Task 1.

A) $\beta = 20^\circ$ – standard deviation.

$\Rightarrow Q = \beta^2 = \underset{\text{(for degrees)}}{400}$ – the covariance of the noise added to the observation function.

The noise added to the transition function is in action space.

$$\varepsilon_t = \begin{bmatrix} \varepsilon\delta_{rot_1} \\ \varepsilon\delta_{trans} \\ \varepsilon\delta_{rot_2} \end{bmatrix} = N\left(0, \underset{?}{R_t}\right) = N\left(0, \underbrace{\begin{bmatrix} \alpha_1\delta_{rot_1}^2 + \alpha_2\delta_{trans}^2 & 0 & 0 \\ 0 & \alpha_3\delta_{trans}^2 + \alpha_4(\delta_{rot_1}^2 + \delta_{rot_2}^2) & 0 \\ 0 & 0 & \alpha_1\delta_{rot_2}^2 + \alpha_2\delta_{trans}^2 \end{bmatrix}}_{R_t}\right)$$

Initial robot command:

$$u = \begin{bmatrix} \delta_{rot_1} \\ \delta_{trans} \\ \delta_{rot_2} \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 0 \end{bmatrix}$$

$$\Rightarrow R_t = \begin{bmatrix} 100\alpha_2 & 0 & 0 \\ 0 & 100\alpha_3 & 0 \\ 0 & 0 & 100\alpha_2 \end{bmatrix}$$

Alphas:

$\alpha_1 = 0{,}05^2$

$\alpha_2 = 0{,}001^2$

$\alpha_3 = 0{,}05^2$

$\alpha_4 = 0{,}01^2$

$$\Rightarrow R_t = \begin{bmatrix} 0{,}0001 & 0 & 0 \\ 0 & 0{,}25 & 0 \\ 0 & 0 & 0{,}0001 \end{bmatrix}$$

Jacobians for odometry model:

$$G_t = \frac{\partial g(x_{t-1}, u_t)}{\partial x_{t-1}}\Bigg|_{\mu_{t-1}} = \begin{bmatrix} 1 & 0 & -\delta_{trans}\sin(\theta+\delta_{rot1}) \\ 0 & 1 & \delta_{trans}\cos(\theta+\delta_{rot1}) \\ 0 & 0 & 1 \end{bmatrix}$$

Initial mean state:

$$\mu_1 = \begin{bmatrix} 180 \\ 50 \\ 0 \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \Rightarrow G_{t=1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 10 \\ 0 & 0 & 1 \end{bmatrix}$$

Remember, that in first robot command we $\begin{bmatrix} \delta_{rot1} \\ \delta_{trans} \\ \delta_{rot2} \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 0 \end{bmatrix}$

$$V_t = \frac{\partial g(x_{t-1}, u_t)}{\partial u_t}\Bigg|_{\bar{\mu}_t} = \begin{bmatrix} -\delta_{trans}\cdot\sin(\theta+\delta_{rot1}) & \cos(\theta+\delta_{rot1}) & 0 \\ \delta_{trans}\cdot\cos(\theta+\delta_{rot1}) & \sin(\theta+\delta_{rot1}) & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$V_{t=1} = \begin{bmatrix} 0 & 1 & 0 \\ 10 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Sensor model:

$$\begin{bmatrix} r_t^i \\ \phi_t^i \\ s_t^i \end{bmatrix} = \underbrace{\begin{bmatrix} \overbrace{\sqrt{(m_{i,x}-x)^2+(m_{i,y}-y)^2}}^{=\sqrt{q}} \\ \operatorname{atan2}(m_{i,y}-y,\ m_{i,x}-x)-\theta \\ s_i \end{bmatrix}}_{h(x_t,\, m_i)} + \underbrace{\begin{bmatrix} \delta_{r_r}^2 \\ \delta_{r_\phi}^2 \\ \delta_{r_s}^2 \end{bmatrix}}_{\delta_t}$$

i - a number of a landmark.

$$H_t^i = \frac{\partial h(x_t)}{\partial x_t}\Bigg|_{\bar{\mu}_t} = \begin{bmatrix} -\frac{(m_{i,x}-\bar{\mu}_{t,x})}{\sqrt{q}} & -\frac{(m_{i,y}-\bar{\mu}_{t,y})}{\sqrt{q}} & 0 \\ \frac{m_{i,y}-\bar{\mu}_{t,y}}{q} & -\frac{(m_{i,x}-\bar{\mu}_{t,x})}{q} & -1 \end{bmatrix} =$$

$$= \begin{bmatrix} -\frac{m_{i,x}-180}{\sqrt{(m_{i,x}-180)^2+(m_{i,y}-50)^2}} & -\frac{m_{i,y}-50}{\sqrt{(m_{i,x}-180)^2+(m_{i,y}-50)^2}} & 0 \\ \frac{m_{i,y}-50}{(m_{i,x}-180)^2+(m_{i,y}-50)^2} & -\frac{m_{i,x}-180}{(m_{i,x}-180)^2+(m_{i,y}-50)^2} & -1 \end{bmatrix}$$

