PMR3502 - Elementos de robótica **Lista 02**

Bruno Scaglione NUSP: 10335812
Lucas Hattori Costa NUSP: 10335847
Data: 27 de junho de 2021

Descrição do sistema

Um robô tricilo como o da Figura 1 se desloca no plano xy. As rodas motoras giram em uma velocidade determinada por um controlador. O ângulo de basculamento da roda dianteira também é controlado.

Mede-se a distância do robô até uma antena, bem como o fluxo magnético terrestre no referencial do robô. A posição da antena é conhecida. O fluxo magnético terrestre no local é presumido constante, uniforme (independente da posição do veículo) e desconhecido. Adota-se neste problema o vetor de estado do sistema dado por:

$$X_t = \begin{bmatrix} x \\ y \\ \theta \\ f_x \\ f_y \end{bmatrix}$$

onde x, y são as coordenadas da posição do robô, θ é o ângulo do eixo longitudinal do mesmo com o eixo x, f_x, f_y são as componentes do vetor fluxo magnético terrestre no local. A ação de controle do sistema é representada pelo vetor:

$$u_t = \begin{bmatrix} v \\ \phi \end{bmatrix}$$

onde v é a velocidade longitudinal imposta pelas rodas motoras e ϕ é o ângulo de basculamento da roda dianteira.

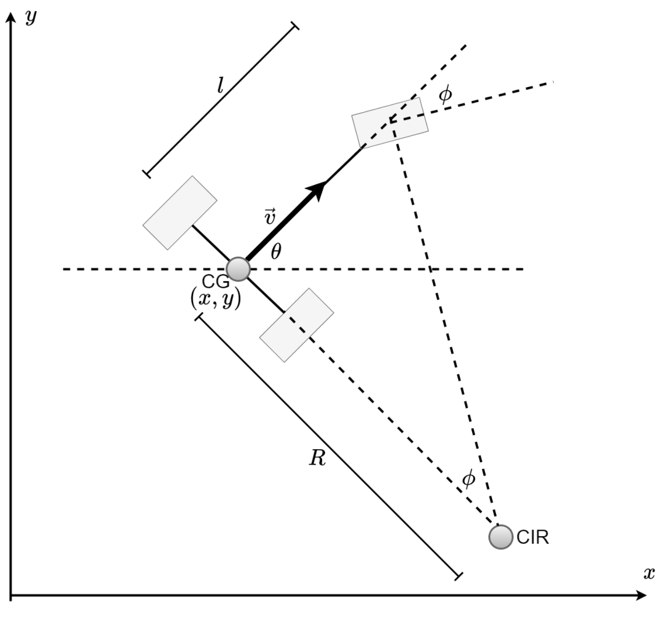


Figura 1: Diagrama do enunciado

Questões

Questão 1

Escreva a lei de recorrência de evolução do vetor de estado na seguinte forma:

$$X_t = F\left(X_{t-1}, u_t\right) + \varepsilon_t$$

onde ε_t é um vetor de 5 componentes aleatório Gaussiano de média nula e matriz de covariância R. Escreva a expressão para $F\left(X_{t-1}, u_t\right)$ (você pode usar $x_t, y_t, \theta_t, f_{x_t}, f_{y_t}$ para referenciar as componentes de X_t e v_t, ϕ_t para as componentes de u_t). Considere que em cada instante a função F representa o deslocamento do robô em um arco de circunferência, determinado por v e ϕ.

Escreva a matriz R que representa a covariância da perturbação aleatória que se soma ao deslocamento determinístico.

Considere para isso que ao final do arco de circunferência percorrido pelo robô soma-se um deslocamento aleatório longitudinal, um deslocamento radial e um deslocamento angular. Estes três deslocamentos são independentes, Gaussianos de média nula e covariâncias $s^2_{l^2_t}, s^2_{rt}$ e $s^2_{\theta t}$ respectivamente. As covariâncias $s^2_{l_t}$ e s^2_{rt} são de medidas de perturbação en um referencial ideal que faz um ângulo de $\hat{\theta}_t$ com o eixo x. Esta orientação seria a do eixo longitudinal do veículo se este descrevesse um arco perfeito de circunferência no instante t de modo que $\hat{\theta}_t = F\left(X_{t-1}, u_t\right)_3$ (ou seja, a orientação determinada pela função F sem ruído). A matriz R, por outro lado, está escrita no referencial Global. Neste referencial, as perturbações em x e y podem não ser independentes. Escreva a matriz R em função de $\hat{\theta}_t$.

