การหาตำแหน่งในร่มและกลางแจ้งในสภาพแวดล้อมที่ไม่รู้ล่วงหน้าสำหรับพาหนะทางอากาศที่ไร้คนขับด้วยการตรวจจับหน้าต่าง

นายลภนชัย จิรชูพันธ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตร์มหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2565

INDOOR AND OUTDOOR LOCALIZATION IN AN UNKNOWN ENVIRONMENT WITH WINDOW DETECTION FOR UNMANNED AERIAL VEHICLES

Mr. Laphonchai Jirachuphun

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2022

Thesis Title	INDOOR AND OUTDOOR LOCALIZATION IN AN UNKNOWN ENVIRONMENT WITH WINDOW DETECTION FOR UNMANNED AERIAL VEHICLES
By	Mr. Laphonchai Jirachuphun
Field of Study	Computer Science
Thesis Advisor	Assistant Professor Nattee Niparnan

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

. . . . Dean of the Faculty of Engineering

(Professor Supot Teachavorasinskun)

THESIS COMMITTEE

. . . . Chairman

(Assistant Professor Attawith Sudsang)

. . . . Thesis Advisor

(Assistant Professor Nattee Niparnan)

. . . . External Examiner

(Doctor Rungroj Jintamethasawat)

ลภนชัย จิรชูพันธ์: การหาตำแหน่งในร่มและกลางแจ้งในสภาพแวดล้อมที่ไม่รู้ล่วงหน้าสำหรับพาหนะทางอากาศที่ไร้คนขับด้วยการตรวจจับหน้าต่าง. (INDOOR AND OUTDOOR LOCALIZATION IN AN UNKNOWN ENVIRONMENT WITH WINDOW DETECTION FOR UNMANNED AERIAL VEHICLES) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ. ดร.นัททที นิภานันท์, 0 หน้า.

วิทยานิพนธ์เล่มนี้นำเสนอการออกแบบและสร้างระบบสำหรับอากาศยานไร้คนขับ (UAVs) ทางผู้จัดทำได้ทำการรวบรวม แก้ไขและปรับปรุงอัลกอริธึมโอเพ่นซอร์ส (open-source algorithm) ที่มีอยู่สำหรับงานหาตำแหน่งตัวตนในสภาพแวดล้อมในร่มและกลางแจ้งด้วยการตรวจจับหน้าต่างในช่วงที่หุ่นยนต์เคลื่อนที่ระหว่างทั้งสองสภาพแวดล้อม สำหรับภายนอกอาคารทางผู้จัดทำใช้ GPS ส่วนในตัวอาคารใช้ Extended Kalman Filter (EKF) ส่วนพื้นที่ระหว่างสองสภาพแวดล้อมที่ GPS ไม่น่าเชื่อถือและสภาพแวดล้อมมีโครงสร้างน้อยเกินไปสำหรับ EKF เราใช้หน้าต่างหรือประตูสำหรับการบะบุตำแหน่งด้วยกล้องสเตอริโอ (stereo camera) เราเปรียบเทียบเทคนิคของเรากับ 4 กรณีได้แก่ 1. GPS เท่านั้น 2. EKF เท่านั้น 3. เส้นการเดินทางที่ประเมินด้วยตนเอง และ 4. ข้อมูลเส้นการเดินทางจากกล้อง ZED2 ซึ่งทางผู้จัดทำใช้เป็นข้อมูลเท็จจริง (ground truth)

ภาควิชา	วิศวกรรมคอมพิวเตอร์	ลายมือชื่อนิสิต	....
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์	ลายมือชื่อ อ.ที่ปรึกษาหลัก	....
ปีการศึกษา	2565		

6170954021: MAJOR COMPUTER SCIENCE
KEYWORDS: LOCALIZATION/ AERIAL ROBOTS / INDOOR / OUTDOOR / ESTIMATION

LAPHONCHAI JIRACHUPHUN : INDOOR AND OUTDOOR LOCALIZATION IN AN UNKNOWN ENVIRONMENT WITH WINDOW DETECTION FOR UNMANNED AERIAL VEHICLES. ADVISOR : ASST. PROF. Nattee Niparnan, 0 pp.

In this thesis, we design and build a system for unmanned aerial vehicles (UAVs). We combine, adjust and improve existing open-source algorithms for localization tasks in an indoor and outdoor environment with window detection for transitioning between the two environments. For outdoor localization, we mainly use GPS, while for indoor localization, we use Extended Kalman Filter (EKF). However, for the transitioning area where the GPS is unreliable and the environment has too few structures for EKf, we use an opening such as a window or a door for localization with a stereo camera. We compare our technique with GPS only, EKF only, manually estimated path, and ground truth from a ZED2 camera.

Department:	Computer Engineering	Student's Signature	....
Field of Study:	Computer Science	Advisor's Signature	....
Academic Year:	2022		

Acknowledgements

I would like to thank assistant professor Nattee Niparnan, my advisor, who patiently guided me from the start until I finished this thesis. Even though I faced many obstacles on the way whether from an academic perspective or life hurdles, he always be there to give me support and valuable advice. There was one specific instance in which I am especially grateful for his guidance. In the beginning, my research and my work at OZT Robotics are aligned. However, halfway through my thesis, due to some business changes, my work at the company diverge from the original proposition. With the professor's counseling, I decided to work on two different works simultaneously instead of changing my thesis to be on the same path as my work. Professor Nattee supported me throughout my decision and patiently gave me time to adjust to this new change. I appreciated his total understanding.

Apart from being a peerless advisor, I would also like to mention that Professor Nattee was also my teacher in a study camp for International Olympiad in Informatics (IOI) from 2008 to 2011. He gave me passion and taught me the beauty of the inner work of computer programming since when I was young. He inspired me to pursue higher education in Computer Science. Nowadays, he still selflessly teaches and encourages new high school students at the IOI study camp.