Task 1C. Plots of pose error versus time
EKF

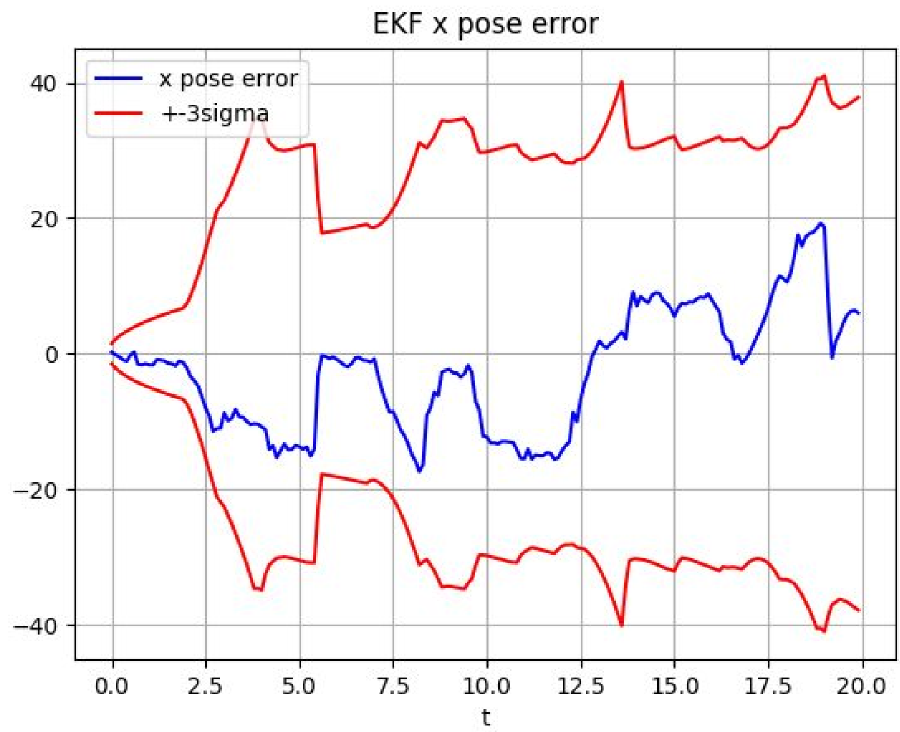

Figure-1. EKF x pose error

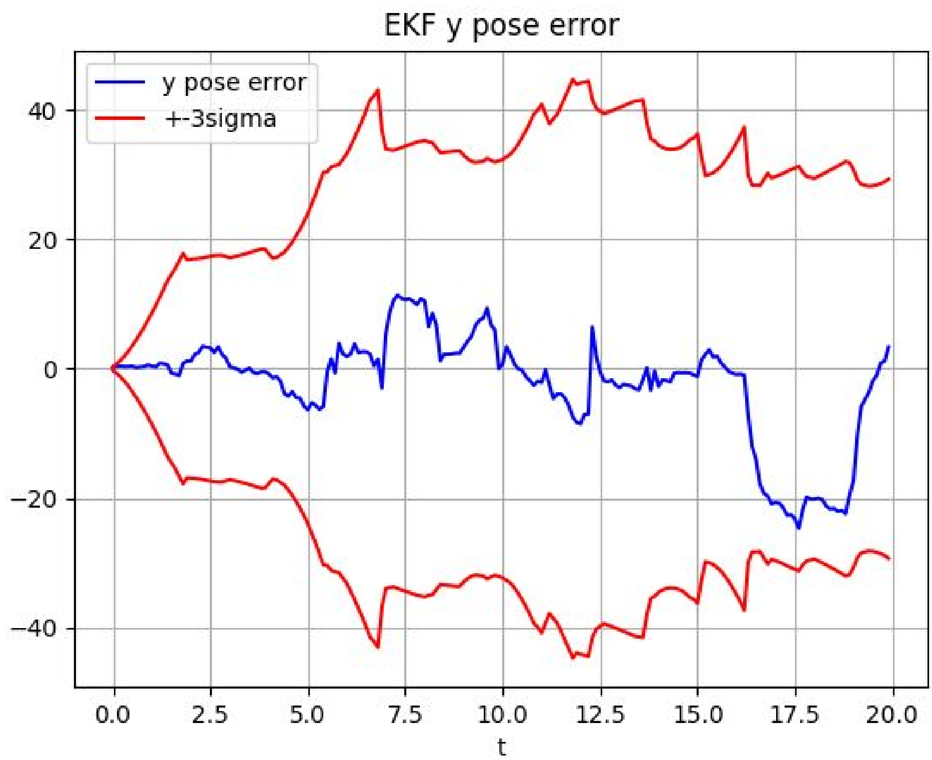

Figure-2. EKF y pose error

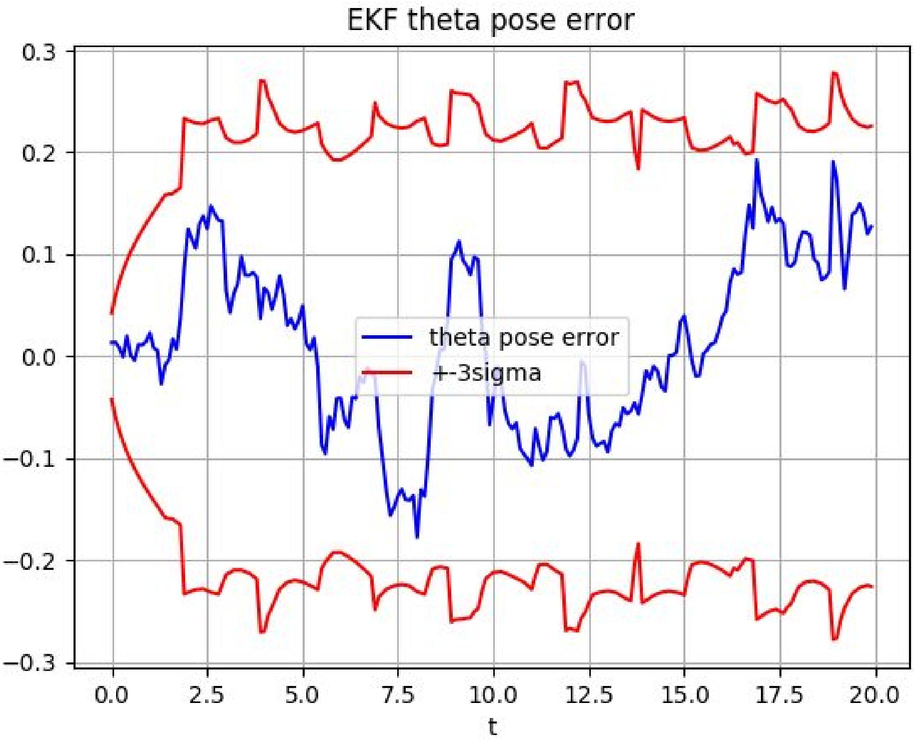

Figure-3. EKF theta pose error

PF

