Sign-Informed Semantic Mapping for Language Interaction

Benned Hedegaard
Computer Science Department
University of Rochester
Rochester, New York
bhedegaa@u.rochester.edu

***Abstract*—Natural language provides an expressive means for human-robot interaction. However, for a robot to communicate about an indoor space, its map of the world must include language-based information about that space. Even without live human interaction, wall signs can act as a rich source of written language for anyone able to read them. In this work, we propose a robot system to read and integrate such language into a representation suitable for human-robot language interaction. We formalize the proposed architecture, describe our real-world evaluation setup, and specify experiments to evaluate our approach. As this work remains ongoing, we conclude with discussion of the challenges in reaching this stage of our implementation.**

I. Introduction

Consider what is required for an intelligent agent to navigate through a building to a particular room. Although human intelligence may obfuscate the complexity of this task, many issues must be solved at the same time. Our agent must *localize* against their surroundings to ensure that their motions are moving them in the right direction. Imagine someone trying to walk along a hallway with their eyes closed. After walking sufficiently far, they will inevitably veer into a wall. Furthermore, an agent must *map* the world around them: where are the walls, where is the sign confirming their destination, and, if they spot it, to which room does the sign refer?

Autonomous robots must achieve all of these tasks using only the information available from noisy actuation and sensing. Due to the complexity and ubiquity of this task, a long line of robotics research has been dedicated to solving this *simultaneous localization and mapping* (SLAM) problem. The SLAM problem arises whenever a robot at an unknown location in an unknown environment must incrementally build an estimate of its pose and a consistent map of the world [5].

There exists a rich body of literature characterizing the SLAM problem and its solutions. One of the earliest solutions, due to Smith and Cheeseman (1986), is based on the extended Kalman filter (EKF) [14]. Their method, called EKF SLAM, represents the posterior using a multidimensional Gaussian distribution. EKF SLAM enjoyed great popularity throughout the 1990s and early 2000s due to its simplicity and practical utility in simple environments. However, the approach is brittle to incorrect associations between robot sensor observations and landmarks in the world [16]. EKF SLAM's quadratic complexity prevents it from scaling to maps larger than about 1,000 landmarks, making it inappropriate for mapping large-scale or complex environments.

However, EKF SLAM is a compelling choice for a text-based indoor mapping system. The wall signs in a building are relatively few, tend to be spatially distant, and necessarily contain unique text describing the corresponding room. These features mitigate the weaknesses of EKF SLAM by preventing a single map from growing too large. The distance between wall signs, as well as their distinct text, provides a fairly unambiguous signal for observation-landmark associations.

Based on this idea, we present a semantic mapping system that detects wall sign landmarks for use in EKF SLAM. Such a mapping pipeline is a prerequisite for our underlying objective: to enable natural language grounding for the names and spatial relations of rooms. Our current contributions include 1) a full presentation of the EKF SLAM algorithm applied to our task, 2) methods for the data association of text-based landmarks, and 3) proposed experiments demonstrating the semantic mapping and language interaction capabilities of the system. Because this work is ongoing, this paper focuses on our algorithmic formalization of this task.

The rest of this paper is organized as follows. In Section II, we discuss prior work on SLAM, the extended Kalman filter, and semantic SLAM. In Section III, we describe our proposed system, beginning with an overview (Section III-A). We then describe our EKF SLAM algorithm in Section III-B, including data association, map management, and landmark initialization procedures for text-based landmarks. In Section IV, we describe our evaluation setup and planned experiments. We conclude in Section V by discussing our challenges during the implementation of the proposed system.

II. Prior Work

In this section, we discuss the classic probabilistic SLAM formulation, the extended Kalman filter, and more recent semantic mapping approaches.

A. Structure of the SLAM Problem

The probabilistic simultaneous localization and mapping (SLAM) problem was first discussed at the 1986 IEEE Robotics and Automation Conference [5]. Since then, SLAM has been a cornerstone of the robotics research community. By the early 2000s, a general probabilistic framework for

SLAM had been established and multiple resources were quickly published on the topic [5, 1, 16]. For consistency, we will follow the notation used in the *Probabilistic Robotics* textbook by Thrun, Burgard, and Fox [16]. As an additional resource, the two-part tutorial by Durrant-Whyte and Bailey (2006) provides historical context as well as a summary of the challenges faced by the field at the time [5, 1].

During SLAM, a mobile robot builds a map of its environment and simultaneously estimates its location within that map. Because many SLAM approaches use discrete update steps, let us denote the passage of time $t = 1, 2, \dots$ in discrete steps. At time t, we denote the robot state as x_t, the current robot motion command as $u_t = (v_x, \omega_z)$, and the current robot observation (i.e. sensor measurement) as z_t. As for mapping, typical SLAM approaches model the world as a set of simple static landmarks $m = \{m_1, m_2, \dots\}$.

