ECE 276A Project 3: Visual-Inertial SLAM

Winston Chou  
*PID: A17460970*

*Abstract*—This project focuses on implementing a visual-inertial simultaneous localization and mapping (SLAM) system using an extended Kalman filter (EKF). The system integrates measurements from an inertial measurement unit (IMU) and a stereo camera to estimate both the trajectory of the robot and the positions of static landmarks in the environment. The IMU provides linear and angular velocity data, while the stereo camera captures visual features with precomputed correspondences between left and right frames. The SLAM process involves two main steps: an EKF prediction step based on IMU kinematics to estimate the robot's pose and an EKF update step using visual observations to refine landmark positions. The project assumes known extrinsic and intrinsic calibration parameters for the sensors. Results demonstrate the effectiveness of the proposed approach in estimating accurate robot trajectories and mapping landmark positions, despite challenges such as noisy measurements and partial observability.

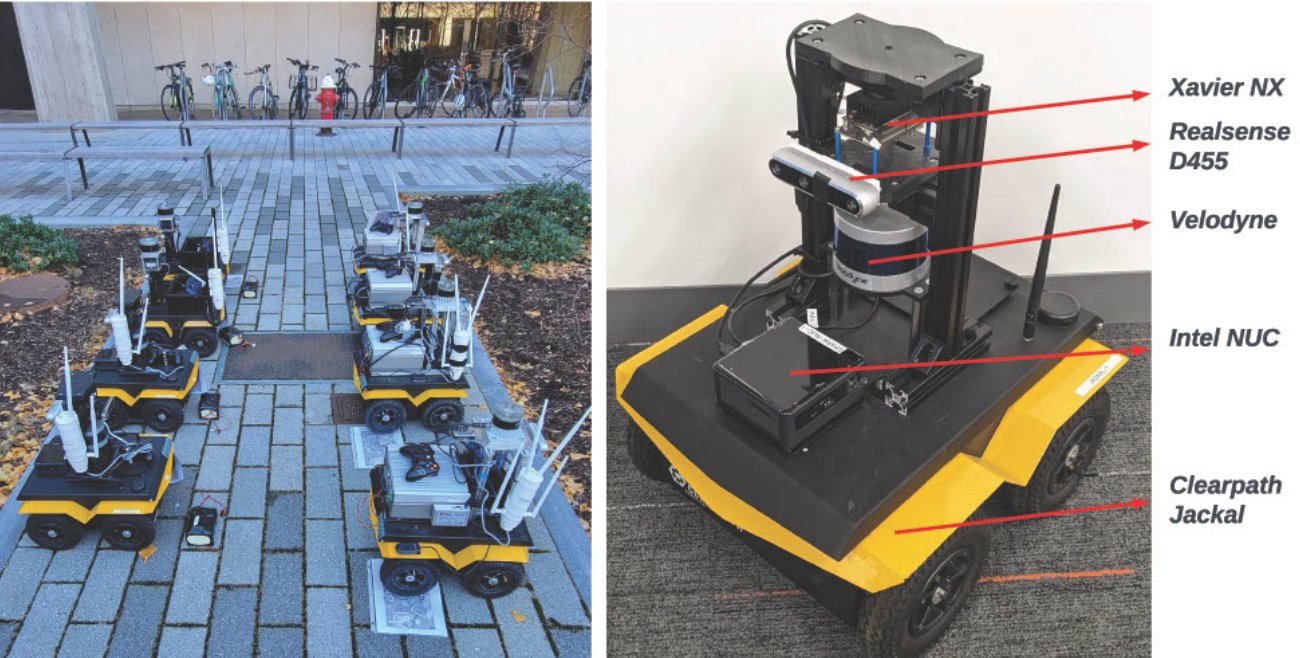
Keywords—SLAM, Markov Chain, pose estimation, multi-sensor fusion, IMU, RGBD camera

# Introduction

Simultaneous localization and mapping (SLAM) is a fundamental problem in robotics that involves estimating a robot's trajectory while simultaneously building a map of its environment. This capability is critical for autonomous navigation in various applications, including self-driving cars, drones, and robotic exploration. SLAM is particularly challenging due to uncertainties in sensor measurements, dynamic environments, and computational complexity.