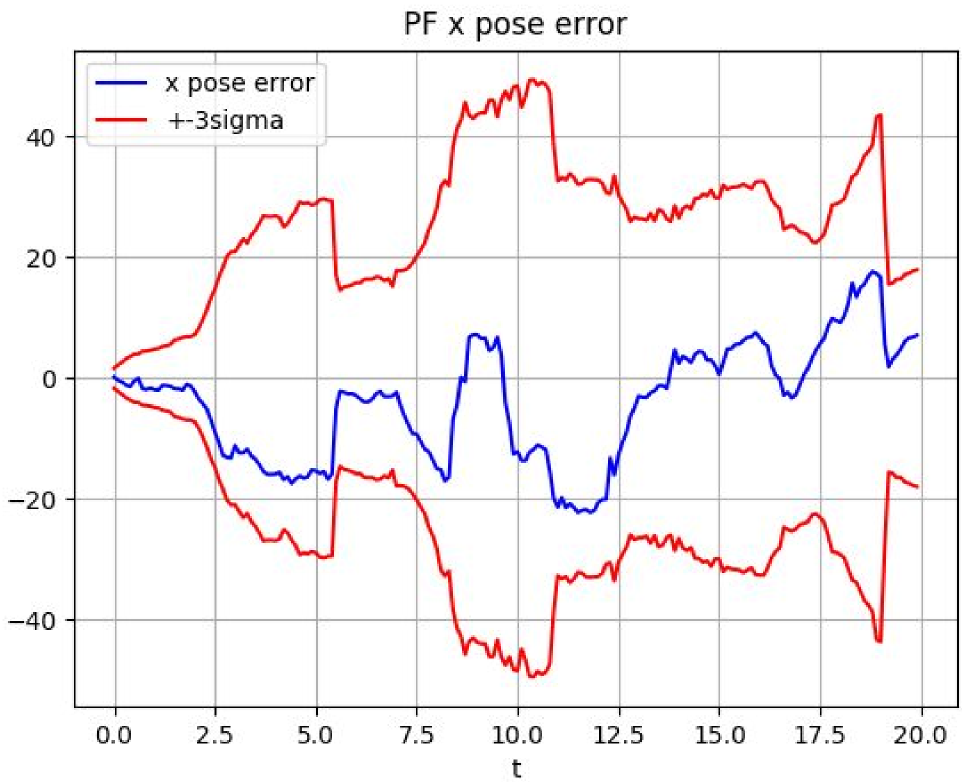

Figure-4. PF x pose error

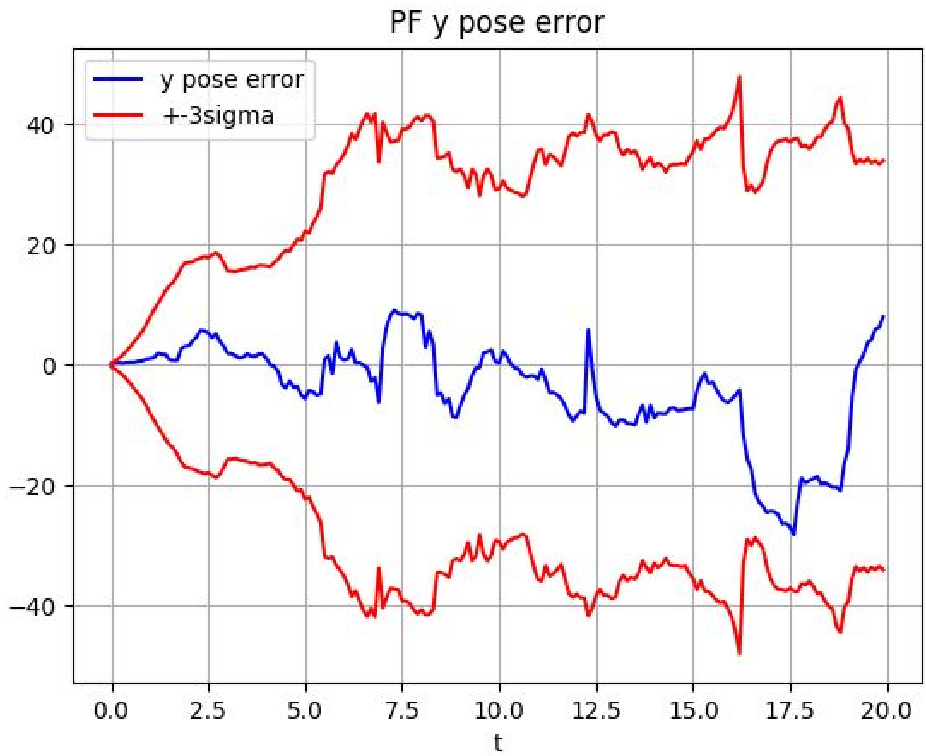

Figure-5. PF y pose error

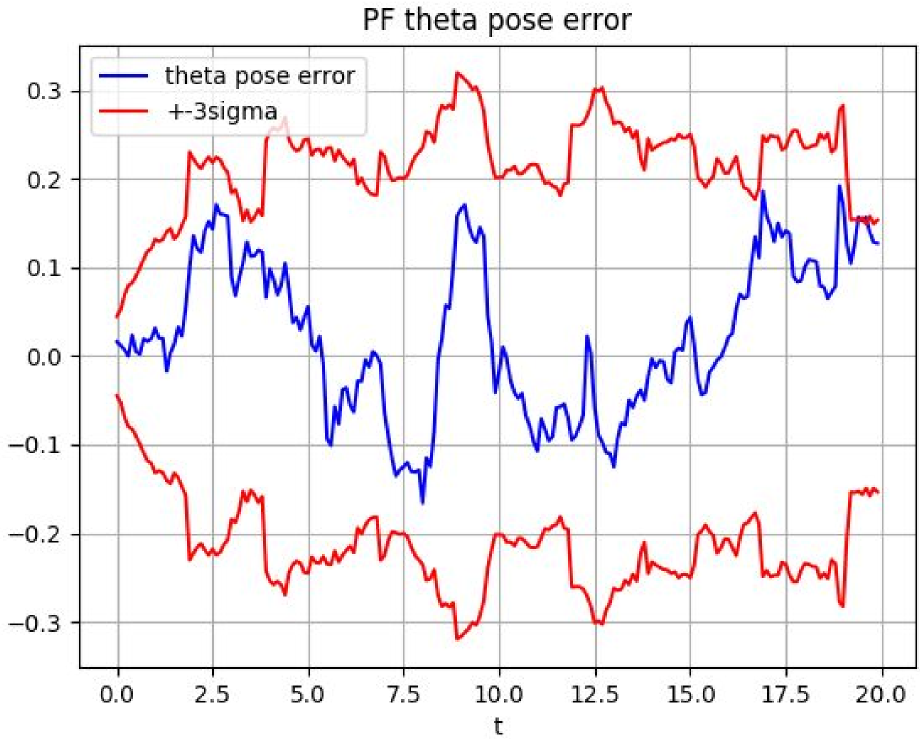

Figure-6. PF theta pose error

All state errors lie within bounds of 3 sigma with some insignificant exceptions.

Task 1D.