The *full SLAM problem* seeks to compute the joint posterior $p(x_{1:t}, m|u_{1:t}, z_{1:t})$ of x_t and m for all times t, given all motion commands $u_{1:t}$ and observations $z_{1:t}$. However, for live language interaction, we are more interested in the *online SLAM problem*, where posterior $p(x_t, m|z_{1:t}, u_{1:t})$ is only over the momentary robot pose and map. A recursive solution to online SLAM takes the form $p(x_t, m|\tilde{x}_{t-1}, u_t, z_t)$, where $\tilde{x}_{t-1}$ is a distribution over the previous robot pose. Breaking this down further, two models are used to represent the effects of motion commands and observations. First, a *motion model* $g(x_{t-1}, u_t)$ defines a probability distribution $p(x_t|x_{t-1}, u_t)$ on state transitions. Second, a *measurement model* $h(x_t, m)$ describes the probability of making an observation z_t as $p(z_t|x_t, m)$. Using these models, a SLAM algorithm is typically implemented in two steps: a *motion update* integrates u_t and then a *measurement update* integrates z_t. In this way, the *belief state*, or the robot's current probabilistic model of the world, can be recursively updated each time step [16].

B. The Extended Kalman Filter

The extended Kalman filter (EKF) implements this recursive two-step process for a Gaussian system. Unlike the Kalman filter, which assumes that $g(x_{t-1}, u_t)$ and $h(x_t, m)$ are linear functions with added Gaussian noise, the EKF allows general nonlinear models with zero-mean Gaussian noise to be used:

$$x_t = g(x_{t-1}, u_t) + \epsilon_t \tag{1}$$

$$z_t = h(x_t) + \delta_t \tag{2}$$

Here $\epsilon_t \sim \mathcal{N}(0, R_t)$ and $\delta_t \sim \mathcal{N}(0, Q_t)$. The EKF represents its belief state as a multidimensional Gaussian composed of mean vector μ_t and covariance matrix Σ_t. In order to filter using nonlinear models, the EKF performs *linearization*, where the Taylor approximation at the current mean is used:

$$g(x_{t-1}, u_t) \approx g(\mu_{t-1}, u_t) + G_t(x_{t-1} - \mu_{t-1}) \tag{3}$$

$$h(x_t) \approx h(\bar{\mu}_t) + H_t(x_t - \bar{\mu}_t), \tag{4}$$

where Jacobians $G_t = \frac{\partial g(\mu_{t-1}, u_t)}{\partial x_{t-1}}$ and $H_t = \frac{\partial h(\bar{\mu}_t)}{\partial \bar{\mu}_t}$. To provide context for our description of the full EKF SLAM algorithm, we reproduce the general extended Kalman filter in Algorithm 1. This presentation has been modified from the *Probabilistic Robotics* textbook so that S_t is explicit [16].

In the algorithm, lines 2 and 3 implement the motion update, producing the predicted belief $\bar{\mu}_t$ and $\bar{\Sigma}_t$. Line 4 then computes the covariance S_t of the predicted observation $\hat{z}_t = h(\bar{\mu}_t)$. Then, line 5 computes the *Kalman gain* K_t, which controls how the observation z_t is incorporated into the new state. Within line 6, we compute $z_t - h(\bar{\mu}_t)$, the *innovation*, or the difference between the actual observation z_t and the expected observation $\hat{z}_t$. In lines 6-8, the new mean μ_t and covariance Σ_t of the posterior belief are calculated and returned.

Algorithm 1: EXTENDED KALMAN FILTER

Input: Mean vector μ_{t-1}, covariance matrix Σ_{t-1}, motion command u_t, observations z_t
Output: Updated mean μ_t and covariance Σ_t

1 **Function** EKF(μ_{t-1}, Σ_{t-1}, u_t, z_t)**:**
2 $\quad \bar{\mu}_t = g(\mu_{t-1}, u_t)$
3 $\quad \bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$
4 $\quad S_t = H_t \bar{\Sigma}_t H_t^T + Q_t$
5 $\quad K_t = \bar{\Sigma}_t H_t^T S_t^{-1}$
6 $\quad \mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
7 $\quad \Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$
8 $\quad$ **return** μ_t, Σ_t

C. Modern Semantic SLAM

Although EKF SLAM was quite successful for its time, recent approaches differ from EKF SLAM in a few notable ways. First, modern approaches to SLAM often combine various forms of metric, topological, and semantic information in order to build more expressive and metrically accurate world representations [8, 13]. Such systems frequently integrate multiple streams of information to create their rich world models. For example, Walter et al. (2013) enable a robot to integrate language information into the semantic layer of their world model, resulting in improved metric accuracy [18].