***Solução**:* Pelo diagrama do enunciado, é possível deduzir cinco equações, cada qual referente à uma das variáveis de estado:

$$x_t = x_{t-1} + [v_{t-1} \cdot \cos(\theta_{t-1})] \cdot \Delta t \tag{1}$$
$$y_t = y_{t-1} + [v_{t-1} \cdot \sin(\theta_{t-1})] \cdot \Delta t \tag{2}$$
$$\theta_t = \theta_{t-1} + \frac{v_{t-1}}{R_{t-1}} \cdot \Delta t \quad \text{onde: } \; R_{t-1} = l \cdot \tan\left(\frac{\pi}{2} - \phi_{t-1}\right) = \frac{l}{\tan\left(\phi_{t-1}\right)} \tag{3}$$
$$f^x_t = f^x_{t-1} \tag{4}$$
$$f^y_t = f^y_{t-1} \tag{5}$$

Com tais equações, é possível escrever a expressão matricial de $F(X_{t-1}, u_{t-1})$ da seguinte forma. Na equação a seguir e no decorrer da lista, utilizamos a notação X_{i_j} onde i indica o índice da variável de estado e j o passo temporal.

$$X_{d_t} = F\left(X_{t-1}, u_{t-1}\right) = \begin{bmatrix} X_{1_{t-1}} \\ X_{2_{t-1}} \\ X_{3_{t-1}} \\ X_{4_{t-1}} \\ X_{5_{t-1}} \end{bmatrix} + \begin{bmatrix} u_{1_{t-1}} \cdot \cos(X_{3_{t-1}}) \\ u_{1_{t-1}} \cdot \sin(X_{3_{t-1}}) \\ \frac{u_{1_{t-1}} \cdot \tan(\phi_{t-1})}{l} \\ 0 \\ 0 \end{bmatrix} \cdot \Delta t \tag{6}$$

onde X_{d_t} é a parte determinística, a extrapolação do estado.

Para se obter a matriz da covariância, primeiramente identificamos as parcelas determinística e estocástica do deslocamento, como abaixo:

$$X_t = \underbrace{\begin{bmatrix} x_{d_t} \\ y_{d_t} \\ \theta_{d_t} \\ f^x_{d_t} \\ f^y_{d_t} \end{bmatrix}}_{X_{d\,t}} + \underbrace{\begin{bmatrix} x_{s_t} \\ y_{s_t} \\ \theta_{s_t} \\ 0 \\ 0 \end{bmatrix}}_{X_{s\,t}} \tag{7}$$

Onde: $X_{s\,t} = F(X_{s\,t-1})$; o subscrito d indica variáveis que se referem a um resultado determinístico, proveniente da extrapolação de estado sem perturbação; e o subscrito s indica variáveis determinadas pela distribuição Gaussiana, ou seja, são resultados estocásticos.

$$X_{s\,t-1} = \underbrace{\begin{bmatrix} Rot^l_{g(2\times2)} & 0 & 0 & 0 \\ 0 \quad 0 & 1 & 0 & 0 \\ 0 \quad 0 & 0 & 1 & 0 \\ 0 \quad 0 & 0 & 0 & 1 \end{bmatrix}}_{M} \cdot \begin{bmatrix} S^T_{t-1(3\times1)} \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

onde: R^l_g é a matriz de mudança de base, ou matriz de rotação, do referencial local dos distúrbios para o referencial global fixo na Terra; e $S^T_{t-1} = [l_s\ r_s\ \theta_s]$ e $S^T_{t-1} \sim \mathcal{N}(\mu = 0, \Sigma = R_{l\,t-1})$ onde $R_{l\,t-1}$ é a matriz de covariância no referencial local dada pelo enunciado.

