

CHAPTER III

RESERCH METHODOLOGY

3.1 Methods of data collection

During the experiments, all data generated by the system, including both processed and raw sensor data, were transmitted through the Robot Operating System (ROS) framework. ROS provides a standardized communication mechanism that allows different system components to exchange information in real time.

To record experimental data, a built-in ROS tool known as rosbag was utilized. Rosbag enables the recording of ROS messages published on selected topics and stores them in files with the bag format. This functionality allows comprehensive logging of sensor measurements, localization outputs, and system states throughout the experiment.

In addition to data recording, rosbag supports data playback, which makes it possible to replay previously recorded experiments offline. This capability is particularly useful for repeated analysis, algorithm evaluation, and performance comparison under identical conditions without requiring the physical robot to be rerun. As a result, rosbag provides a reliable and reproducible method for collecting and analyzing experimental data in this research.

3.2 Procedure

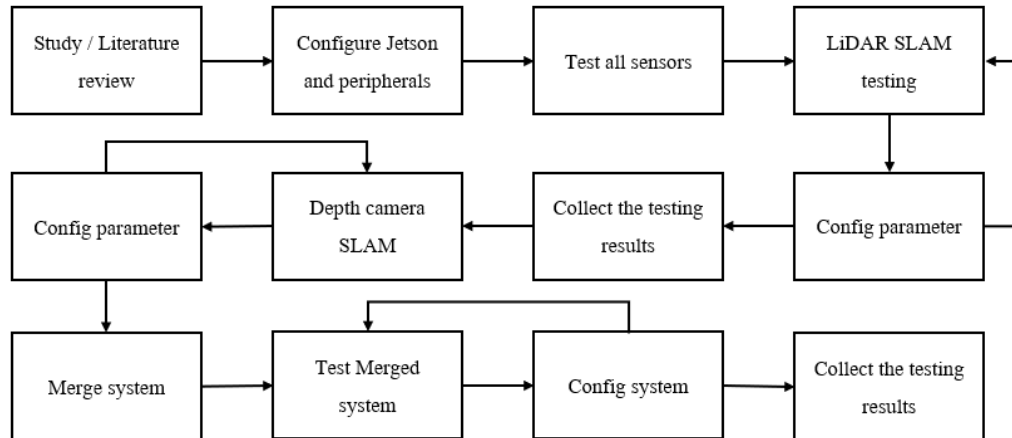


Figure 3.1 Procedure

3.2.1 Study and Literature review

comprehensive literature review was conducted based on academic papers, technical documents, and previous research related to SLAM, LiDAR, sensor fusion, and depth camera technologies. In addition, studies concerning robotic control systems and data communication within the Robot Operating System (ROS) framework were reviewed.

The purpose of this review was to gain a clear understanding of the underlying principles and theoretical foundations relevant to this research. Particular attention was given to the strengths and limitations of different sensing modalities, as well as their suitability for localization and mapping tasks. Furthermore, existing experimental methodologies and data collection strategies were examined to support the design of appropriate testing procedures.

This review provided essential guidance for selecting suitable sensors, designing experimental scenarios, and collecting relevant data, ultimately contributing to a complete research framework capable of accurately estimating robot position in environments suitable for camera-based operation.

3.2.2 Configure Jetson and Peripherals

This stage focuses on configuring the NVIDIA Jetson Xavier NX and its associated peripheral devices for use on the F1TENTH autonomous platform. The setup process includes establishing remote communication with the Jetson via Secure Shell (SSH) and connecting an external monitor to enable system monitoring and configuration.

These steps allow the Jetson to execute autonomous programs on the F1TENTH platform while maintaining reliable communication with a laptop for development, debugging, and system control. The following figure illustrates the data and hardware connections used within the system architecture.