This project aims to solve the SLAM problem using data from an IMU and a stereo camera mounted on a vehicle. The IMU provides linear and angular velocity measurements, while the stereo camera captures visual features used for landmark mapping. By leveraging an extended Kalman filter (EKF), the system integrates these sensor inputs to estimate both the robot's pose and the positions of static landmarks in 3D space.

The proposed approach consists of two key components: an EKF prediction step that uses IMU kinematics to estimate the robot's trajectory over time, and an EKF update step that incorporates stereo camera observations to refine landmark positions. Known extrinsic calibration between the IMU and camera frames, as well as intrinsic camera parameters, are utilized to ensure accurate sensor fusion.

This report details the implementation of the visual-inertial SLAM system, including problem formulation, technical approach, results, and analysis. The findings highlight the potential of combining visual and inertial data to achieve robust localization and mapping in real-world scenarios.

1. Sensor Setup. Clearpath Jackal robots on MIT’s campus, equipped withIMU, 3-D LIDAR scanner, and an RGBD camera, Realsense D455. [Source: ECE276A\_PR3.pdf]

# problem formulation

## Simultaneous Localization And Mapping

SLAM can be described as a probabilistic process modeled as a Markov Chain. At discrete time steps , the robot's pose  evolves based on the control input  and motion noise . The pose at the next time step  is determined by a probabilistic function:

Where represents motion noise. This relationship is commonly referred to as the **motion model**.

To map the environment, the robot collects observations at each time step. Let the observation at time  be denoted as  and the environment as . The sensor observations follow a probabilistic relationship:

Where represents observation noise. This relationship is commonly referred to as the **observation model**.

The SLAM problem involves estimating the environment  and the robot's poses  using observations and control inputs at each time step . The goal is to compute the joint probability distribution of the environment and robot poses conditioned on the observations and control inputs. This probabilistic formulation captures the uncertainty inherent in SLAM:

The relationship between the environment and robot poses is challenging to determine directly. However, leveraging the Markov assumptions allows for decomposing the joint probability density function into manageable components. This decomposition includes terms for the initial state of the robot and environment, observation likelihoods based on sensor readings, and motion probabilities derived from control inputs and previous poses.

In practical SLAM implementations, maximum likelihood estimation (MLE) is often used to find optimal values for the robot's trajectory  and the environment . The optimization process involves maximizing the sum of log-likelihoods for sensor observations and motion probabilities over all time steps. This approach helps determine both the robot's poses and a map of the environment effectively.

## Bayesian Filtering

Bayes filtering is a probabilistic method used to estimate the state of dynamic systems, such as a robot's pose, by combining information from control inputs and observations. This technique relies on the Markov assumptions and Bayes' rule to infer the system's state over time. The Bayes filter operates in two main steps:

1. *Prediction Step:*  
   In this step, the prior probability distribution of the system state at time , along with the control input, is used to predict the state at the next time step . The motion model governs this process, and the predicted probability distribution is computed by integrating over all possible states at time :
2. *Update Step:*  
   Once a new observation is received at time , the predicted probability distribution is updated using the observation model. This step incorporates measurement information to refine the estimate of the system's state at . The posterior distribution is normalized by dividing by a constant factor (marginal likelihood) to ensure it remains a valid probability distribution:

## Landmark-based Mapping

Landmark-based mapping focuses on creating a map of the environment using noisy and uncertain sensor observations, assuming the robot's poses are known. The environment is modeled as a set of static landmarks, with each landmark represented by its location in 3D space. A landmark's position is denoted as , where  ranges from 1 to , the total number of landmarks. Collectively, the landmarks are represented as a matrix , with each landmark specified by three numerical values corresponding to its coordinates.

The robot can detect landmarks at each time step , and the observations are denoted as . Since multiple landmarks may be sensed simultaneously,  represents a composite observation encompassing all detected landmarks at that time.

The goal of this mapping process is to estimate the locations of the landmarks based on the robot's pose  and the observations . This estimation relies on an observation model, which defines the probabilistic relationship between the observations, robot pose, environment landmarks, and measurement noise.