1) **Sensor noise value goes toward zero in EKF filter**

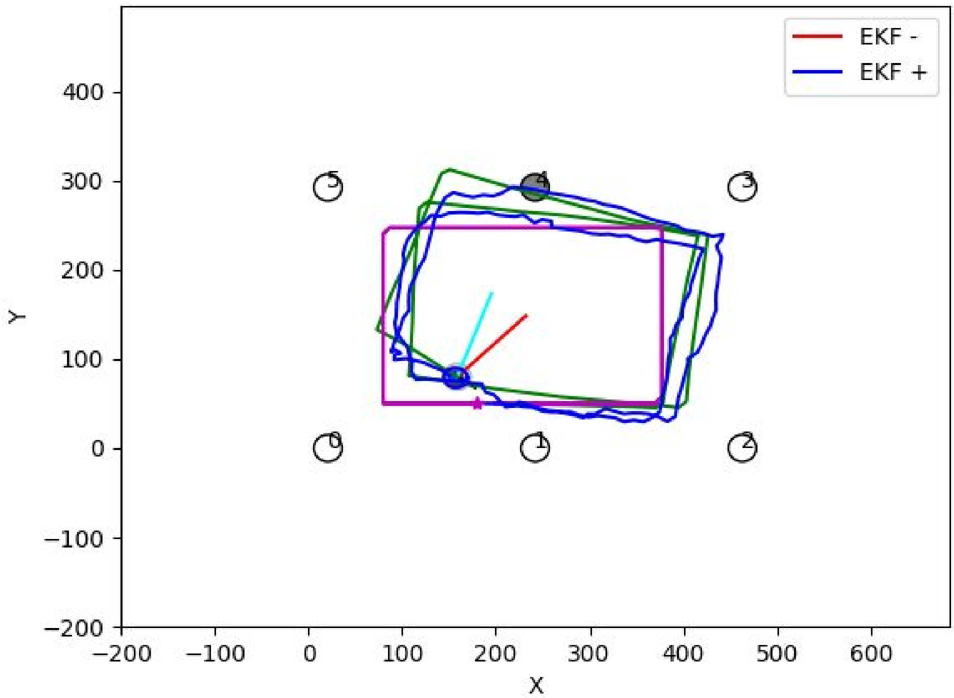

Figure-1. Simulator plot. Q = Q/16

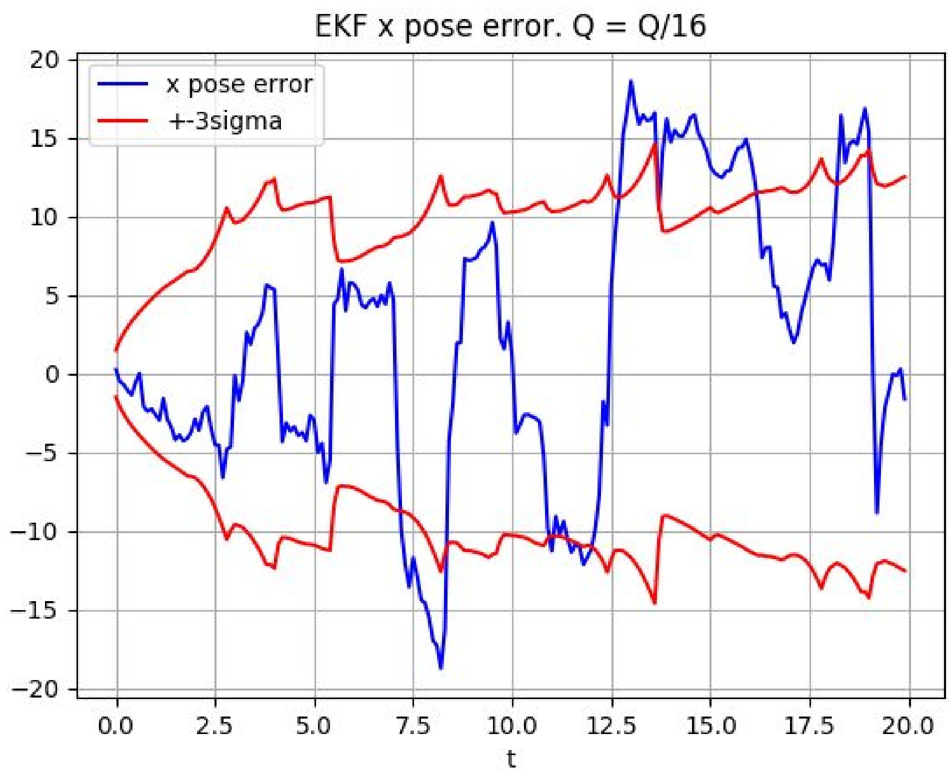

Figure-2. EKF x pose error. Q = Q/16

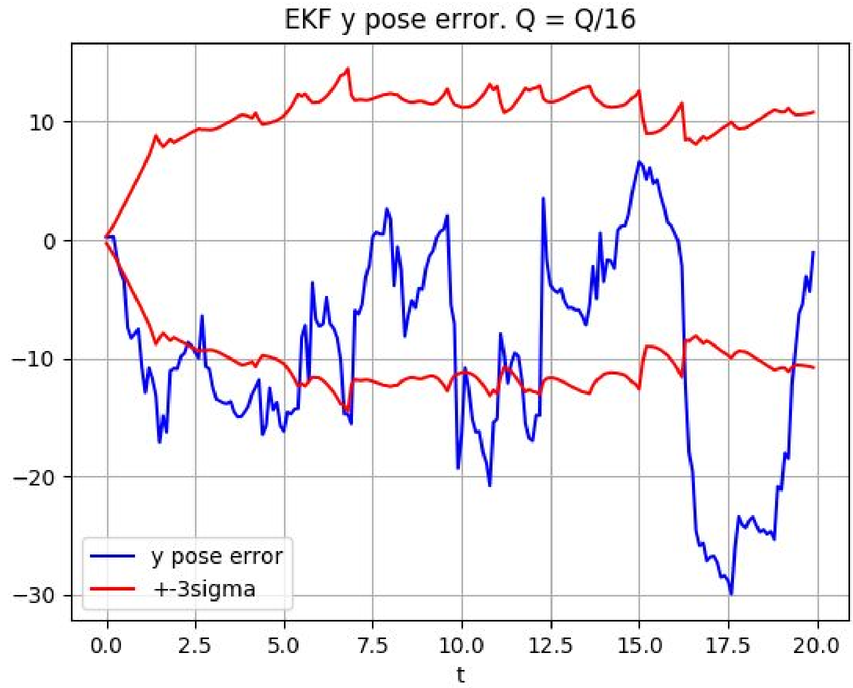

Figure-3. EKF y pose error. Q = Q/16

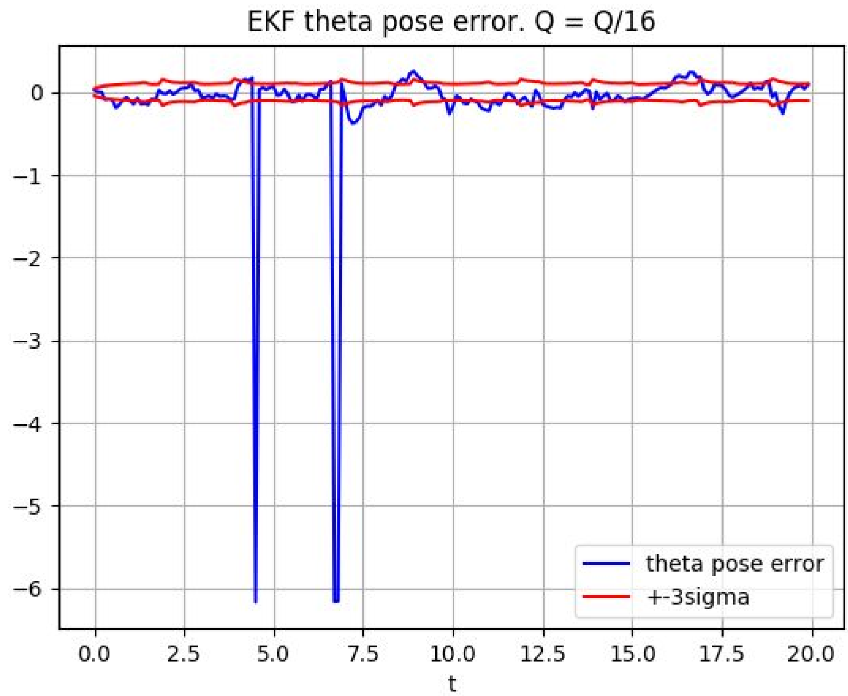

Figure-4. EKF theta pose error. Q = Q/16

When the sensor noise value is reduced (the covariance matrix is reduced by 16), the sensor is assumed more precise than before and 2D Gaussian becomes smaller. However, the robot's ground-truth position was under higher sensor noise than we use in our path reconstruction. As a result, the state error is sometimes much higher than uncertainty bounds of 3-sigma.