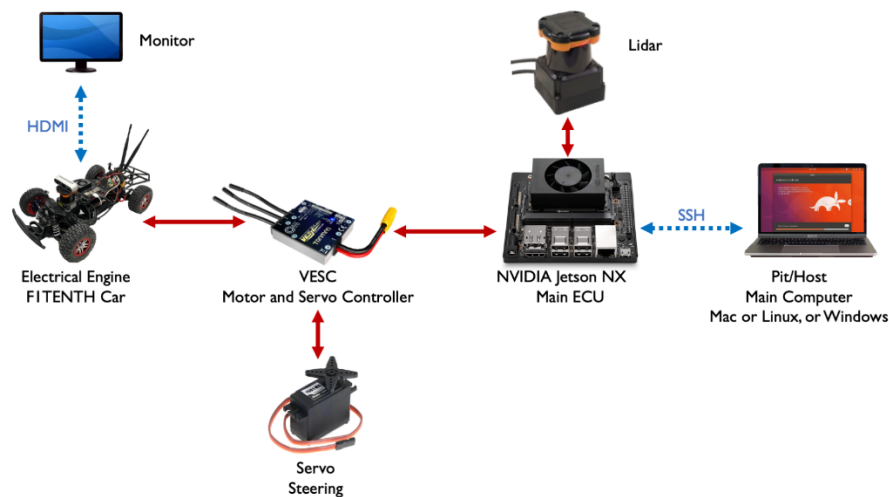


Figure 3.2 System architecture of the F1TENTH platform, illustrating the data flow and control connections between the NVIDIA Jetson Xavier NX, LiDAR sensor, VESC motor and steering controller, onboard actuators, and the pit/host computer via SSH and HDMI interfaces.

The NVIDIA Jetson Xavier NX serves as the central processing unit of the entire system. It is responsible for issuing control commands to the Vedder Electronic Speed Controller (VESC), which manages the steering servo and the brushless motor of the F1TENTH platform. In addition to control tasks, the Jetson NX receives sensory data from the Light Detection and Ranging (LiDAR) sensor and the depth camera, which are connected via USB or Ethernet interfaces.

All communication and data exchange within the system are handled through the **ROS 2** framework. A laptop acting as the pit or host computer is used to

remotely access the onboard system, enabling monitoring, debugging, and control during operation. This host computer also supports real-time visualization of system data through a connected monitor using an HDMI interface.

3.2.3 Test all sensors

The objective of this testing phase is to verify the operational integrity and data accuracy of the primary perception and localization suite—specifically the 2D LiDAR, the RGB-D depth camera, and the wheel-encoder-based odometry. The validation process follows a tiered approach, beginning with individual sensor stream verification and concluding with integrated motion testing. Verification commences by initializing the Robot Operating System (ROS) environment to ensure all sensor nodes are publishing to their designated topics. Using the `rostopic echo` or `ros2 topic echo` command, the frequency and consistency of data on `scan` (LiDAR), `camera`, `depth`, `image_raw` (Depth Camera), and `odom` (Odometry) are monitored. A consistent publishing rate.

The 2D LiDAR is tested using a scan-matching algorithm (e.g., SLAM Toolbox). Key parameters, specifically `scan_buffer_size` and `minimum_time_interval`, are tuned to handle high-speed data intake.

3.2.4 SLAM testing and configuration parameter

This research phase focuses on the independent optimization of LiDAR-based and Depth-based SLAM systems. The objective is to achieve a synchronized state where both modalities exhibit minimal drift and high map fidelity, ensuring they are prepared for sensor fusion.

3.2.5 Merge system

The core of the navigation stack relies on a LiDAR-centric SLAM approach (e.g., utilizing a LiDAR Odometry and Mapping framework). The justification for using LiDAR as the exclusive localization base the Depth Camera as the volumetric data source.