An index map  is used to track which landmarks correspond to specific observations. At each time step , the robot observes  landmarks, with each observation denoted as  (Homogeneous Coordinates), where . The data association map specifies that the  landmark corresponds to the observation indexed by .

## Sensors Setup

The proposed solution addresses the SLAM problem using data from an IMU and a stereo camera mounted on a vehicle. The IMU provides measurements of linear velocity () and angular velocity (), both expressed in the IMU's frame of reference. The stereo camera captures visual data, with precomputed visual features and correspondences established between the left and right camera frames as well as across time steps (data association).

At each time step , the visual features are represented as , where each column corresponds to a landmark. Specifically, the  column contains the pixel coordinates of the  landmark in both the left and right camera images. If a landmark is not observable at time , its corresponding column in  is set to .

The system assumes that both followings are known:

1. *The transformation from the IMU frame to the stereo camera's optical frame* ( ) *is known (extrinsic calibration)*
2. *The stereo camera calibration matrix () is also known (intrinsic calibration). The calibration matrix is defined as:*

Where is the focal length, , are pixel scaling, and are the principal points, and is the stereo baseline.

# Technical approach

The implementation of the project is presented as follows:

## Sensor Data Interpolation on Timstamps

Robotic sensors operate at varying frequencies, and synchronization is not enforced. To align measurements temporally, interpolation is critical for accurate sensor fusion and downstream tasks. The following strategies are employed:

All interpolations are done by using “np.interp”.

* Wheel encoder data is linearly interpolated to match the IMU’s timestamp.
* The motion model’s poses (derived from differential-drive kinematics and IMU) are interpolated to LiDAR timestamps when providing a reliable initial guess for the ICP algorithm, accelerating convergence.
* Poses are interpolated at LiDAR timestamps in occupancy mapping.
* Poses are interpolated at camera timestamps in occupancy mapping.

## Encoders Preprocess

The encoders count the rotations of the four wheels at 40 Hz. The encoder counter is reset after each reading. Therefore, we are actually dealing with a twist with a timestep. Encoder data from the four wheels (FL, FR, RL, RR) is processed to compute the traveled distance in meters. The steps are as follows:

This is how we convert encoder counts to distance in meters for all four encoders.

## 2D and 3D Conversion

The poses often need to be converted between 2D and 3D representations.

## Transform Tree

Based on the “RobotConfiguration.pdf”, the following transformations are required for converting reference frames.

## Iterative Closest Point (ICP)

The Iterative Closest Point (ICP) algorithm is a widely used method for aligning two point clouds by iteratively minimizing the distance between corresponding points. In this project, ICP is applied for scan matching to estimate relative transformations between LiDAR scans and refine the robot's trajectory. Below is an elaboration of the ICP process and its performance during warm-up tests and scan matching.

To improve ICP performance in scan matching, several optimizations are implemented:

### Nearest Neighbor Search:

### Two methods were tested for finding point correspondences:

### KDTree from scipy

* NearestNeighbors from sklearn

The NearestNeighbors method was chosen due to its faster response time under identical tolerance conditions.

### Outlier Filtering:

### An ICP threshold version is used to filter outliers by discarding points that are too far apart.

### Error Calculation with Huber Loss:

### A Huber loss function is employed to calculate alignment errors, reducing the influence of outliers and improving overall robustness.

### Multiple Initial Guesses:

### To mitigate sensitivity to initial guesses, multiple guesses with different yaw angles are tested. The guess yielding the lowest RMSE is selected as the starting point for ICP.

## Factor Graph

In this project, two factor graphs are constructed to compare the performance of SLAM with and without loop closure. The factor graph is designed to represent the robot's trajectory and constraints between poses, including motion, observation, and loop closure constraints.