The *pose graph* has proven to be a useful structure for many of these SLAM solutions. Such a graph is composed of a set of robot poses connected by odometry edges [18]. Additional constraints (e.g. landmark observations or loop closures) can be added between poses to increase metric accuracy. Then, an optimization approach such as iSAM2 efficiently relaxes the entire graph toward a more accurate estimate [9]. By storing the entire robot trajectory, these approaches retain valuable information for loop closure, potentially improving accuracy by solving the full SLAM problem even for online tasks.

Recent work in SLAM has also increasingly leveraged vision as a means for accurate odometry. By identifying viewpoint-independent landmarks via object recognition, metric accuracy can be improved during loop closure [2]. Crucially, these contributions demonstrate that the addition of semantic information (e.g. object types or language annotations) into a SLAM system not only benefits the support of robot

understanding, but can even improve the accuracy of other components of the map (e.g. metric or topological).

In light of these trends, our system merges semantic information directly into its metric component. By choosing wall signs as our landmarks, we ensure that our system can access many pieces of relevant information for natural language interaction regarding rooms: signs can provide room numbers, the names of office occupants, and a spatially grounded object indicating the location of the room.

III. Technical Approach

In this section, we begin with a sketch of the proposed semantic mapping architecture. Next, to explore the full system in depth, we describe our EKF SLAM approach, including data association, map management, and landmark initialization for text-based landmarks. Finally, we briefly discuss the language grounding and computer vision modules of the system, both of which are imported from external sources.

A. System Overview

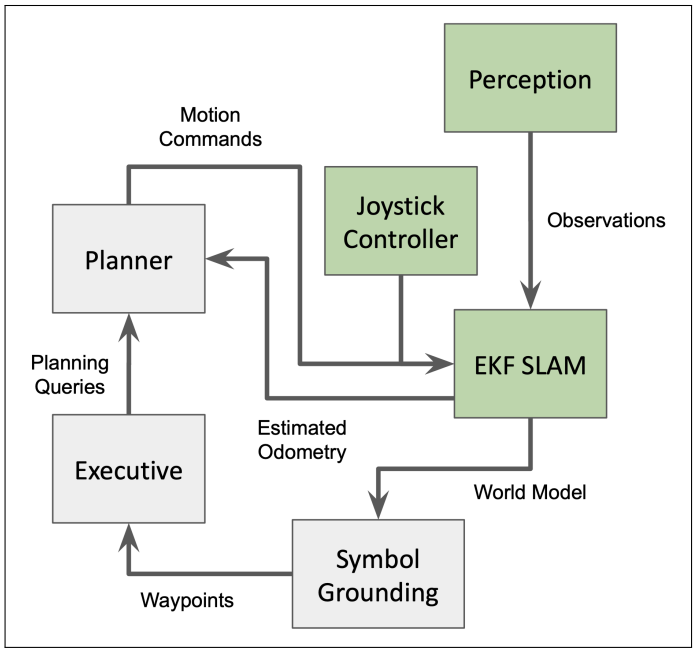

Fig. 1. Illustration of all modules in the proposed system. Modules colored in green are new to the architecture, which was originally developed for the ECE 232: Autonomous Mobile Robots course. Joystick control will be used for data collection, while autonomous planning will be used for language interaction experiments.

B. Applying EKF SLAM to Text-Based Landmarks

Here, we provide a coherent treatment of EKF SLAM which unifies disparate sources into a single notational convention. The basis for our approach is the treatment presented in the *Probabilistic Robotics* textbook [16]. However, our description improves upon theirs in three ways:

1) We introduce explicit submatrix notation to clarify the exact computation required by EKF SLAM. This allows us to remove sparse matrices otherwise required to extend Jacobians to the full dimension of μ_t and Σ_t.
2) We integrate pseudocode for a map management procedure based on the landmark candidate list described by Dissanayake et al. (2001) [4].
3) We include an explicit landmark intialization procedure, due to Dissanayake et al. (2001). This approach has been used in a number of works [4, 11, 19] under various notations. In particular, we find the presentation due to Joan Solà to be the most clear [15].

We adapt these various descriptions in order to provide a consistent notation for our approach. The core of our algorithm is the state vector $\mu_t = [\mu_t^R \ \mu_t^M]^T$. If the system currently tracks N confirmed landmarks, then:

$$\mu_t^R = [x, y, \theta] \tag{5}$$

$$\mu_t^M = [m_{1,x}, m_{1,y}, \ldots, m_{N,x}, m_{N,y}] \tag{6}$$

Here μ_t is $(2N+3) \times 1$ and Σ_t is $(2N+3) \times (2N+3)$. To simplify later computation, we define Σ_t using submatrices:

$$\Sigma_t = \begin{bmatrix} \Sigma_t^{RR} & \Sigma_t^{RM} \\ \left[\Sigma_t^{RM}\right]^T & \Sigma_t^{MM} \end{bmatrix} \tag{7}$$

where Σ_t^{RR} is the 3×3 covariance for μ_t^R, Σ_t^{RM} is the $3 \times 2N$ covariance between the robot pose and the map, and Σ_t^{MM} is the $2N \times 2N$ covariance of the map's landmark locations. Let us write Σ_t^k for the 5×5 covariance matrix between the robot pose estimate μ_t^R and a single landmark m_k.