$$R_{l\,t-1} = \begin{bmatrix} s_{l\,t-1} & 0 & 0 & 0 & 0 \\ 0 & s_{r\,t-1} & 0 & 0 & 0 \\ 0 & 0 & s_{\theta\,t-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{9}$$

onde $s_{l_{t-1}} = \left(\frac{v_{t-1}\cdot\Delta t}{6}\right)^2$, $s_{r_{t-1}} = \left(\frac{v_{t-1}\cdot\Delta t}{12}\right)^2$ e $s_{\theta_{t-1}} = \left(\frac{v_{t-1}\cdot\Delta t}{8\cdot l}\right)^2$

$$Rot^l_{g\,t-1} = \begin{bmatrix} \cos(\theta_{d\,t-1}) & -\sin(\theta_{d\,t-1}) \\ \sin(\theta_{d\,t-1}) & \cos(\theta_{d\,t-1}) \end{bmatrix} \tag{10}$$

Finalmente, pela propriedade de multiplicação por constante, da covariância:

$$\Sigma(X_{s_{t-1}}) = M \cdot \Sigma(S^T_{t-1}) \cdot M^T \tag{11}$$

Substituindo as matrizes de covariância:

$$R_g = M \cdot R_l \cdot M^T \tag{12}$$

onde R_g é a matriz de covariância global que vai determinar $\left[x_{s_{t-1}}\ y_{s_{t-1}}\ \theta_{s_{t-1}}\ 0\ 0\right]^T$

Questão 2
As medições são representadas pelo vetor:

$$z_t = \begin{bmatrix} \rho \\ b_f \\ b_e \end{bmatrix}$$

Onde ρ é medição da distância do robô até uma antena, b_l é a medição da projeção do campo magnético terrestre na direção longitudinal do robô (positiva para a dianteira) e b_t é a medição do campo magnético na direção transversal (positiva para a esquerda). A antena está sobre a origem do sistema de coordenadas x, y e a uma altura h. Estas medições são feitas com ruídos aleatórios Gaussianos independentes de média nula e covariâncias $s^2_{\rho t}, s^2_{ft}$es^2_{et} respectivamente. Escreva a lei de medição do sistema na seguinte forma:

$$z_t = G\left(X_t\right) + \delta_t$$

onde δ_t é um vetor aleatório Gaussiano de média nula e matriz de covariância Q_t dada por:

$$Q_t = \begin{bmatrix} s^2_{\rho_t} & 0 & 0 \\ 0 & s^2_{f_t} & 0 \\ 0 & 0 & s^2_{et} \end{bmatrix}$$

Solução:

$$z_t = \begin{bmatrix} \sqrt{x_t^2 + y_t^2 + h^2} \\ \\ f_t^x \cdot \cos\left(\theta_t\right) + f_t^y \cdot \sin\left(\theta_t\right) \\ \\ -f_t^x \cdot \sin\left(\theta_t\right) + f_t^y \cdot \cos\left(\theta_t\right) \end{bmatrix} + \delta_{t\,(3\times 1)} \tag{13}$$

onde $\delta_t \sim \mathcal{N}(\mu = 0,\, \Sigma = Q_{\,t})$

Questão 3
No filtro extendido de Kalman, a estimativa de covariância do estado antes da incorporação da medição é dada por:

$$\bar{\Sigma}_t = A_t \Sigma A_t^T + R_t$$

onde a matriz A_t é dada por $\nabla \times F(X, u)$. Escreva a matriz A_t do sistema em função de X_{t-1} e u_t

Solução:

$$A_{t-1} = \nabla \times F(X_{t-1}, u_{t-1}) = A_{t-1}(\theta_{t-1}, v_{t-1}) = \begin{bmatrix} 1 & 0 & -v_{t-1} \cdot \sin\left(\theta_{t-1}\right) \cdot \Delta t & 0 & 0 \\ 0 & 1 & v_{t-1} \cdot \cos\left(\theta_{t-1}\right) \cdot \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{14}$$

Questão 4
Implemente o passo de previsão do Filtro Extendido utilizando a linguagem de programação que achar mais adequada (Sugestão: Python com o pacote numpy). Neste passo você deve estimar o estado recursivamente com as equações:

$$\bar{\mu}_t = F\left(\mu_{t-1}, u_t\right)$$
$$\bar{\Sigma}_t = A_t \Sigma A_t^T + R_t$$

