Massachusetts Institute of Technology
Department of Mechanical Engineering

2.160 Identification, Estimation, and Learning
Fall 2021

Context-Oriented Project No. 2
Simultaneous Localization and Measurement

Out: October 20, 2021 Due: October 28, 2021
Study Group Discussion: October 22, 2021

In this context-oriented project, you will simultaneously estimate the pose (position and orientation) of a wheeled robot in a room, and the position and orientation of the 10 walls in the room. A map of the room is shown in Figure 2. First, you will estimate the pose trajectory of the robot based solely on odometry (wheel rotation measurement) data motor encoders. Next, you will integrate LIDAR data to leverage knowledge of the structure of the room to improve the pose estimate from that of odometry alone. This process simultaneously allows for improvement of the robot's model of the room. The Extended Kalman Filter will be the filter of choice for this analysis.

Section 1: Trajectory Prediction using Odometry

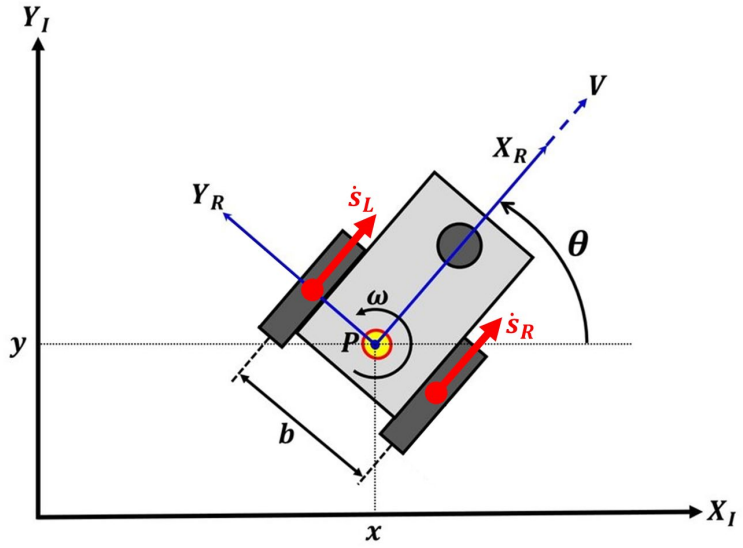

Figure 1 Diagram illustrating main odometry-related quantities

Refer to the SLAM Lecture Slides for a treatment of the kinematic relationship modeling motion of a two-wheeled robot over a time-step. You will use this model, along with the wheel path-distance data in `odom.csv`, to produce a prediction of the pose trajectory of the robot, which propagates covariance.

a) In order to use a Kalman Filter-like algorithm to propagate belief, it is important to have a Gaussian initial belief; that is, a belief based on an expected value (state estimate) and on belief covariance.
The central point of the robot, P in Figure 1, has a uniform probability of starting anywhere in a square starting region, marked red in the map in Figure 2. The initial orientation of the robot is independent of its starting position, and has a uniform probability of being between $-5\pi/9$ and $-4\pi/9$ radian (centered about pointing in the negative y direction).
Find the mean and covariance of the specified multivariate uniform initial belief. A multivariate Gaussian distribution with the same mean and covariance will form the initial belief used in the Extended Kalman Filter.
Hint: When random variables are independent, their joint distribution can be factored as a product of single-variable distributions.

b) Download and examine `odom.csv`. Its contents are described in detail in `README.txt`. This file contains the accumulated lengths of the paths traced by the right and left wheels of the robot, s_R and s_L respectively. The wheel displacements Δs_R and Δs_L for each time step can be calculated by subtracting s_R and s_L between time points. **Use these measurements to generate a prediction model to predict the pose trajectory of the robot, as well as propagate the estimation error covariance.** For this prediction model, use an Extended Kalman Filter with no measurement updates implemented. **Plot both the estimated pose trajectory and covariance over time.**

c) **Discuss the evolution of the covariance matrix over time, and how it relates to the robot's activity.**