C. Motion Update

The first step of EKF SLAM consists of a motion update, where the motion command u_t is simulated forward in time by duration Δ_t. For this simulation, we use the motion model:

$$g(\mu_{t-1}^R, u_t) = \mu_{t-1}^R + \begin{bmatrix} -r_t \sin(\theta) + r_t \sin(\theta') \\ r_t \cos(\theta) - r_t \cos(\theta') \\ \omega_t \Delta_t \end{bmatrix} \tag{8}$$

where $\theta' = \theta + \omega_t \Delta_t$ and $r_t = \frac{v_t}{\omega_t}$ is the radius of the current path of motion. Given this motion model, we can compute the Jacobian G_t, the derivative of g w.r.t. x_{t-1} evaluated at μ_{t-1} and u_t. We find that G_t is given by:

$$G_t = \begin{bmatrix} 1 & 0 & r_t(-\cos(\theta) + \cos(\theta')) \\ 0 & 1 & r_t(-\sin(\theta) + \sin(\theta')) \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

$$M_t = \begin{bmatrix} \alpha_1 v_t^2 + \alpha_2 \omega_t^2 & 0 \\ 0 & \alpha_3 v_t^2 + \alpha_4 \omega_t^2 \end{bmatrix} \tag{10}$$

As a reminder, the Jacobian J of a function $f(x_1, \ldots, x_n) : \mathbb{R}^n \to \mathbb{R}^m$ is $m \times n$, with entry $J_{r,c}$ equal to $\frac{\partial f_r}{\partial x_c}$. Based on motion command noise parameters $\alpha_1 \ldots \alpha_4$, we use matrix M_t to model noise in the control space (see 10). Finally, we

map this noise into the state space using V_t, the 3×2 Jacobian of g w.r.t. u_t evaluated at μ_{t-1} and u_t:

$$V_t = \frac{\partial g(\mu^R_{t-1}, u_t)}{\partial u_t} \tag{11}$$

$$= \begin{bmatrix} \frac{-\sin(\theta)+\sin(\theta')}{\omega_t} & r_t \cos(\theta')\Delta_t + \frac{v_t(\sin(\theta)-\sin(\theta'))}{\omega_t^2} \\ \frac{\cos(\theta)-\cos(\theta')}{\omega_t} & r_t \sin(\theta')\Delta_t + \frac{v_t(\cos(\theta')-\cos(\theta))}{\omega_t^2} \\ 0 & \Delta_t \end{bmatrix} \tag{12}$$

These pieces allow us to specify the full motion update in Algorithm 2, where first the motion model is applied to the robot pose estimate (line 1). In line 2, V_t is used to map the control space noise into an approximation of the motion covariance R_t in state space. Finally, the updated covariance is computed using G_t and R_t (line 4).

Algorithm 2: MOTION UPDATE

Input: Robot pose estimate μ^R_{t-1}, robot pose covariance Σ^R_{t-1}, motion command u_t
Output: Motion-updated mean $\bar{\mu}_t$ and covariance $\bar{\Sigma}_t$

1 **Function** MOTIONUPDATE(μ^R_{t-1}, Σ^R_{t-1}, u_t)**:**
2 $\quad \bar{\mu}^R_t = g(\mu^R_{t-1}, u_t)$
3 $\quad R_t = V_t M_t V_t^T$
4 $\quad \bar{\Sigma}^{RR}_t = G_t \Sigma^{RR}_{t-1} G_t^T + R_t$
5 $\quad$ **return** $\bar{\mu}^R_t$, $\bar{\Sigma}^R_t$

D. Measurement Update

After motion error has been added to the belief state, observations of landmarks are used to reduce uncertainty during the measurement update. For our implementation, we model each landmark as $m_k = (x_k, y_k, s_k)$, where $(x_k, y_k) \in \mathbb{R}^2$ and s_k is a string signature stored separately from μ_t. Because a measurement z_t might sense multiple landmarks at once, individual observations z^i_t are assumed to be independent:

$$p(z_t|x_t, m) = \prod_i p(z^i_t|x_t, m) \tag{13}$$

An individual observation $z^i_t = (r^i_t, \phi^i_t, s^i_t)$ consists of a range r^i_t between the sensor and the detected landmark, a bearing ϕ^i_t in the robot frame, and an identifying signature s^i_t. This so-called range-bearing feature model originated in Leonard and Durrant-Whyte (1991) [10]. The measurement model for generating such an observation is given by:

$$h(\bar{\mu}_t) = \hat{z}^i_t = \begin{bmatrix} \sqrt{q_k} \\ \text{atan2}(\delta_{k,y}, \delta_{k,x}) - \bar{\mu}_{t,\theta} \end{bmatrix} \tag{14}$$

where $\delta_{k,x} = \bar{\mu}_{k,x} - \bar{\mu}_{t,x}$, $\delta_{k,y} = \bar{\mu}_{k,y} - \bar{\mu}_{t,y}$, and $q_k = \delta^2_{k,x} + \delta^2_{k,y}$ are defined to simplify later notation. To permit the use of $\hat{z}^i_t$ as a vector, we will assume its signature $s^i_t = s_k$.

Let us apply this model in an EKF SLAM measurement update. To do so, assume we have determined the most likely landmark, say m_k, that was detected in order to produce observation z^i_t. We proceed by computing the Jacobian H^k_t of $h(\bar{\mu}_t)$ w.r.t. the robot pose x_t and landmark m_k:

$$H^k_t = \begin{bmatrix} \left[\frac{\partial h(\bar{\mu}_t)}{\partial x_t}\right]_{2\times 3} & \left[\frac{\partial h(\bar{\mu}_t)}{\partial m_k}\right]_{2\times 2} \end{bmatrix} \tag{15}$$

$$= \left[\begin{array}{ccc|cc} \frac{-\delta_{k,x}}{\sqrt{q_k}} & \frac{-\delta_{k,y}}{\sqrt{q_k}} & 0 & \frac{\delta_{k,x}}{\sqrt{q_k}} & \frac{\delta_{k,y}}{\sqrt{q_k}} \\ \frac{\delta_{k,y}}{q_k} & \frac{-\delta_{k,x}}{q_k} & -1 & \frac{-\delta_{k,y}}{q_k} & \frac{\delta_{k,x}}{q_k} \end{array}\right] \tag{16}$$

Using H^k_t and the 2×2 observation covariance $Q_t = \begin{bmatrix} \sigma^2_r & 0 \\ 0 & \sigma^2_\phi \end{bmatrix}$, we can compute a full state measurement update as described in Algorithm 3.

Algorithm 3: FULL STATE MEASUREMENT UPDATE

Input: Motion-updated mean $\bar{\mu}_t$ and covariance $\bar{\Sigma}_t$, observation z^i_t, landmark m_k
Output: New mean vector $\bar{\mu}_t$ with covariance $\bar{\Sigma}_t$

1 **Function** MEASUREMENTUPDATE($\bar{\mu}_t$, $\bar{\Sigma}_t$, z^i_t, m_k)**:**
2 $\quad \Psi_k = H^k_t \bar{\Sigma}^k_t [H^k_t]^T + Q_t$
3 $\quad \bar{\Sigma}^K_t = \begin{bmatrix} \bar{\Sigma}^{RR}_t & \bar{\Sigma}^{Rk}_t \\ \bar{\Sigma}^{1R}_t & \bar{\Sigma}^{1k}_t \\ \vdots & \vdots \\ \bar{\Sigma}^{NR}_t & \bar{\Sigma}^{Nk}_t \end{bmatrix}$
4 $\quad K^i_t = \bar{\Sigma}^K_t [H^k_t]^T \Psi^{-1}_k$
5 $\quad \hat{z}^k_t = h(\bar{\mu}_t)$
6 $\quad \mu_t = \bar{\mu}_t + K^i_t (z^i_t - \hat{z}^k_t)$
7 $\quad \bar{\Sigma}_t = (I - K^i_t H^k_t) \bar{\Sigma}_t$
8 $\quad$ **return** $\bar{\mu}_t$, $\bar{\Sigma}_t$

E. Data Association

Note that we have (so far) taken for granted the *data association* problem, where a SLAM system must identify the most likely correspondence $c(i)$ between z^i_t and some landmark $m_{c(i)}$ in the map. To identify the maximum-likelihood (ML) correspondence, we use two modules: a position-based probabilistic distance and a string-based filter on signatures.