Note que as matrizes A_t e R_t dependem da estimativa de estado μ_{t-1} e da entrada u_t. Os valores das covariâncias $s_{l_t}^2, s_{rt}^2$ e $s_{\theta_t}^2$ dependem do parâmetro v_t (o primeiro coeficiente de u_t) de acordo com as seguintes expressões:

$$s_{l_t}^2 = \left(\frac{\Delta t v_t}{6}\right)^2$$
$$s_{rt}^2 = \left(\frac{\Delta t v_t}{12}\right)^2$$
$$s_{\theta_t}^2 = \left(\frac{\Delta t v_t}{8l}\right)^2$$

Onde l é a distância entre os eixos do veiculo e vale 0,3 . Como nesta etapa *não* são consideradas medições, use $\mu_t = \bar{\mu}_t \mathrm{e} \Sigma_t = \Sigma_t$. Para o estado inicial, use:

$$\mu_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ e } \Sigma_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 10 \end{bmatrix}$$

O seu programa deve processar uma entrada no formato descrito pela seção "Dados". Processe os parâmetros u_t no arquivo csv anexado neste enunciado. Plote os valores de x_t, y_t de $\bar{\mu}_t$ no plano x, y.

Considere a distância entre os eixos do veículo $l = 0,3m$ e $\delta t = 0,25$.

Quais são os valores das covariâncias de x e y para $t = 250$?

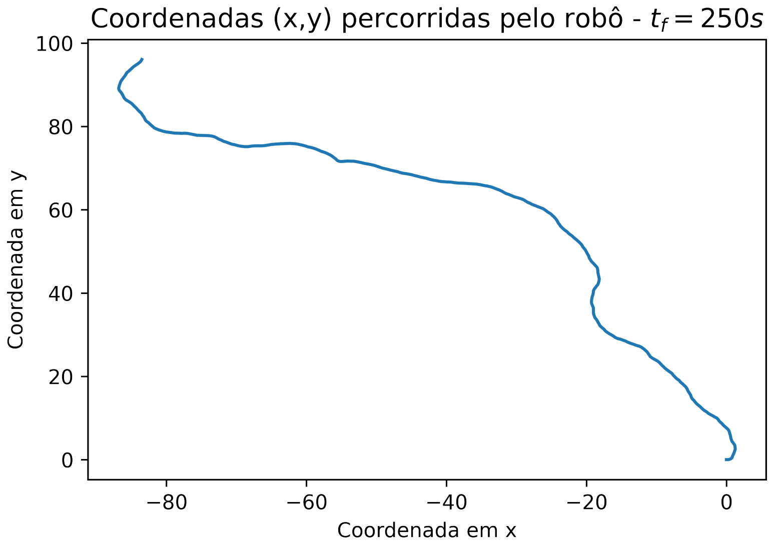
Solução: Utilizando o método `predict` da classe EKF desenvolvida, obtêm-se a seguinte matriz de covariância:

$$\bar{\Sigma}_{t=250} = \begin{bmatrix} 16014.5 & 16888.8 & -273.8 & 0 & 0 \\ 16888.8 & 18574.0 & -293.9 & 0 & 0 \\ -273.8 & -293.9 & 6.5 & 0 & 0 \\ 0 & 0 & 0 & 10.0 & 0 \\ 0 & 0 & 0 & 0 & 10.0 \end{bmatrix} \quad (15)$$

logo, para o instante pedido:

- $cov(\bar{x}, \bar{x}) = 16014.5$
- $cov(\bar{y}, \bar{y}) = 18574.0$
- $cov(\bar{x}, \bar{y}) = 16888.8$

Plotando $(\bar{x}_t, \bar{y}_t)$ de $\bar{\mu}_t$, obtém-se a seguinte imagem:



Questão 5
No filtro extendido de Kalman o ganho de Kalman é dado por:

$$K_t = \bar{\Sigma}_t C_t^T \left(C_t \bar{\Sigma}_t C_t^T + Q_t \right)^{-1}$$

Onde a matriz C_t é definida como $\nabla \times G(X)$. Escreva a matriz C_t em função de X_t.