3.3 The overall system operation.

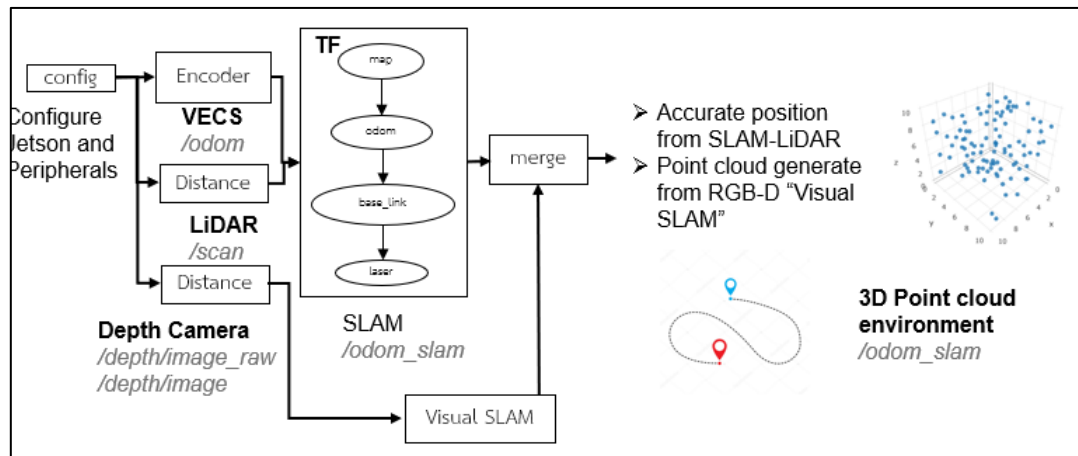


Figure 3.3 The Overall system operation

Methodology used to develop and evaluate a multi-sensor SLAM system on the F1TENTH autonomous robotic platform. The proposed approach integrates 2D LiDAR-based SLAM for robust localization with depth camera-based sensing for dense environmental mapping, particularly under challenging lighting conditions where vision-based localization becomes unreliable

The overall system configuration consists of three main components: hardware configuration, software architecture, and sensor integration. These components work together to enable localization, mapping, and data fusion in real time.

3.3.1 Computation Unit: NVIDIA Jetson NX

An NVIDIA Jetson NX is used as the primary onboard computing unit. This embedded computer is responsible for processing LiDAR scan data, Acquiring and processing depth camera images, executing SLAM algorithms, Managing ROS2 node communication, Broadcasting and receiving TF transformation



Figure 3.4 NVIDIA Jetson NX

The Jetson NX is selected due to its high-performance CPU and GPU capabilities, which enable real-time processing of depth images and point clouds while maintaining low power consumption. The system runs Ubuntu Linux as the operating system, with ROS 2 installed as the middleware framework.

The Jetson is mounted centrally on the chassis to reduce vibration and ensure stable cable routing to connected sensors.

3.3.2 Hardware Configuration

Computation Unit An NVIDIA Jetson NX is used as the main onboard computer. It performs sensor data processing, SLAM computation, and communication between software nodes. The Jetson platform is selected due to its balance between computational performance and power efficiency, making it suitable for real-time robotic applications.

Actuation and Odometry Source Vehicle motion is controlled using a Vedder Electronic Speed Controller (VESC), which drives the electric motor and steering servo. Wheel encoder data obtained from the VESC is used to estimate vehicle motion, producing raw odometry information that represents incremental displacement over time.

Sensor Mounting and Frame Alignment All sensors, including the LiDAR and depth camera, are rigidly mounted on the vehicle chassis. Their relative positions are defined using static transforms to ensure consistent spatial relationships between sensor frames and the robot base frame

3.3.2.1 Actuation and Odometry Source: VESC Controller



Figure 3.5 VESC

The vehicle's propulsion and steering are controlled using a Vedder Electronic Speed Controller (VESC). The VESC interfaces with the electric motor and steering servo and provides feedback from wheel encoders.

Wheel encoder measurements are used to estimate the rotational speed of the drive wheels. From these measurements, incremental motion information is computed and published as odometry data (/odom). This odometry represents relative motion based on wheel rotation and is subject to cumulative drift over time due to wheel slip and surface irregularities.

Despite these limitations, wheel odometry provides high-frequency motion estimates that are essential for short-term motion tracking and sensor synchronization.