2) Motion noise value goes toward zero in EKF filter

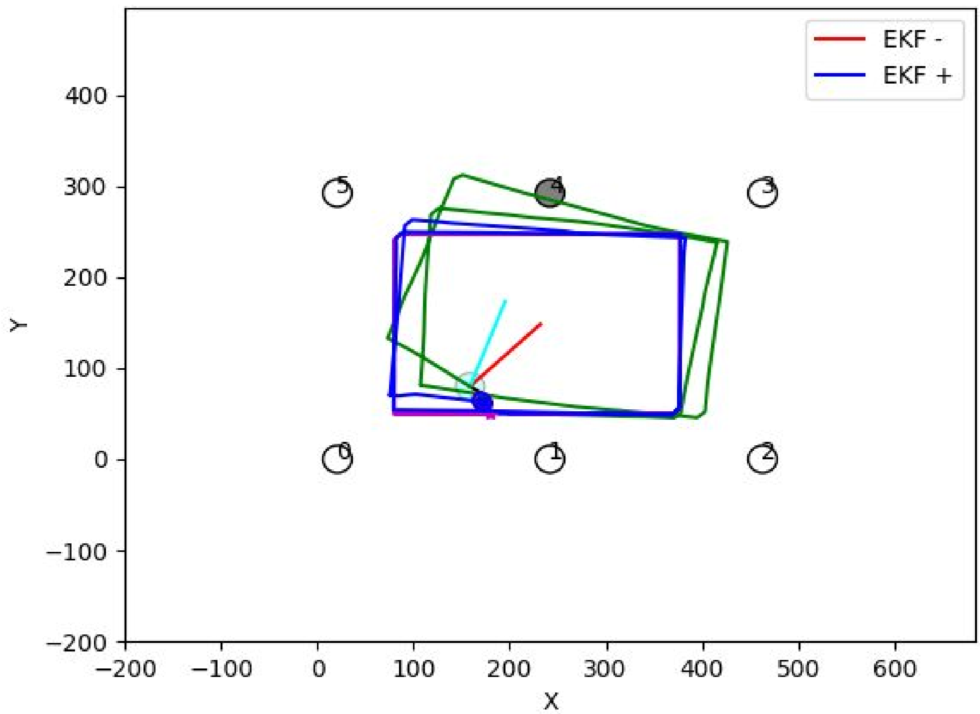

Figure-5. Simulator plot. R = R/100

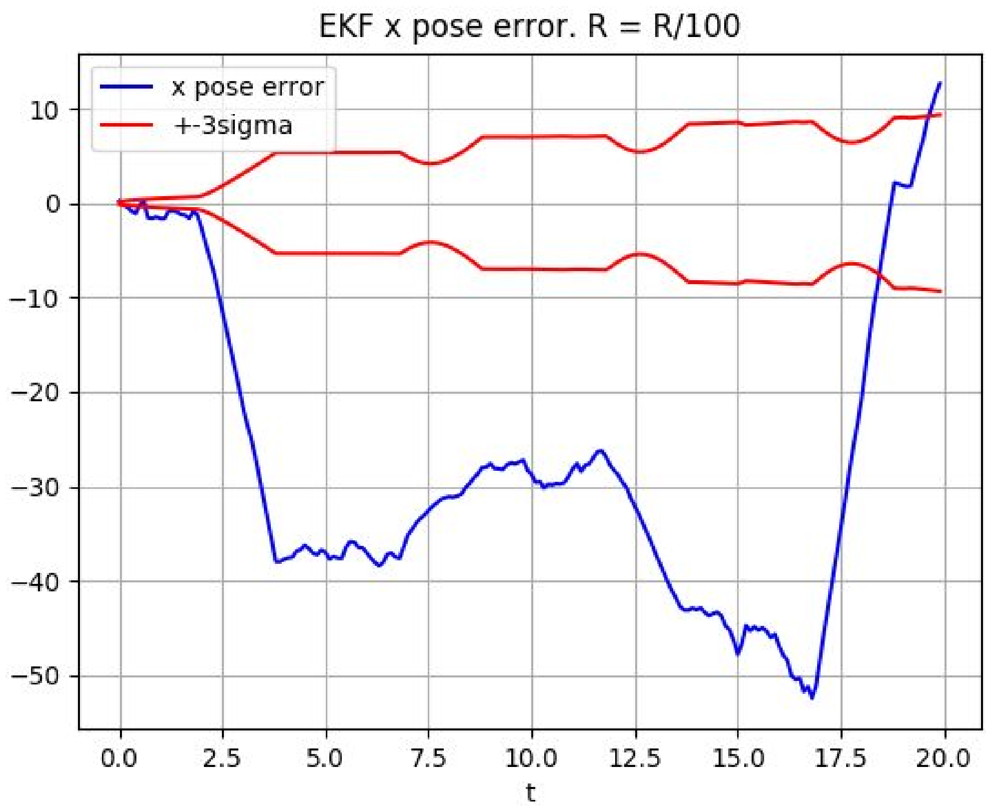

Figure-6. EKF x pose error. R = R/100

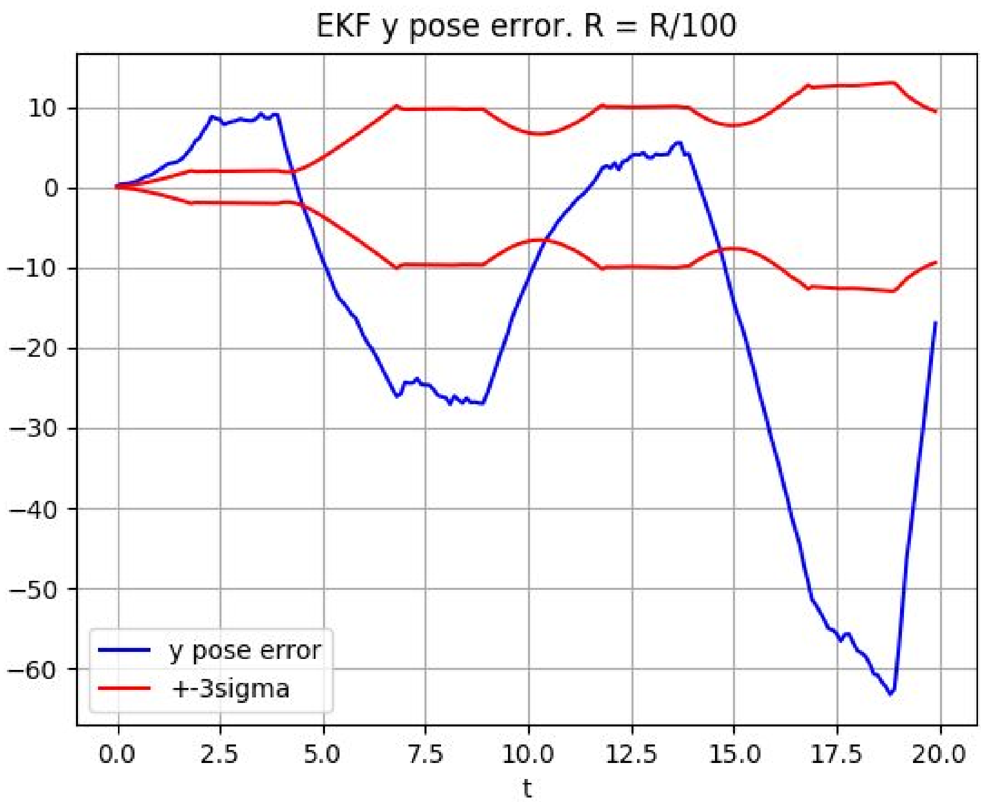

Figure-7. EKF y pose error. R = R/100

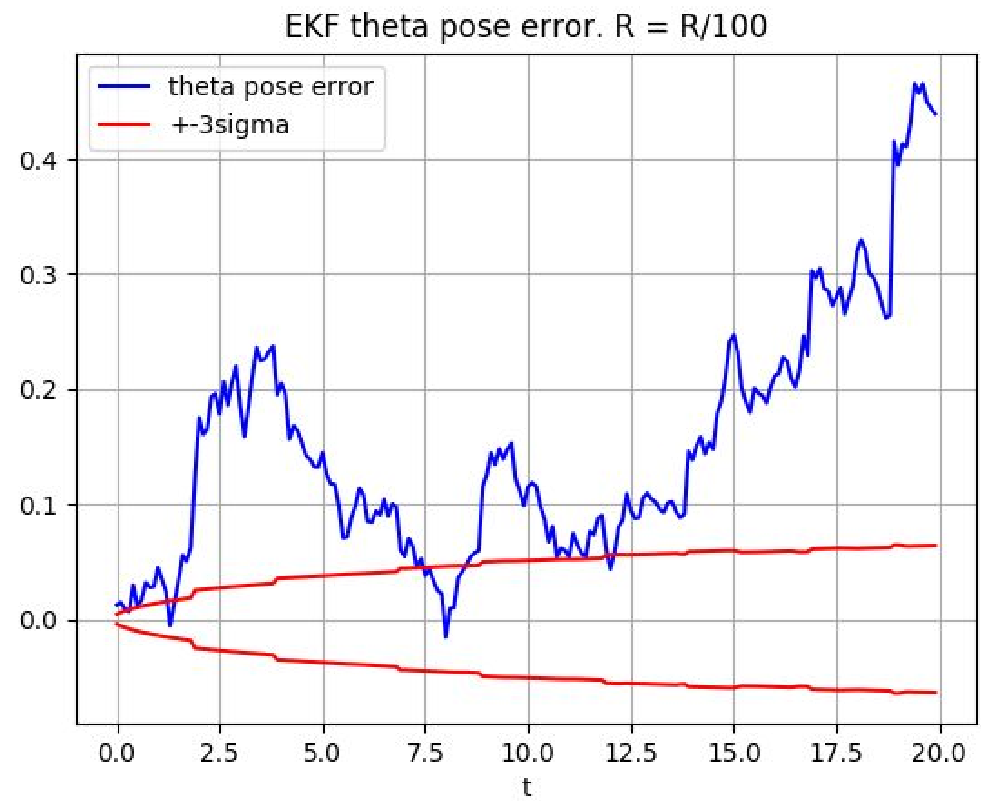

Figure-8. EKF theta pose error. R = R/100

When the motion noise is reduced (the covariance matrix is reduced by 100), the reconstructed robot's path becomes much closer to imagined one, which the robot thinks it moves along (magenta). However, the reconstructed path is rather far from the ground-truth one because of presence of real motion noise. As a result, state errors higher than 3-sigma are obtained.

