EE 6900
"Simultaneous Localization and Mapping (SLAM) for Ground and Aerial Robotics"

Spring 2015

Project # 3
(due Monday 16 March 2015 at midnight via e-mail)

The goal of this project is to perform the filter definition for an EKF SLAM implementation for the Victoria park data set and compute the unaided trajectory. In the next project we will then implement EKF SLAM.

The van used to collect the Victoria park data set is shown in Figure 1a and a diagram with the various dimensions is shown in Figure 1b.

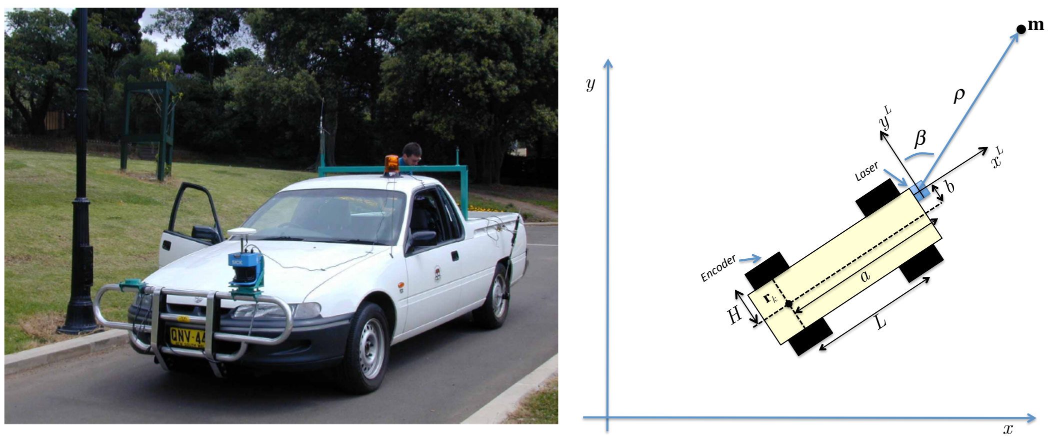


Figure 1. (a) Data collection van at Victoria Park (left); (b) dimensions and coordinate frames.

The various dimensions shown in Figure 1b are: H = 0.75m (distance between center axis and wheel encoder position), L = 2.83m (wheel base), a = L + 0.95m (distance in x-dimension to the laser scanner), and b = 0.5m (distance in y-direction to the laser scanner).

The control actions for the van in Figure 2, are speed, v, and steering angle, α, or:

$$\mathbf{u}_k = \begin{bmatrix} v_k & \alpha_k \end{bmatrix}^T$$

Given the van pose (position and orientation) at the reference point (between the rear wheels)

$$\mathbf{x}_c = \begin{bmatrix} x_c & y_c & \theta_c \end{bmatrix}^T$$

the dynamics model is given by the so-called Ackerman model [1] as:

$$\mathbf{x}_{c,k} = \mathbf{g}_c(\mathbf{x}_{c,k-1}, \mathbf{u}_k) = \begin{bmatrix} x_{c,k-1} + \dfrac{v_k \Delta t}{\left[1-(H/L)\tan(\alpha_k)\right]}\cos(\theta_{c,k-1}) \\ y_{c,k-1} + \dfrac{v_k \Delta t}{\left[1-(H/L)\tan(\alpha_k)\right]}\sin(\theta_{c,k-1}) \\ \theta_{c,k-1} + \dfrac{v_k \Delta t}{L\left[1-(H/L)\tan(\alpha_k)\right]}\tan(\alpha_k) \end{bmatrix}$$

However, all laser scanner measurements are made with respect to the laser scanner, not at the rear reference point. This means that, to correctly interpret the laser scanner measurements during the EKF update stage, the reference point should be moved to the point of the laser scanner. Mathematically this can be accomplished as follows [1]:

$$\mathbf{x}_k = \mathbf{g}(\mathbf{x}_{k-1}, \mathbf{u}_k) = \begin{bmatrix} x_{k-1} + \dfrac{v_k \Delta t}{\left[1-(H/L)\tan(\alpha_k)\right]}\left\{\cos(\theta_{k-1}) - \dfrac{1}{L}\left(a\sin(\theta_{k-1}) + b\cos(\theta_{k-1})\right)\tan(\alpha_k)\right\} \\ y_{k-1} + \dfrac{v_k \Delta t}{\left[1-(H/L)\tan(\alpha_k)\right]}\left\{\sin(\theta_{k-1}) + \dfrac{1}{L}\left(a\cos(\theta_{k-1}) - b\sin(\theta_{k-1})\right)\tan(\alpha_k)\right\} \\ \theta_{k-1} + \dfrac{v_k \Delta t}{L\left[1-(H/L)\tan(\alpha_k)\right]}\tan(\alpha_k) \end{bmatrix}$$