3.3.2.2 2D LiDAR Sensor

A 2D LiDAR sensor is mounted on the top of the vehicle, aligned with the forward direction of motion. The LiDAR emits laser beams in a horizontal plane and measures the distance to surrounding obstacles based on time-of-flight or phase-shift principles.



Figure 3.6 2D LiDAR

The LiDAR publishes distance measurements as a laser scan (/scan), which is used as the primary input for LiDAR-based SLAM. The LiDAR frame is statically transformed to the robot base frame to ensure accurate spatial alignment during localization.

3.3.2.3 Depth Camera Sensor



Figure 3.7 D435i

A depth camera (Intel RealSense D435i) is mounted at the front of the vehicle to capture three-dimensional information about the environment. The camera provides depth images in which each pixel corresponds to the distance between the camera and observed surfaces.

The camera includes both infrared sensors and an RGB sensor; however, in this research, the depth stream is the primary data source. The depth camera is rigidly mounted, and its coordinate frame is defined relative to the base_link frame through a static transformation.

3.3.2.4 Sensor Mounting and Coordinate Frame Alignment

Accurate sensor fusion requires precise knowledge of the relative positions and orientations of all sensors. Each sensor is mounted rigidly to minimize mechanical vibration and displacement during motion.

Static transformations are defined between:

base_link and LiDAR frame

base_link and camera frame

Camera frame and camera optical frames

These transformations are published using ROS 2 static transform publishers and remain constant throughout operation. Correct frame alignment ensures that LiDAR scans, depth point clouds, and odometry data can be consistently represented in a shared coordinate system.

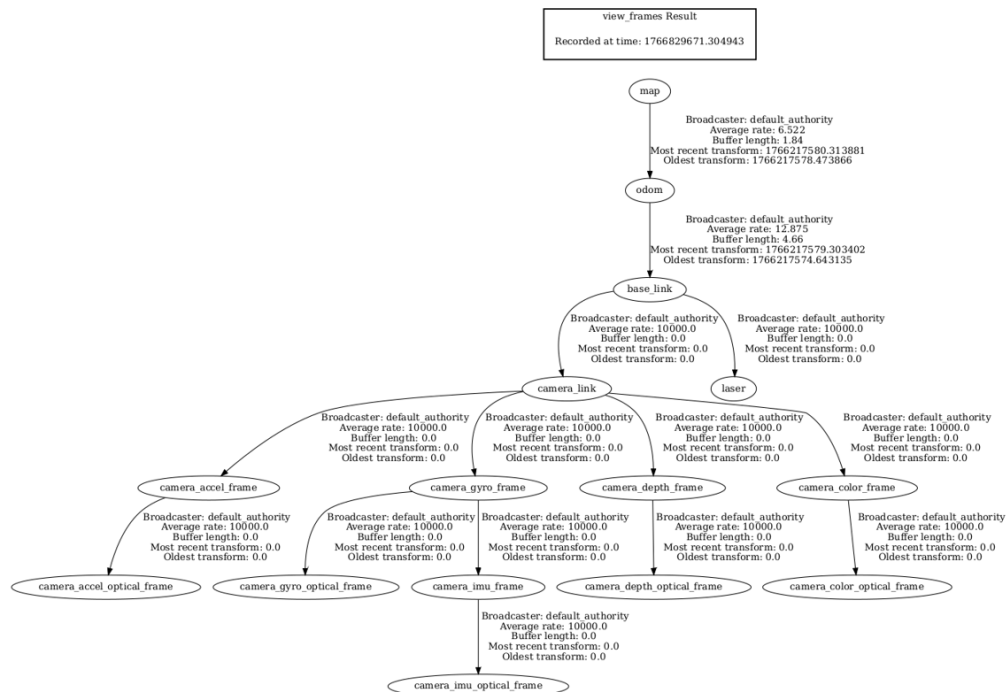


Figure 3.8 Coordinate frame hierarchy of the proposed system visualized using the ROS TF tree. The diagram illustrates the transformation relationships between the global map frame, odometry frame, robot base frame, LiDAR frame, and RGB-D camera frames.