3) Number of particles decreases in PF filter

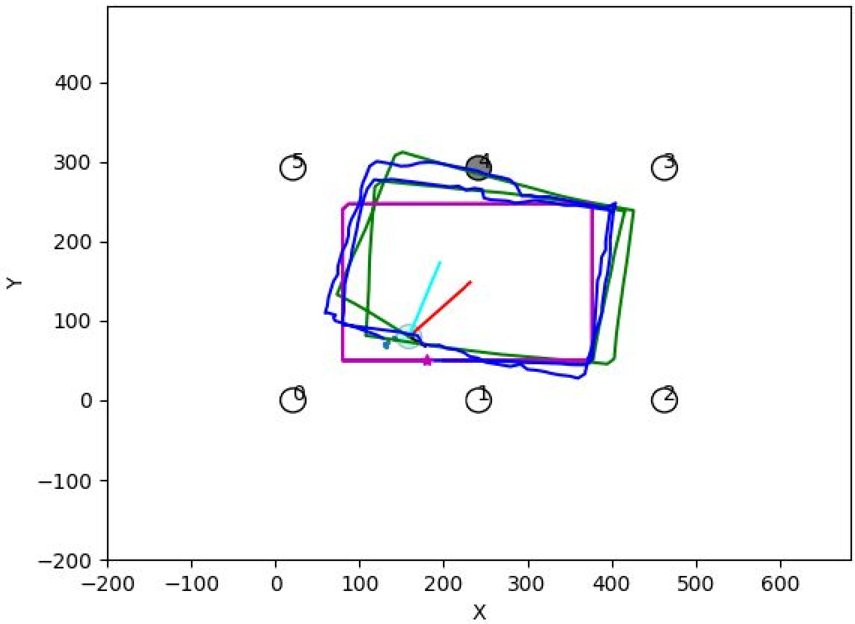

Figure-9. Simulator plot. N = N/10

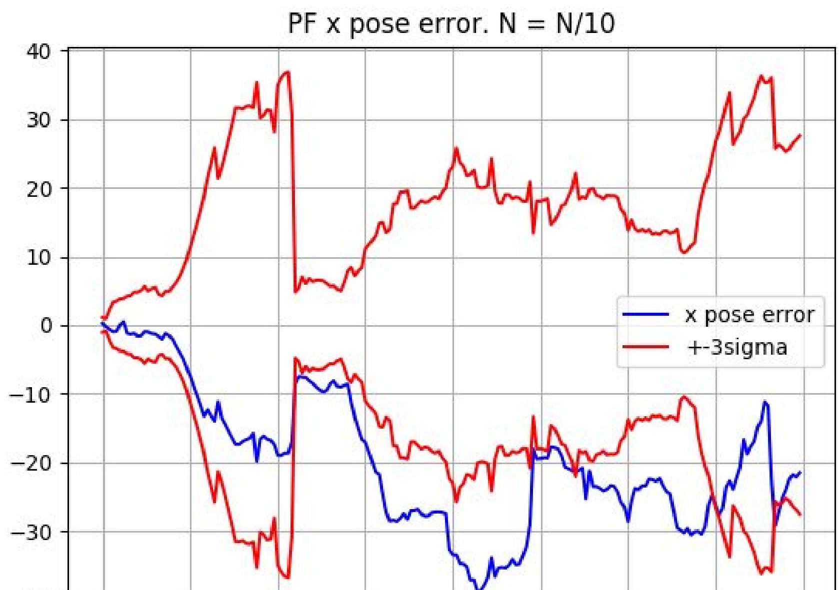

Figure-10. PF x pose error. N = N/10

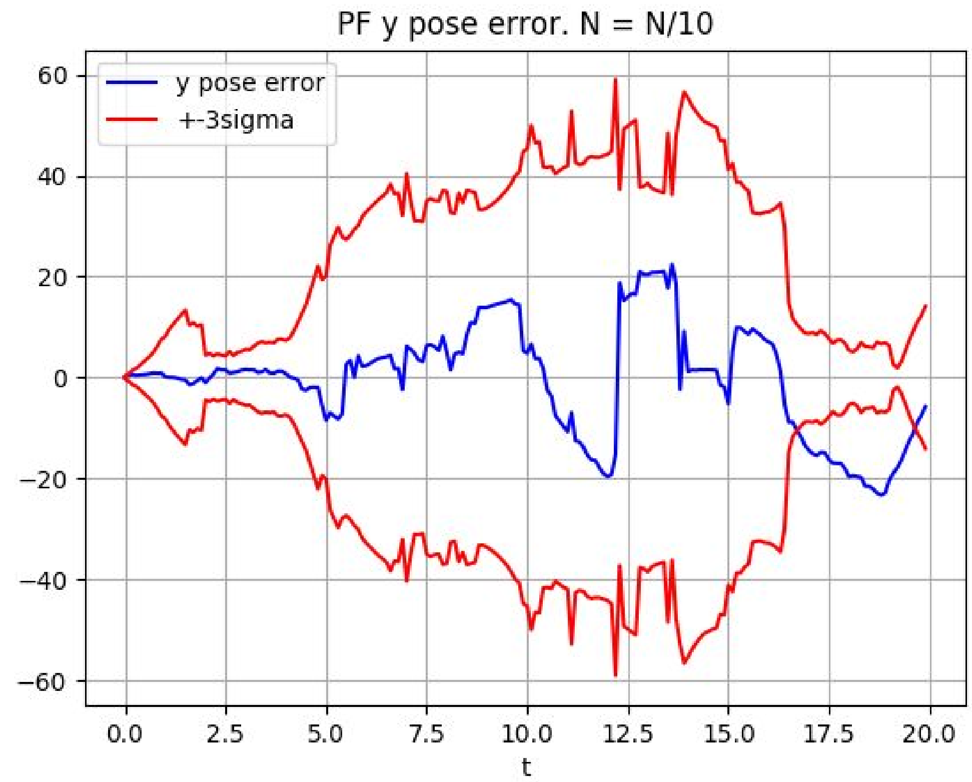

Figure-11. PF y pose error. N = N/10

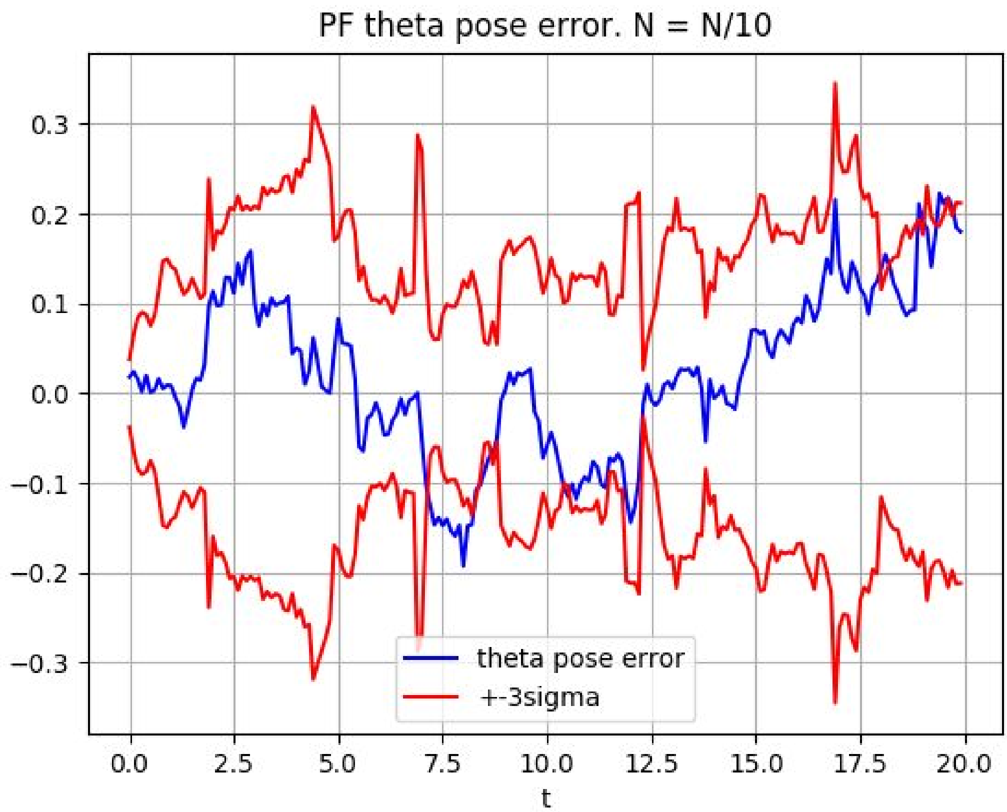

Figure-12. PF theta pose error. N = N/10

When a number of particles N is reduced, i.e. by 10, state errors become sometimes higher than 3-sigma bounds. It occurs because of low amount of particles. Less amount of particles, less the precision of localization.

4) The filter noise parameters underestimate or overestimate the true noise parameters

When the filter noise parameters **underestimate** the true noise parameters, state errors become higher than 3-sigma uncertainty bounds which is a low localization accuracy.

When the filter noise parameters **overestimate** the true noise parameters, 3-sigma uncertainty bounds become so wide that calculated assumed mean position of the robot becomes far away from the real robot's path despite of state errors are inside of uncertainty bounds. As a result, low localization accuracy is obtained too.

A case of overestimated sensor noise with covariance Q = Q*9 is provided as a proof.