Solução:

$$C_t = \nabla \times G_t(X_t) = \begin{bmatrix} \frac{x_t}{\sqrt{x_t^2+y_t^2+h^2}} & \frac{y_t}{\sqrt{x_t^2+y_t^2+h^2}} & 0 & 0 & 0 \\ 0 & 0 & f_y \cos(\theta_t) - f_x \sin(\theta_t) & \cos(\theta_t) & \sin(\theta_t) \\ 0 & 0 & -f_x \cos(\theta_t) - f_y \sin(\theta_t) & -\sin(\theta_t) & \cos(\theta_t) \end{bmatrix}$$

Questão 6
Complete a implementação de seu filtro de Kalman. Agora você deve estimar o estado e sua covarância de acordo com as leis:

$$\mu_t = \bar{\mu}_t + K_t \left(t - G\left(\bar{\mu}_t\right)\right)$$
$$\Sigma_t = \left(I - K_t C_t\right) \bar{\Sigma}_t$$

Note que a matriz C_t depende da estimativa de estado $\bar{\mu}_t$. Use $h = 0,5$ O valor $s_{\rho t}^2$ deve ser calculado a partir do estado estimado do sistema $\bar{\mu}_t$ de acordo com a seguinte expressão:

$$s_{\rho t}^2 = \frac{h^2 + \bar{x}_t^2 + \bar{y}_t^2}{20^2}$$

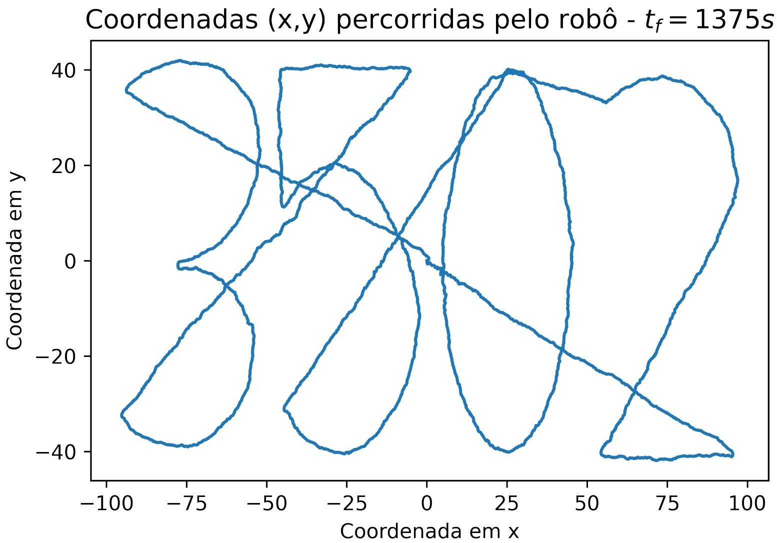
onde $\bar{x}_t$ e $\bar{y}_t$ são respectivamente a primeira e segunda componentes de $\bar{\mu}_t$. Considere $s_{ft}^2 = s_{et}^2 = 1/4$.

Novamente, seu programa deve processar os dados anexados a este enunciado. Plote os valores de x_t, y_t de μ_t no plano x, y.

***Solução**:* Utilizando o método `predict` e `update` da classe EKF desenvolvida, obtêm-se a seguinte matriz de covariância:

$$\bar{\Sigma}_{t=250} = \begin{bmatrix} 1.08e-02 & 2.99e-03 & 5.14e-06 & -8.06e-06 & -3.97e-06 \\ 2.98e-03 & 8.31e-04 & 1.33e-06 & -2.12e-06 & -1.05e-06 \\ 5.14e-06 & 1.33e-06 & 2.67e-04 & -2.50e-04 & -1.24e-04 \\ -8.07e-06 & -2.12e-06 & -2.50e-04 & 5.00e-04 & 2.24e-04 \\ -3.98e-06 & -1.05e-06 & -1.24e-04 & 2.24e-04 & 1.56e-04 \end{bmatrix}$$

Logo, para o instante pedido: $cov(\bar{x}, \bar{x}) = 1.08e{-}02$; $cov(\bar{y}, \bar{y}) = 8.31e{-}04$ $cov(\bar{x}, \bar{y}) = 2.99e{-}03$. Plotando os pontos $(\bar{x}_t, \bar{y}_t)$ no plano, obtem-se a seguinte figura:



Código Desenvolvido

June 26, 2021

0.1 Libraries and data

[1]:
```python
import numpy as np
import pandas as pd
import numpy.linalg as LA
import matplotlib.pyplot as plt

data = np.genfromtxt('./data/valores.csv', delimiter=',')

%matplotlib inline
```

0.2 Global parameters

[2]:
```python
l = 0.3
h = 0.5
dt = 0.25
t_end = 250
n_steps = int(t_end/dt)
init_sigma = np.zeros((5, 5))
init_sigma[3,3] = 10
init_sigma[4,4] = 10
init_state = np.zeros((5,))
```

0.3 Auxiliary Functions

[3]:
```python
def jacobian(X: np.array, u: np.array, which: str) -> np.array:
    '''
    Calculados analiticamente.
    '''
    if which=='A':
        ddx = (-u[0]*np.sin(X[2])*dt)
        ddy = (u[0]*np.cos(X[2])*dt)
        J = [
            [1, 0, ddx, 0, 0],
            [0, 1, ddy, 0, 0],
            [0, 0, 1, 0, 0],
```

```
            [0, 0, 0, 1, 0],
            [0, 0, 0, 0, 1]
        ]
    elif which=='C':
        sin = np.sin(X[2])
        cos = np.cos(X[2])
        J = [
            [X[0]/np.sqrt(X[0]**2 + X[1]**2 + h**2), X[1]/np.sqrt(X[0]**2 +
 ↪X[1]**2 + h**2), 0, 0, 0],
            [0,      0,     -X[3]*sin + X[4]*cos, cos, sin],
            [0,      0,     -X[3]*cos - X[4]*sin, -sin, cos]
        ]

    return np.array(J, dtype='float')

def F(X_prev: np.array, u_prev: np.array) -> np.array:
    '''
    Calculada analiticamente.
    '''
    return np.array([
        X_prev[0] + u_prev[0] * np.cos(X_prev[2]) * dt,
        X_prev[1] + u_prev[0] * np.sin(X_prev[2]) * dt,
        X_prev[2] + u_prev[0] * dt * np.tan(u_prev[1])/ l,
        X_prev[3],
        X_prev[4]
    ])

def Q(x_prev):
    sf, se = 0.25, 0.25
    srho = ((h**2 + x_prev[0]**2 + x_prev[1]**2) / (20**2))
    return np.array([
        [srho, 0, 0],
        [0, sf, 0],
        [0, 0, se]
    ])

def R(x_prev, u):
    '''
    DE ESTADO
    '''
    cos = np.cos(x_prev[2])
    sin = np.sin(x_prev[2])
    M =  np.array([
        [cos, -sin, 0, 0, 0],
        [sin,  cos, 0, 0, 0],
        [0, 0, 1, 0, 0],
        [0, 0, 0, 1, 0],
```

```
            [0, 0, 0, 0, 1]
        ])

        sl = (dt * u[0] / 6)**2
        sr = (dt * u[0] / 12)**2
        stheta = (dt * u[0] / (8*l))**2

        # Matriz R no referencial local
        Rl = np.array([
            [sl, 0, 0, 0, 0],
            [0, sr, 0, 0, 0],
            [0, 0, stheta, 0, 0],
            [0, 0, 0, 0, 0],
            [0, 0, 0, 0, 0]
        ])

        return M @ Rl @ M.T

    def G(x_prev):
        sin = np.sin(x_prev[2])
        cos = np.cos(x_prev[2])
        return np.array([
            np.sqrt(x_prev[0]**2 + x_prev[1]**2 + h**2),
            ( x_prev[3] * cos + x_prev[4] * sin),
            (-x_prev[3] * sin + x_prev[4] * cos)
        ])
```