Section 2: SLAM using LIDAR

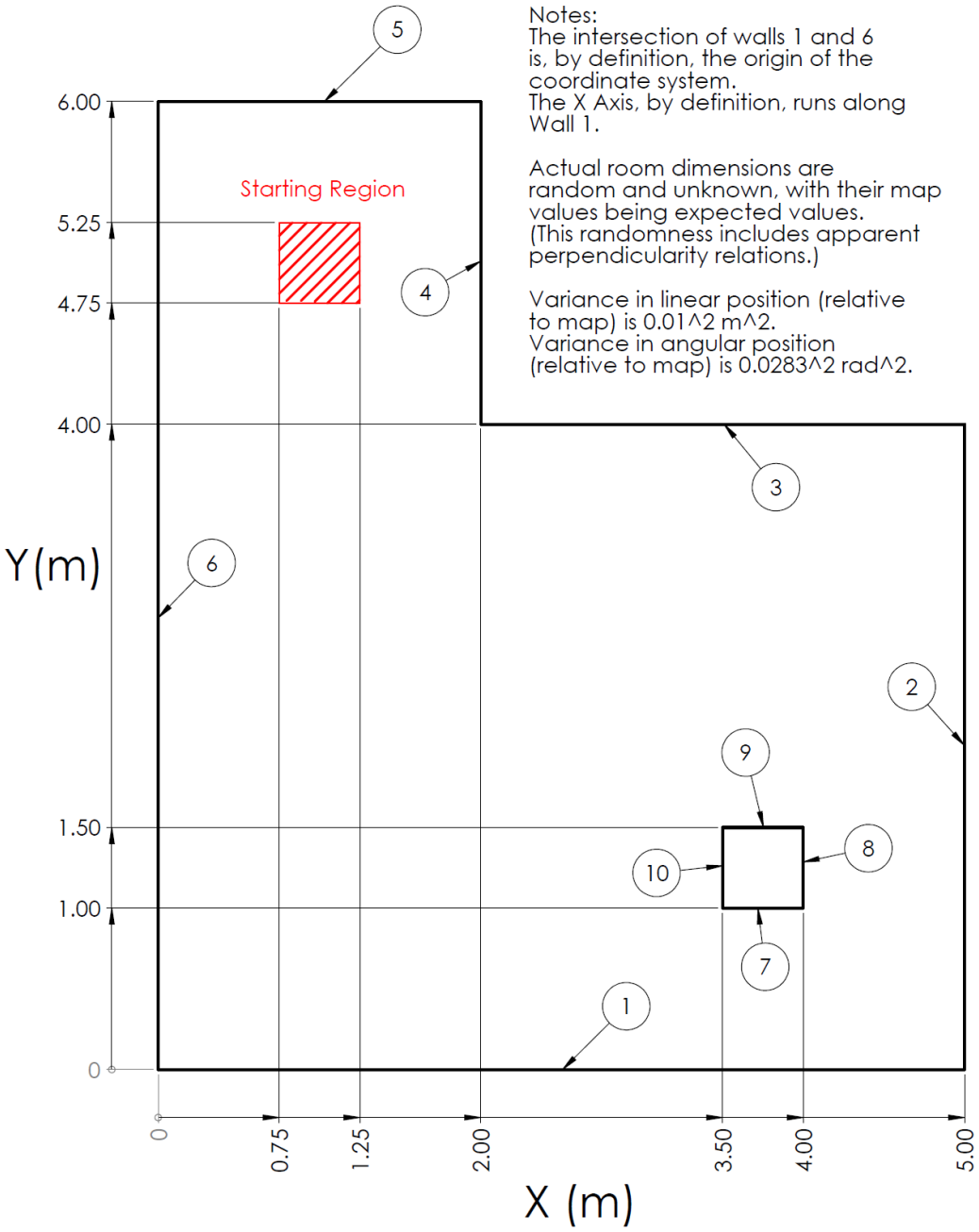

Figure 2 Map of the room traversed in this problem, with walls labeled

Now that you have an effective prediction model set up, the next step will be to integrate the structure of the room (i.e., the location of each of the 10 walls) and measurements by the LIDAR mounted on the robot (centered on point *P* in Figure 1, for simplicity), in order to correct the robot pose estimate, and to learn about the room.

Subsection 2.1: Setup and Theoretical Preliminaries

The first step in setting this up is to define an augmented state for estimation; as per the SLAM Lecture Slides, it consists of the robot pose, augmented with the distance parameter r_{map}^{j} and angle parameter α_{map}^{j} for the polar line representations of each wall $j \in \{1, \dots, 10\}$, as defined in the stationary reference frame of the map.

d) Based on the map in Figure 2, **generate an initial a priori estimate and initial covariance for the entire augmented state.** Assume that all elements of the initial a priori augmented state estimate are independent.
Hint 1: You've already calculated the robot pose section of the initial a priori augmented state estimate and covariance. For the wall parameters section, use the dimensions in the map in Figure 2 to generate the state estimate. Refer to the Notes for information on variance.
Hint 2: Consider how the map coordinate system is defined in terms of the walls. This may result in certain wall parameters having zero variance relative to the map coordinate system.

e) Given a general pose of the robot $(x_R, y_R, \theta_R)^T$, and given the map-reference-frame parameters r_{map}^{j} and α_{map}^{j} of a wall j, **calculate the robot-reference-frame parameters $\boldsymbol{r_R^j}$ and $\boldsymbol{\alpha_R^j}$ of the same wall $\boldsymbol{j}$.**
This function, $\left(r_R^j, \alpha_R^j\right)^T = h_{wall}\left(x_R, y_R, \theta_R, r_{map}^{j}, \alpha_{map}^{j}\right)$, is the function that forms the basis for the measurement function h of the Extended Kalman Filter.
Hint: Refer to Figure 1. The map-reference-frame is represented by $\{X_I, Y_I\}$, and the robot-reference-frame is represented by $\{X_R, Y_R\}$. (x_R, y_R) is the position of origin of the $\{X_R, Y_R\}$ reference frame, in terms of the $\{X_I, Y_I\}$ reference frame.