For our position-based ML estimate, we use the approach presented in Dissanayake et al. (2001) [4]. To formalize this approach, consider a *landmark model* which provides the expected landmark $\hat{m}^t_i$ given the robot state estimate $\bar{\mu}_t$ and observation z^i_t:

$$f(\bar{\mu}_t, z^i_t) = \hat{m}^i_t = \begin{bmatrix} \bar{\mu}_{t,x} + r^i_t \cos(\phi^i_t + \bar{\mu}_{t,\theta}) \\ \bar{\mu}_{t,y} + r^i_t \sin(\phi^i_t + \bar{\mu}_{t,\theta}) \end{bmatrix} \tag{17}$$

We introduce this notation in order to take the Jacobian of $f(\bar{\mu}_t, z^i_t)$. Let us denote the Jacobian of f w.r.t. $\bar{\mu}^R_t$ as $F^{i,R}_t$

and the Jacobian of f w.r.t. z_t^i as $F_t^{i,z}$, each given by:

$$F_t^{i,R} = \frac{\partial f(\bar{\mu}_t, z_t^i)}{\partial \bar{\mu}_t^R} \tag{18}$$

$$= \begin{bmatrix} 1 & 0 & -r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \\ 0 & 1 & r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \end{bmatrix} \tag{19}$$

$$F_t^{i,z} = \frac{\partial f(\bar{\mu}_t, z_t^i)}{\partial z_t^i} \tag{20}$$

$$= \begin{bmatrix} \cos(\phi_t^i + \bar{\mu}_{t,\theta}) & -r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \\ \sin(\phi_t^i + \bar{\mu}_{t,\theta}) & r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \end{bmatrix} \tag{21}$$

We will use $F_t^{i,R}$ to convert the robot pose uncertainty Σ_t^{RR} into the landmark space, while $F_t^{i,z}$ allows us to convert the measurement uncertainty Q_t into the landmark space.

For the second part of our data association method, we utilize the observation text signature s_t^i. Following previous work in autonomous sign reading, we propose using the Needleman-Wunsch algorithm for sequence alignment [12]. By finding the maximum alignment a_k^i for each signature pair s_k and s_t^i, we can filter out landmarks with uninformative or non-matching signatures. We choose the default value for this alignment threshold $a_{min} = 4$ based on Case et al. (2011), who suggest that alignments less than four are unreliable from OCR-obtained wall sign text [3]. However, we plan to vary this hyperparameter in our own experiments.

Our full data association method is detailed in Algorithm 4. In lines 2 and 3, we compute the position and covariance of the expected landmark based on $\bar{\mu}_t^R$ and z_t^i. Then, for each landmark m_k, we reject correspondences with maximum sequence alignments shorter than a_{min} (lines 4-7). In line 8, we compute ρ_k as a scaling of a_k^i based on w_s, a weighting parameter that allows us to weight the importance of the signature compared to the position. Finally, we compute the Mahalanobis distance π_k between $\hat{m}_t^i$ and m_k (line 9). The algorithm concludes by returning $c(i)$, the minimum of $\frac{\pi_k}{\rho_k}$ over all non-rejected landmarks (lines 10 and 11).

Algorithm 4: TEXT-BASED DATA ASSOCIATION

Input: Motion-updated mean $\bar{\mu}_t$ and covariance $\bar{\Sigma}_t$, observation z_t^i, signature weight w_s
Output: Maximum-likelihood correspondence $c(i)$

1 **Function** CORRESPONDENCE($\bar{\mu}_t$, $\bar{\Sigma}_t$, z_t^i)**:**
2 $\quad \hat{m}_t^i = f(\bar{\mu}_t, z_t^i)$
3 $\quad \hat{\Sigma}_t^i = F_t^{i,R} \bar{\Sigma}_t^{RR} [F_t^{i,R}]^T + F_t^{i,z} Q_t [F_t^{i,z}]^T$
4 $\quad$ **for** $k \in [1, \ldots, N]$ **do**
5 $\quad\quad a_k^i = \text{MAXIMUMALIGNMENT}(s_k, s_t^i)$
6 $\quad\quad$ **if** $a_k^i < a_{min}$ **then**
7 $\quad\quad\quad$ **continue**
8 $\quad\quad \rho_k = (a_k^i)^{w_s}$
9 $\quad\quad \pi_k = (\hat{m}_t^i - m_k)^T (\hat{\Sigma}_t^i + \bar{\Sigma}_t^{kk})^{-1} (\hat{m}_t^i - m_k)$
10 $\quad c(i) = \underset{k}{\text{argmin}} \frac{\pi_k}{\rho_k}$
11 $\quad$ **return** $c(i)$

F. Map Management

We have yet to specify what occurs when the signature alignment of all landmarks is shorter than a_{min}. In this case, we turn to a list of M *candidate landmarks*, denoted $\{m_{N+1,x}, m_{N+1,y}, \ldots, m_{N+M,x}, m_{N+M,y}\}$, that have not yet been confirmed into the full map [4]. Only after a landmark has been observed at least c_{min} times, shrinking its uncertainty, is it added to the full map. Keeping this provisional landmark list allows the EKF SLAM system to deal with spurious landmark sightings. This method also significantly reduces the number of landmarks in the map, while retaining all physical landmarks with high probability [16].