0.4 EKF

Implementação do Filtro de Kalman extendido

[4]:
```
class EKF:
    def __init__(self, f, g, x_est, Sigma_est, Q, R, data):
        '''EKF(f, h, x0, u0)

        x' = f(x, u)    state transition function
        z' = g(x)       observation function
        x0              initial state estimate
        Sigma0          initial cov matrix estimate
        Q               stochastic matrix for observation
        R               stochastic matrix for state'''
        self.f, self.g = f, g
        self.x = np.array(x_est)
        self.I = np.eye(len(x_est))
        self.Sigma = Sigma_est
        self.Q, self.R = Q, R
        self.u = data[:,:2]
```

```
        self.z = data[:,2:]
        self.X = np.array([self.x])
        self.first = True

    def predict(self, t: int):
        A = jacobian(X=self.x, u=self.u[t], which='A')
        self.Sigma = (A @ self.Sigma @ A.T) + self.R(self.x, self.u[t])
        self.x = self.f(self.x, self.u[t])
        self.X = np.append(self.X,[self.x], axis=0)

    def update(self, t: int):
        x, Sigma, Q, I, g, z = self.x, self.Sigma, self.Q, self.I, self.g, self.z
        C = jacobian(X=x, u=z[t], which='C')
        y = z[t] - g(x)
        y = np.array([y]).T
        S = (C @ Sigma @ C.T) + Q(x)
        K = Sigma @ C.T @ LA.inv(S)
        innovation = K @ y

        self.x = np.array([x+innovation[i][0] for (i,x) in np.ndenumerate(self.
↪x)])
        self.Sigma = (I - K @ C) @ Sigma
```

0.5 Questão 04

Como nessa etapa não são consideradas as medições, o filtro de Kalman se reduz à parte de predição.

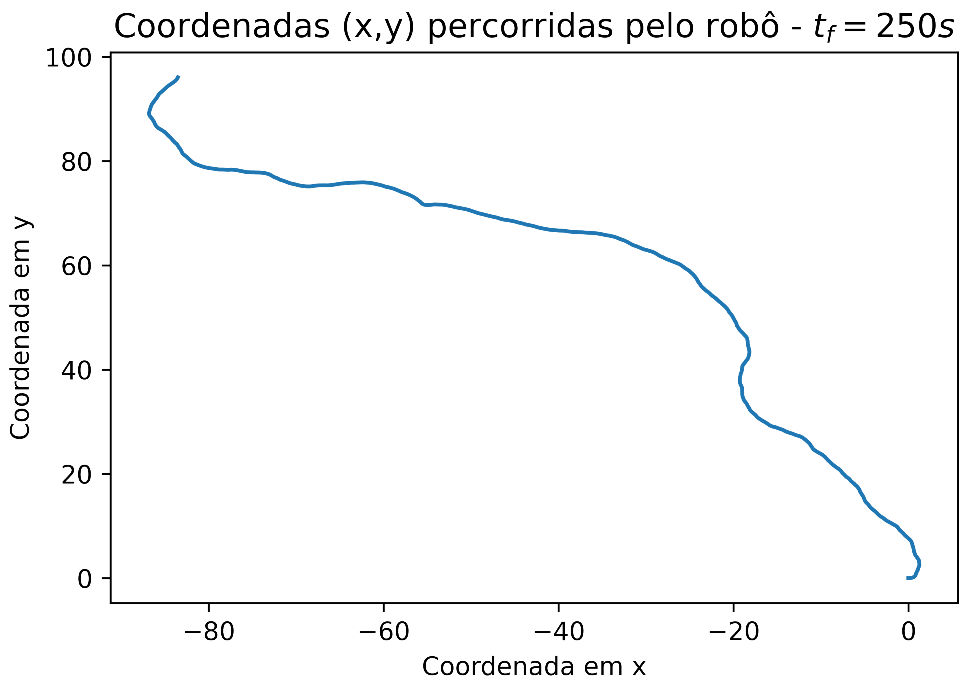
[5]:
```
ekf_q4 = EKF(F, G, init_state, init_sigma, Q, R, data)
for t in range(n_steps):
    ekf_q4.predict(t)
```

[6]:
```
pd.DataFrame(ekf_q4.Sigma, columns=['x','y',r'$\theta$',r'f_x',r'f_y'])
```

[6]:

	x	y	θ	f_x	f_y
0	16014.490169	16888.802363	-273.775722	0.0	0.0
1	16888.802363	18573.995386	-293.912740	0.0	0.0
2	-273.775722	-293.912740	6.510417	0.0	0.0
3	0.000000	0.000000	0.000000	10.0	0.0
4	0.000000	0.000000	0.000000	0.0	10.0