For each observed landmark 'i', the measurement vector is given by the range and bearing to that landmark, or:

$$\mathbf{z}_i = \begin{bmatrix} \rho & \beta \end{bmatrix}^T$$

Within the context of SLAM, however, the landmark is defined in terms of a position in the "world" frame and can thus be defined by:

$$\mathbf{m}_i = \begin{bmatrix} m_{x,i} \\ m_{y,i} \end{bmatrix} = \mathbf{g}^m(\mathbf{x}_k, \mathbf{z}_i) = \begin{bmatrix} x_k + \rho_i \sin\beta_i \cos\theta_k + \rho_i \cos\beta_i \sin\theta_k \\ y_k + \rho_i \sin\beta_i \sin\theta_k - \rho_i \cos\beta_i \cos\theta_k \end{bmatrix}$$

Inversely, the measurement equation is then given by:

$$\mathbf{z}_i = \mathbf{h}(\mathbf{x}_k, \mathbf{m}_i) = \begin{bmatrix} \sqrt{\left(x_k - m_{x,i}\right)^2 + \left(y_k - m_{y,i}\right)^2} \\ \operatorname{atan}\left(\dfrac{y_k - m_{y,i}}{x_k - m_{x,i}}\right) - \theta_k - \dfrac{\pi}{2} \end{bmatrix}$$

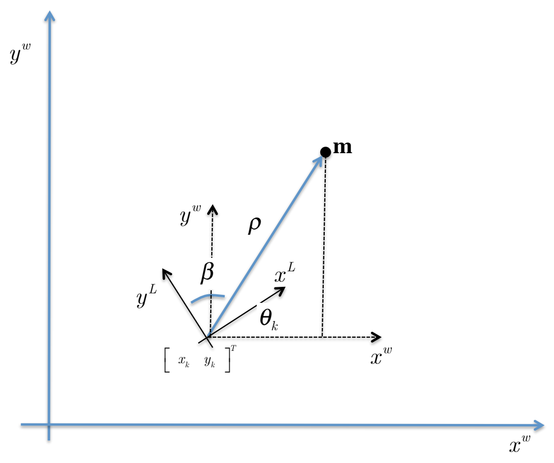


Figure 2. Measurement geometry.

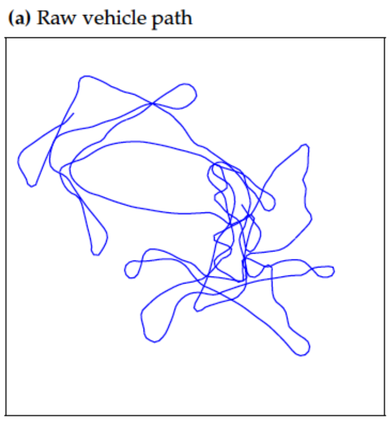
Given the above information perform the following tasks:

a) Derive all the Jacobians $\nabla \mathbf{g}_x, \nabla \mathbf{g}_u, \nabla \mathbf{g}_x^m, \nabla \mathbf{g}_z^m$ and $\nabla \mathbf{h}_y$ required to implement the Extended Kalman Filter (EKF) prediction step and show your work.

b) Implement a Matlab function that performs the EKF prediction step. The Matlab function should have the following header:

```
function [x, P] = prediction_step(x,P,u,Q,dt)
```

c) Run the Matlab template with your "prediction_step" routine and compare your results with those in Figure 13.10 of the textbook:



d) Write a routine that adds a new landmark to the state and covariance matrix. The header of the function would be:

```
function [x,P]= add_landmarks(x,P,z,R,cor)
```

where 'cor' is a vector whose elements indicate if a particular measurement is new and should be added (corresponding entry in 'cor' is -1) or already existing (corresponding entry in ‘cor’ equals the index of the landmark i.e. 4 for landmark 4). An example would be cor(1) = -1 means that measurement 1 is a measurement of a new landmark, whereas cor(4) = 6 means that the 4th measurement is a measurement of landmark 6.

References

[1] J. Guivant, et al., “Autonomous Navigation and Map building Using Laser Range Sensors in Outdoor Applications,” Journal of Robotics, Vol. 17, No. 10, October 2000, pp. 565-583.