3.3.3 Software Architecture

The software framework used in this research is the Robot Operating System 2 (ROS 2). ROS 2 provides a modular and distributed architecture that enables communication between different software components through topics, services, and the TF transformation system. The TF framework is used to manage coordinate

transformations between frames such as map, odom, base_link, and sensor frames. This ensures that data from different sensors can be fused consistently within a common reference frame.

3.3.3.1 LiDAR-Based Localization

A 2D LiDAR sensor is used to measure distances to surrounding obstacles in a horizontal plane. The LiDAR continuously publishes range measurements as a laser scan, which represents the environment as a set of angular distance samples.

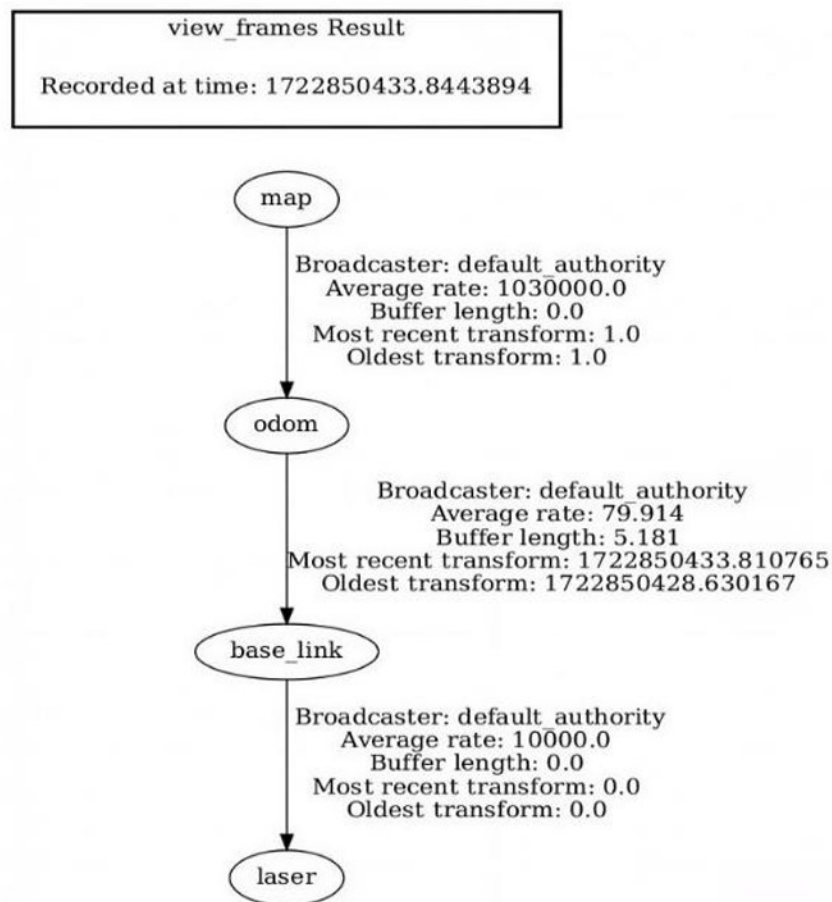


Figure 3.9 Simplified ROS TF frame structure showing the transformation chain from the global map frame to odom, base_link, and the LiDAR (laser) frame.

The LiDAR is configured to operate at a fixed scanning frequency and is aligned with the robot's forward direction. The LiDAR data is published in the sensor frame and transformed into the robot base frame using a static TF.

In this research, LiDAR-based SLAM is employed primarily for localization rather than dense mapping. The SLAM algorithm estimates the robot's pose by matching consecutive laser scans against a growing map of the environment.

The output of the LiDAR SLAM system includes, A 2D occupancy grid map representing the environment, A pose estimate of the robot relative to the map, A transformation between the map and odom frames

This localization output is published as `/odom_slam` and provides a drift-corrected estimate of the robot's motion. Because LiDAR measurements are largely invariant to lighting conditions, the localization remains stable even when visual sensors degrade.