G. Landmark Initialization

Suppose an observation corresponds to neither a confirmed nor candidate landmark. Whenever we see a new sign, how do we initialize its representation in the candidate landmark list? We have already seen how to compute the expected landmark's position $\hat{m}_t^i$ and covariance $\hat{\Sigma}_t^i$ based on $\bar{\mu}_t$ and z_t^i. Although it is sufficient to track these for each candidate landmark, a different measurement update procedure is needed for the smaller representation. We provide this procedure in Algorithm 5. Its steps are analogous to Algorithm 3 over smaller matrices.

Algorithm 5: CANDIDATE LANDMARK UPDATE

Input: Motion-updated mean $\bar{\mu}_t$ and covariance $\bar{\Sigma}_t$, observation z_t^i, candidate landmark $(\hat{m}_k, \Sigma_t^{kk})$
Output: Updated candidate landmark $(\hat{m}_k, \Sigma_t^{kk})$

1 **Function** CANDIDATEUPDATE($\bar{\mu}_t$, $\bar{\Sigma}_t$, z_t^i, $\hat{m}_k$, Σ_t^{kk})**:**
2 $\quad \bar{\Sigma}_t^{kR} = F_t^{i,R} \bar{\Sigma}_t^{RR}$
3 $\quad \bar{\Sigma}_t^{k} = \begin{bmatrix} \bar{\Sigma}_t^{RR} & [\bar{\Sigma}_t^{kR}]^T \\ \bar{\Sigma}_t^{kR} & \Sigma_t^{kk} \end{bmatrix}_{5\times 5}$
4 $\quad \Psi_k = H_t^k \bar{\Sigma}_t^k [H_t^k]^T + Q_t$
5 $\quad K_t^i = \bar{\Sigma}_t^{kk} [H_t^k]^T \Psi_k^{-1}$
6 $\quad \hat{z}_t^k = h(\bar{\mu}_t)$
7 $\quad \hat{m}_k = \hat{m}_k + K_t^i (z_t^i - \hat{z}_t^k)$
8 $\quad \Sigma_t^{kk} = (I - K_t^i H_t^k) \Sigma_t^{kk}$
9 $\quad$ **return** $(\hat{m}_k, \Sigma_t^{kk})$

IV. PROPOSED EXPERIMENTS

In this section, we describe our planned evaluation setup for real-world evaluation of the proposed semantic mapping pipeline.

A. Robot Platform

For our physical experiments, we will use the TurtleBot2 with a Kobuki base. This open-source platform provides convenient Robot Operating System (ROS) wrappers to convert (v_x, ω_z) motion commands into robot-frame motion. Our platform is equipped with an Asus Xtion Pro Live camera which provides an RGB-D image stream with 480p at 30 FPS. From this stream, optical character recognition will be performed using the EasyOCR package [7]. This package was chosen due to its ease-of-use and fast CPU performance.

B. Sensor Model

We will convert EasyOCR text detections into EKF SLAM observations using the following process. Given an image, EasyOCR provides a list of text detections, each with a bounding box, string of detected text, and a confidence level. For each detection in this list, we first use the four bounding box corners to compute an average depth of the detection. Then, the center point of the detection is transformed into the robot frame to provide the range r_t^i and bearing ϕ_t^i for an observation.

A potential challenge for the system will be the merging of nearby text boxes, as text clustering across one or more signs may be inconsistent across observations. A naive solution to this is to treat each word as its own landmark, or generalize our landmark signatures to sets of string tags. A second challenge with this approach may be additional text in the environment from posters or directory signage. Although this information may provide useful landmarks for EKF SLAM, it could be automatically filtered out by ignoring images with more than d_{max} text detections. We may also experiment with discarding detections below some confidence threshold.

C. Language Interaction

We plan to use the Human to Structured Language (H2SL) library for *language grounding*, or connecting a language command to the robot's world model [17]. H2SL provides functionality for real-time natural language symbol grounding using the Distributed Correspondence Graph (DCG) model [6]. The DCG approach leverages the hierarchical structure of language to efficiently search over sets of potential groundings.

D. Proposed Experiments

We propose two experiments to evaluate and demonstrate the proposed semantic mapping pipeline. These experiments address the hyperparameter tuning and human-robot interaction aspects of the system.

1) *Experiment 1* - Using manual joystick control, the Turtlebot2 is driven around a floor of an academic building on the University of Rochester. The floor will be selected such that the path includes at least one loop. This will result in a dataset on which variations of the EKF SLAM pipeline can be tested with various hyperparameters. To qualitatively evaluate the variations, the resulting map estimates will be visually superimposed onto the building's blueprint.
2) *Experiment 2* - Offline, a simple dataset of language references to rooms will be compiled using the world model resulting from Experiment 1. Then, a DCG model will be trained on these examples. We will demonstrate the system's utility for online language interaction by manually operating the system on an unseen floor of the same building, collecting a new map, and then instructing the robot to navigate to particular rooms on this new floor. Such an experiment will demonstrate the capability of the system the generalize to unseen environments to support online language interaction.

