into regions with local coordinate systems and arranged in hierarchical manner. Updates can be made on periodic time intervals. This reduces the computation time

4) Divide And Conquer Algorithm : Map joining SLAM and DC SLAM carry out the same number of map joining operations. The fundamental difference is that in DC SLAM, the size of the maps involved in map joining increases

at a slower rate than in Map Joining SLAM.

4) Represent Map posterior by relative information between features in the map, and between the map and the robot's pose. This would make the computation time smaller.