### Loop Closure Criteria

*The criteria for detecting potential loop closure pose pairs are as follows:*

* ***Maximum location difference****:*
* ***Maximum yaw difference****:*
* ***Minimum time interval****:*

*These criteria ensure that loop closures are only added when poses are spatially close and temporally distant, reducing the risk of false associations.*

### Constraints

* ***Motion Constraints****: Represented with a static diagonal variance of:*

*These constraints are derived from the motion model based on differential-drive kinematics and IMU data.*

* ***Observation and Loop Closure Constraints****: The variances for these constraints are proportional to the ICP error:*

*Here, ϵsϵs is the ICP alignment error for the corresponding scan match.*

### Comparison

* ***Without Loop Closure****: The factor graph forms a chain structure where errors accumulate over time due to the lack of global corrections.*
* ***With Loop Closure****: The graph incorporates additional constraints (loops) that connect spatially close but temporally distant poses. This reduces drift and improves the accuracy of both trajectory estimation and mapping.*

## Mapping

Mapping is performed in two stages: occupancy mapping using LiDAR data and texture mapping using RGB-D data.

### Discretized Map

The world is discretized into a grid with a resolution of **10 cells per meter** (i.e., each cell represents a 0.1 m×0.1 m area). All points are rounded to their respective cell centers for computational simplicity.

### LiDAR to Occupancy Mapping

LiDAR scans are used to update the occupancy probabilities of grid cells:

* ***Empty Cells****:* Cells along the line between the LiDAR sensor origin and endpoints decrease their odds ratio by −2.
* ***Occupied Cells****:* Endpoint cells increase their odds ratio by +2.

The occupancy probability is calculated using logistic regression:

This probabilistic approach accounts for uncertainty in sensor measurements while maintaining computational efficiency.

*3) RGB-D Texture Mapping*

RGB-D data is used to create a textured map of the environment:

* ***Intrinsic Matrix****:* The camera's intrinsic matrix K is given as:
* ***Depth Conversion****:* Disparity values dd are converted to depth z using:
* ***Coordinate Transformation****:*
  + The optical-to-camera rotation matrix is:
  + Floor points are isolated based on height
* ***RGBD Alignment****:*
  + The disparity camera and RGB camera have an offset along the x-axis.
  + Pixel coordinates in the disparity image (i,j) are mapped to RGB image coordinates (rgbi, rgbj) using:

# results

***Dataset 20***

*Motion Model:* The motion model provided a solid baseline trajectory (Fig. 1a). While it is not perfect, it serves as a reliable starting point for further optimization.

*Observation Model:* The observation model improved upon the motion model by refining details.

*Factor Graph Without Loop Closure:* Not Implemented

*Factor Graph With Loop Closure:* Somehow did not work will with dataset 20, could be poor loop closure detections due to dense overlap between scans.

***Dataset 21***

*Motion Model:* The motion model struggled in areas such as the vertical hallway, introducing errors in the trajectory (Fig. 2a).

*Observation Model:* The observation model failed to correct these errors and instead converged to a local minimum, worsening the trajectory (Fig. 2b). This highlights the sensitivity of ICP to poor initial guesses.

*Factor Graph Without Loop Closure:* Not Implemented

*Factor Graph With Loop Closure:* Loop closure constraints helped correct some errors in the x-coordinate but introduced orientation inconsistencies, leading to unreliable mapping results (Fig. 2d).

***Key Observations***

*ICP Sensitivity:*

Both the standard ICP algorithm and its modified version are prone to converging to local minima. A good initial guess for translation and rotation is critical for achieving accurate results. Using multiple initial guesses with varying yaw angles, as done in this project, helps mitigate this issue.

***Potential Improvements:***

Equipping the robot with a 3D LiDAR or utilizing a full 3D point cloud from an RGB-D camera could enhance ICP performance by providing more data points and an additional dimension for alignment.

These enhancements would reduce reliance on precise initial guesses and improve robustness against local minima.

***Conclusion***

The results demonstrate that while motion models provide a reliable starting point, observation models and factor graphs with loop closure constraints are essential for refining trajectories and correcting drift. However, ICP's sensitivity to initial guesses remains a limitation, particularly in challenging environments like Dataset 21. Future work should explore integrating richer sensor data (e.g., 3D LiDAR) to enhance robustness and accuracy in scan matching and trajectory estimation.

1. Dataset 20
2. Dataset 21
3. ICP warm up images

##### References

1. N. Atanasov, UCSD ECE276A: Sensing & Estimation in Robotics (Winter 2025), https://natanaso.github.io/ece276a/schedule.html.