V. Conclusion

We have presented a semantic mapping pipeline based on EKF SLAM with wall signs as landmarks. Due to time constraints and conflicts, our implementation is not yet complete, but we have described our formalization of the semantic mapping problem, our proposed solution, and the experiments and physical platform that will be necessary to field this system.

$$x_t = (x, y, \theta) \tag{22}$$
$$u_t = (v_x, \omega_z) \tag{23}$$
$$z_t^i = (r_t^i, \phi_t^i, s_t^i) \tag{24}$$
$$m_k = (x_k, y_k, s_k) \tag{25}$$

$$\mu_t = [\mu_t^R \; \mu_t^M]^T \tag{26}$$
$$\mu_t^R = [x, y, \theta] \tag{27}$$
$$\mu_t^M = [m_{1,x}, m_{1,y}, \ldots, m_{N,x}, m_{N,y}] \tag{28}$$

Acknowledgments

References

[1] T. Bailey and H. Durrant-Whyte. "Simultaneous Localization and Mapping (SLAM): Part II". In: *IEEE Robotics Automation Magazine* 13.3 (2006), pp. 108–117. DOI: 10.1109/MRA.2006.1678144.

[2] Sean L. Bowman et al. "Probabilistic Data Association for Semantic SLAM". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 1722–1729. DOI: 10.1109/ICRA.2017.7989203.

[3] Carl Case et al. "Autonomous Sign Reading for Semantic Mapping". In: *2011 IEEE International Conference on Robotics and Automation* (2011), pp. 3297–3303.

[4] M.W.M.G. Dissanayake et al. "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem". In: *IEEE Transactions on Robotics and Automation* 17.3 (2001), pp. 229–241. DOI: 10.1109/70.938381.

[5] H. Durrant-Whyte and T. Bailey. "Simultaneous Localization and Mapping: Part I". In: *IEEE Robotics Automation Magazine* 13.2 (2006), pp. 99–110. DOI: 10.1109/MRA.2006.1638022.

[6] Thomas Howard, Stefanie Tellex, and Nicholas Roy. "A Natural Language Planner Interface for Mobile Manipulators". In: May 2014, pp. 6652–6659. DOI: 10.1109/ICRA.2014.6907841.

[7] JaidedAI. *EasyOCR*. https://github.com/JaidedAI/EasyOCR. 2022.

[8] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. "iSAM: Incremental Smoothing and Mapping". In: *IEEE Transactions on Robotics* 24.6 (2008), pp. 1365–1378. DOI: 10.1109/TRO.2008.2006706.

[9] Michael Kaess et al. “iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree”. In: *The International Journal of Robotics Research* 31.2 (2012), pp. 216–235. DOI: 10.1177/0278364911430419.

[10] J.J. Leonard and H.F. Durrant-Whyte. “Simultaneous Map Building and Localization for an Autonomous Mobile Robot”. In: *Proceedings IROS ’91:IEEE/RSJ International Workshop on Intelligent Robots and Systems ’91*. 1991, 1442–1447 vol.3. DOI: 10.1109/IROS.1991.174711.

[11] John J. Leonard et al. “Mapping Partially Observable Features from Multiple Uncertain Vantage Points”. In: *The International Journal of Robotics Research* 21.10-11 (2002), pp. 943–975. DOI: 10 . 1177 / 027836490202101088 9.

[12] Saul B. Needleman and Christian D. Wunsch. “A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins”. In: *Journal of Molecular Biology* 48.3 (1970), pp. 443–53.

[13] Andrzej Pronobis and Patric Jensfelt. “Large-scale Semantic Mapping and Reasoning with Heterogeneous Modalities”. In: *2012 IEEE International Conference on Robotics and Automation*. 2012, pp. 3515–3522. DOI: 10.1109/ICRA.2012.6224637.

[14] Randall C. Smith and Peter C. Cheeseman. “On the Representation and Estimation of Spatial Uncertainty”. In: *The International Journal of Robotics Research* 5 (1986), pp. 56–68.

[15] Joan Solà. *Simultaneous localization and mapping with the extended Kalman filter*. 2014.

[16] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN: 0262201623.

[17] tmhoward. *Human to Structured Language*. https : / / github.com/tmhoward/h2sl. 2014.

[18] Matthew R. Walter et al. “Learning Semantic Maps from Natural Language Descriptions”. In: *The International Journal of Robotics Research* (June 2013). DOI: 10.15607/RSS.2013.IX.004.

[19] Stefan Williams, Hugh Durrant-Whyte, and Gamini Dissanayake. “Constrained Initialization of the Simultaneous Localization and Mapping Algorithm”. In: *I. J. Robotic Res.* 22 (July 2003), pp. 541–564. DOI: 10.1177/02783649030227006.