f) For the Extended Kalman Filter update step of SLAM, the measurement vector y_t is the vector of the robot-reference-frame parameters r_R^j, α_R^j of all the walls j that the LIDAR can see at time t. (These r_R^j, α_R^j are, in turn, estimated by fitting lines to the partitioned LIDAR data.)
For this question, assume that, at a given time t, three different walls, numbered i, j, k, are visible to the LIDAR. **Write the measurement function $\boldsymbol{h_t}$ such that $\boldsymbol{y_t = h_t(x_t)}$,** where x_t is the full augmented state at time t. **Next, calculate the Jacobian matrix $\boldsymbol{H_t := \frac{\partial h_t}{\partial x_t}}$.** (Note: This is the same H_t matrix used for updating the covariance and calculating Kalman gain in the Extended Kalman Filter.)

Subsection 2.2: LIDAR Processing and Filtering

Download and examine `scan_dist.csv`, `scan_partition.csv`, and `scan_partition_labels.csv`. Their contents are described in detail in `README.txt`.

Raw LIDAR data (i.e., the contents of `scan_dist.csv`) simply contain the measured distances of points along the LIDAR's angle sweep, without any metadata on what points correspond to what features in the environment. Normally when creating a SLAM algorithm, you would need to implement an algorithm to partition the raw LIDAR data into distinct features (or walls, in this example). A popular line-partitioning algorithm utilizes the *Hough transform* to extract features from an image. This is, however, outside of the scope of 2.160, so we've provided a partitioning for you to use, found in `scan_partition.csv`.

g) **Create a function in MATLAB which performs the following task:**
Given a partition of N LIDAR data corresponding to a single wall, fit robot-reference-frame line parameters r_R, α_R to that wall partition. Calculate the covariance matrix R_{wall} that corresponds to the vector of these two fit parameters.
Hint: You can use the MATLAB function lsqcurvefit to perform this fit. The covariance matrix R_{wall} can be calculated as follows, in terms of the output names of lsqcurvefit:

$$R_{wall} = (jacobian^T jacobian)^{-1}(residual^T residual)/(N-2)$$

h) **Apply the 2g function to the complete partitioned LIDAR sweep at $t = 3.5\ sec$ from the data, plotting the partitioned sweep data and their fit lines. Discuss how the R_{wall} covariances differ for the different walls plotted.**

i) From the partitioning, we know which points belong to different walls within a single scan. That said, we don't necessarily know which walls in the room these walls actually correspond to.
For extra credit, implement an algorithm that automatically assigns each wall partition in a scan to a "true" wall label, corresponding to the wall labels in the map in Figure 2. This algorithm is allowed to take as inputs the estimated full augmented state x_t, as well as the LIDAR data, and any post-processed output of the LIDAR data generated by functions you have created. You are NOT allowed to use `scan_partition_labels.csv`. *Hint: See the Lecture Slides.*
For regular credit, use `scan_partition_labels.csv` to assign wall partitions to their corresponding true wall labels.

j) **Create a function in MATLAB which performs the following task:**
At time t, given that we have a full augmented state estimate $\hat{x}_t$, we know which walls are observed by the LIDAR at time t, and we have access to each observed wall's LIDAR-estimated r_R, α_R, R_{wall}, construct the following things, as defined in 2f:

- The measurement, y_t
- The covariance matrix of the measurement, R_t
- The predicted measurement, $\hat{y}_t = h_t(\hat{x}_t)$
- The measurement Jacobian H_t

This function is partly a generalization of the function calculated in 2f.

k) **Implement the complete Extended Kalman Filter for performing SLAM.**
Something to note is that the LIDAR scan measurement frequency is much lower than the odometry measurement frequency. Consequently, you will not be able to apply the EKF update from LIDAR every time step. This is not a problem, as the modular structure of discrete Kalman-like filters allows this.

Subsection 2.3: Commentary

l) **Plot the SLAM-filtered estimated path of the robot, in the context of the final estimated walls of the room. How does this SLAM-filtered path compare to the path predicted by odometry alone?**

m) **Show how the covariance of the robot pose estimate changed over the course of the SLAM run. What events seemed to most greatly affect the robot pose covariance? Why do you think this was?**

n) **Extra credit: Examine how the covariance of the wall parameters changed over the course of the SLAM run. What events seemed to most greatly affect the wall parameter covariance? Why do you think this was?**

Bonus question (no credit, optional): **Do you have any recommendations regarding how this context-oriented project could be improved in future iterations?**