[7]:
```
plt.plot(ekf_q4.X[:,0], ekf_q4.X[:,1])
plt.ylabel('Coordenada em y')
plt.xlabel('Coordenada em x')
plt.title(r'Coordenadas (x,y) percorridas pelo robô - $t_f=250s$', fontsize=13)
plt.savefig('./images/caminhoQ4.png', dpi=600)
plt.show()
```



0.6 Questão 06

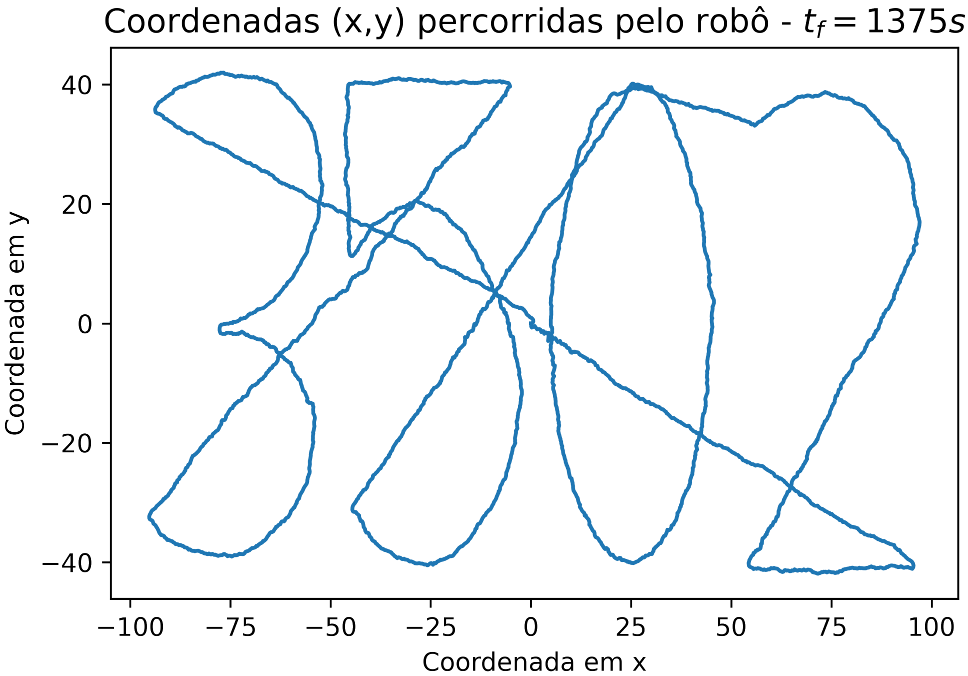
[8]:
```
ekf_q6 = EKF(F, G, init_state, init_sigma, Q, R, data)
for t in range(int(1375/dt)):
    ekf_q6.predict(t)
    ekf_q6.update(t)
```

[9]:
```
pd.DataFrame(ekf_q6.Sigma, columns=['x','y',r'$\theta$',r'f_x',r'f_y'])
```

[9]:

	x	y	θ	f_x	f_y
0	0.010847	0.002988	0.000005	-0.000008	-0.000004
1	0.002988	0.000831	0.000001	-0.000002	-0.000001
2	0.000005	0.000001	0.000267	-0.000250	-0.000124
3	-0.000008	-0.000002	-0.000250	0.000500	0.000224
4	-0.000004	-0.000001	-0.000124	0.000224	0.000156

[10]:
```
plt.plot(ekf_q6.X[:,0], ekf_q6.X[:,1])
plt.ylabel('Coordenada em y')
plt.xlabel('Coordenada em x')
plt.title(r'Coordenadas (x,y) percorridas pelo robô - $t_f=1375s$', fontsize=13)
plt.savefig('./images/caminhoQ6.png',dpi=600)
plt.show()
```



Coordenadas (x,y) percorridas pelo robô - $t_f = 1375s$
40
20
0
−20
−40
Coordenada em y
−100
−75
−50
−25
0
25
50
75
100
Coordenada em x