3.3.3.2 LiDAR-Based Localization

Depth images are converted into 3D point clouds using a pinhole camera model. Each pixel in the depth image is reprojected into a 3D point based on its depth value and the camera's intrinsic parameters.

This process produces a dense point cloud that represents the local geometry of the environment in the camera coordinate frame. The point cloud is then transformed into the global reference frame using TF information.

3.3.4 Integration of LiDAR Localization and Depth Mapping

The proposed system adopts a loosely-coupled sensor fusion approach. In this framework, each sensor processes its data independently, and the results are combined at the pose level rather than at the raw measurement level.

LiDAR-based SLAM provides the robot pose estimate, while the depth camera provides dense spatial information. The two subsystems communicate through shared coordinate frames rather than tightly integrated measurement models. The pose output from the LiDAR SLAM system (`/odom_slam`) is used to position the depth-based point clouds in the global map frame. This ensures that the dense map follows the accurate trajectory estimated by LiDAR, even when depth data quality degrades. By decoupling localization and dense mapping, the system prevents visual drift from affecting the robot's estimated trajectory.

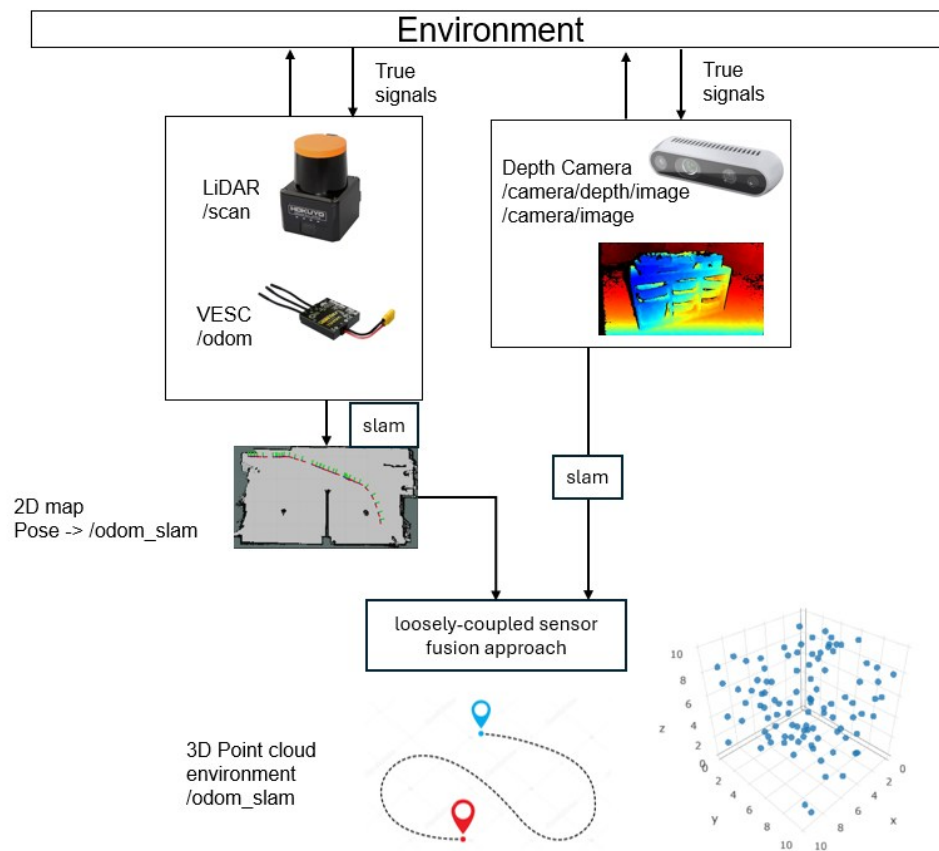


Figure 3.10 Loosely-coupled sensor fusion concept