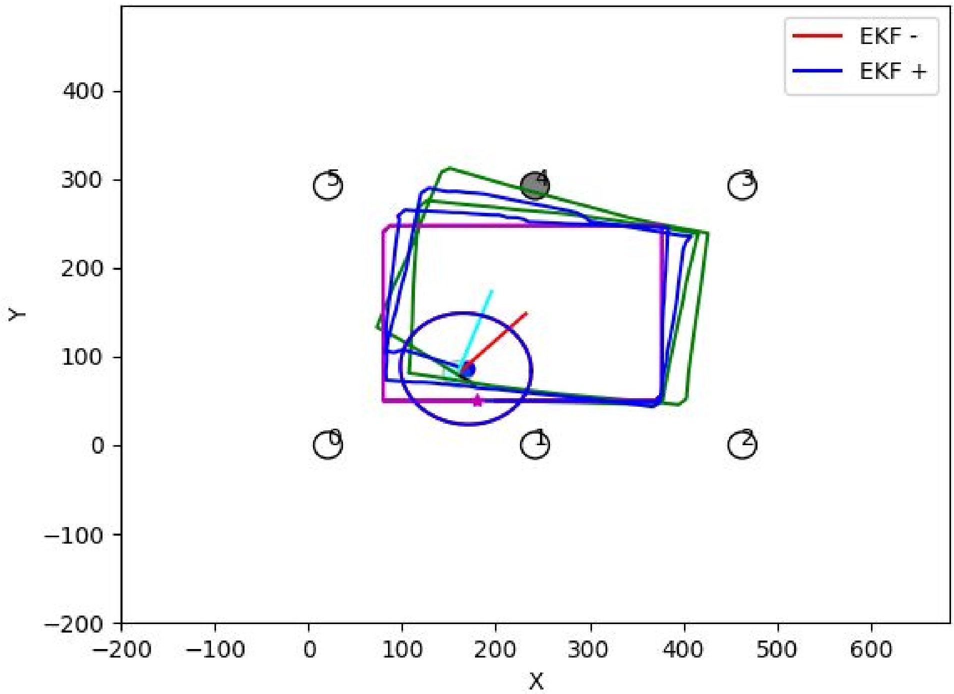

Figure-13. Simulator plot. Q = Q*9

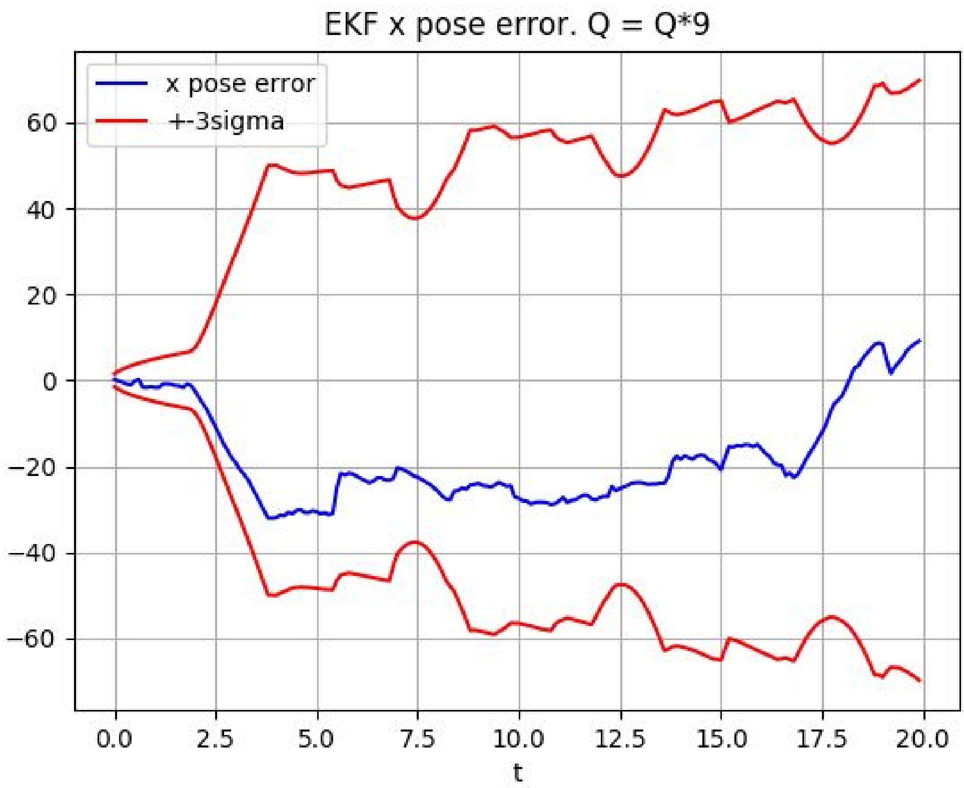

Figure-14. EKF x pose error. Q = Q*9

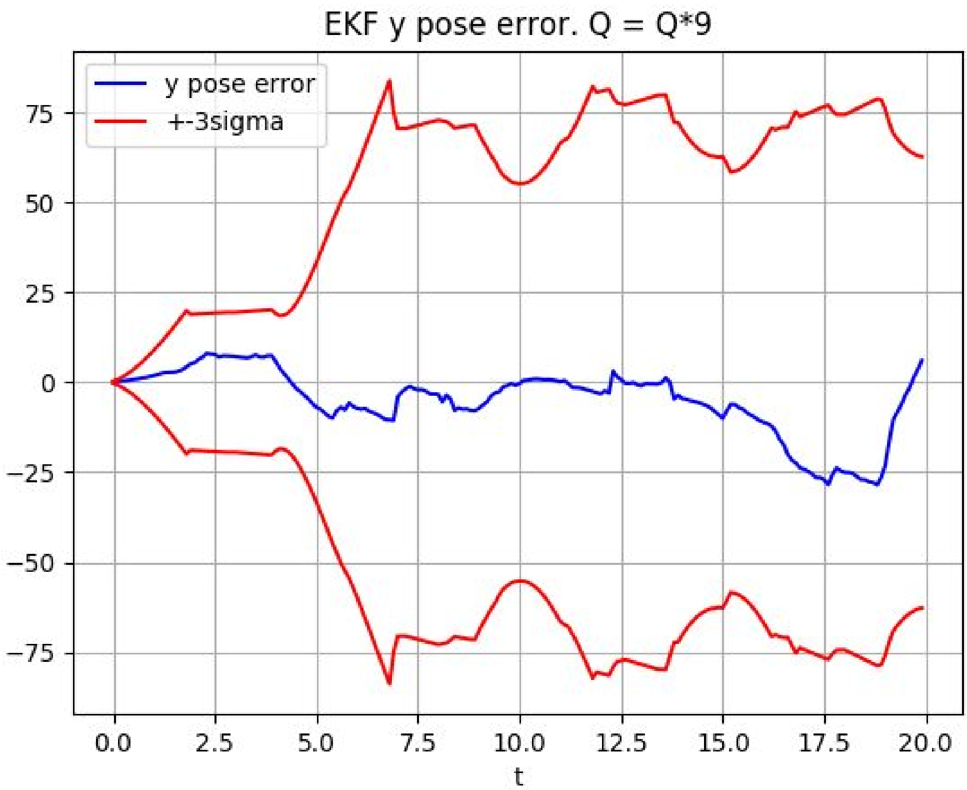

Figure-15. EKF y pose error. Q = Q*9

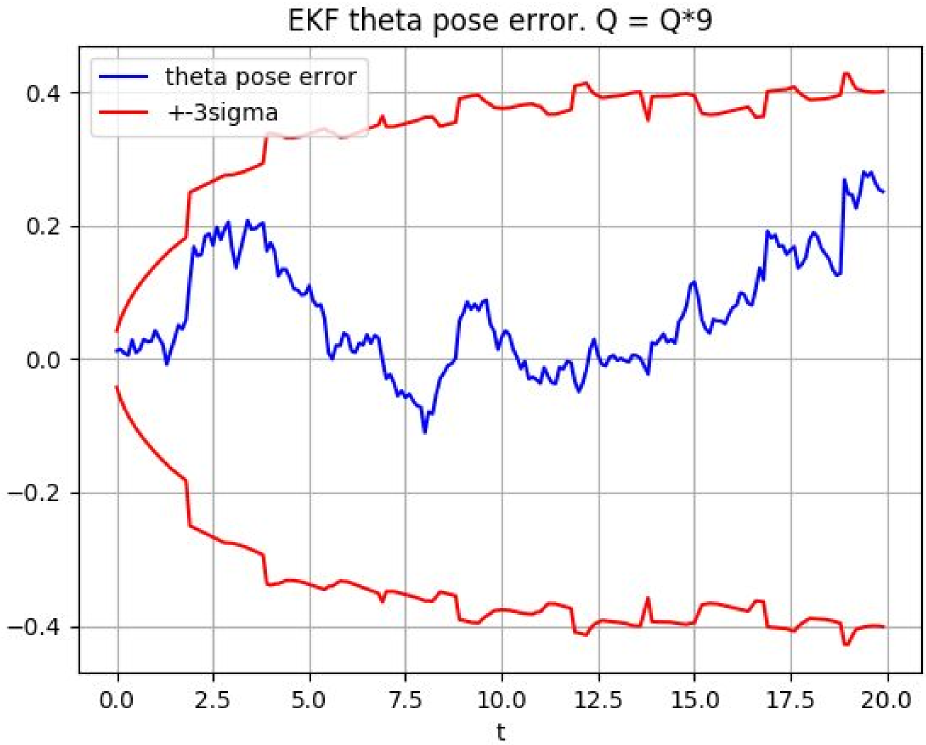

Figure-16. EKF theta pose error. Q = Q*9