I also thanked all professors and my friends in the ISL2 lab for exchanging great ideas during our weekly meeting. I also thanked OZT Robotics company for letting me work and study at the same time and always giving me a resource to help with this thesis.

Last but not least, I would like to thank my family who support me unconditionally and be my solid rock. They always listened to every one of my complaints, cheered me up when I was down, and encouraged me to finish this journey.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter I

INTRODUCTION

1.1 Introduction and Overview

Recently robotics field increasingly plays an important role in life. Nevertheless, many autonomous robotics tasks still pose challenging problems. One clear example is the DARPA Grand Challenge from the Defense Advanced Research Projects Agency in the United States of America. Going up a set of stairs, a very easy task for humans has proven to be surprisingly hard for a humanoid robot. For a robot to balance itself, it requires a fair amount of sensors and highly complex calculations, since the robot has many degrees of freedom with each one for each joint motion that the robot can move. This calculation is usually too expensive to carry out, so we have to do an approximation. Hence, we need to balance out the trade-off between time and accuracy. Apart from humanoid robots, other types of robots also face similar challenges, especially aerial vehicles which have a higher degree of freedom as we need to take into account roll and pitch as well as drastic changes in height.

This thesis explores an important task for aerial vehicles, how the robots localize themselves in an unknown environment. Moreover, the thesis will focus more on a specific type of environment, a transition between indoor and outdoor GPS-denied environments. Usually, aerial vehicle, such as drone, uses multiple sensors to detect their locations on the map. For an outdoor flight, it relies on Global Positioning System (GPS) for localization, and it uses cameras to do obstacles avoidance. For an indoor flight, where GPS is unreliable, it can use Laser sensors combined with Inertial Measurement Unit (IMU) to localize itself in an X-Y plane in Cartesian coordinates on an unknown map. Then, it can use a downward single-point laser to get a Z coordinate. However, the transition between indoor and outdoor environments is still a hard problem. On one hand, the GPS signal near buildings or tall objects is usually unusable due to signal loss or multipath problems. On the other hand, Laser sensors do not work well in an environment with little structure like in front of a building. The speed and angular velocity of the robot are also limited by the range of laser scans.

Even though the problem seems to limit the robot's movement in this uncertain environment, between indoor and outdoor environments, we can still exploit the fact that the robot needs to go through an opening for it to enter a building. We can use this information for localization. For simplicity, in this case, we assume that the robot needs to go through a window or a door. We can use cameras to do object detection and use the position and distance of the frame to increase the accuracy of the localization module. Thus, we need to combine these three scenarios for the robot to safely navigate from the outdoor to the indoor environment and vice versa.

1.2 Objective

The main objective of this thesis is to design and build a system and combine, adjust, and improve existing algorithms for localization in an unknown indoor/ outdoor environment for unmanned aerial vehicles (UAVs) using door/window detection. The main challenge is that the GPS signal inside and in nearby indoor environments is too unreliable. These types of environments are common in forests, Skytrain (BTS), or warehouses. For military use, this type of challenge has many applications such as intelligence, surveillance, target acquisition, or reconnaissance. For civilian use, drones will tremendously help in agriculture, warehouse management, or plant and factory inspection. At present, most of these tasks in complex environments require experienced pilots for manual flights. Hence, this thesis will enable the robotics community to explore indoor/outdoor localization more, leading to a better, faster, and cheaper way to do autonomous navigation in an unknown environment.

1.3 Scope

This research covers four scopes.

1. This work will build a system and adjust and combine existing algorithms and methods to localize for indoor/outdoor unknown environments with both 2D-laser scanners and cameras with window detection.
2. The work uses a self-created hexacopter with multiple different sensors for experiments. The technique achieved from this work should be able to apply to other types of robots, but the performance is not guaranteed to be the same.
3. This work focuses only on the localization part. It will not focus on the control, mapping, and navigation parts. Therefore, our datasets will be collected by hand-holding the robot.
4. All the doors and windows in this experiment are represented by large rectangular empty spaces framed by some obstacles. The window is not necessarily real windows.

Chapter II

BACKGROUND

The background for this work comprises five main topics: State Estimation, Mapping, Optical Flow, Scan Matcher, and Object Detection.

2.1 State Estimation

State estimation is a process of determining a robot's state from sensor measurements and control actions. The state of the robot or a robot's pose usually indicates its location and orientation. The pose for a standard aerial vehicle has six degrees of freedom (6-DOF). Three degrees for its position (X, Y, and Z) and another three for its orientation (roll, pitch, and yaw). The technique used in this thesis is a common technique called the Extended Kalman Filter (EKF), a non-linear version of the Kalman Filter (Thrun et al., 2005).

Kalman Filter is a type of Bayesian Filters that holds Markov assumption and assumes the system to be a linear Gaussian system. States of the robot, control actions noises, and sensor measurement noises are all Gaussian distributions. The state distribution is a linear function of its previous state probability, the current control action, and additive noise. The measurement probability is also a Gaussian distribution with different additive noise. These can be expressed by the following equations:

State transition

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{2.1}$$

Measurement

$$z_t = C_t x_t + \delta_t \tag{2.2}$$

Where x_t and x_{t-1} are the robot states at time t and $t-1$ respectively. u_t is a control action and z_t is a measurement. ε_t and δ_t are normal distribution noises. A_t, B_t, and C_t are transition matrix for the system.

Kalman Filter has two main steps: 1) the prediction step and 2) the measurement update step. The algorithm recursively runs these two steps to update the mean and variance of the robot state. For the prediction step, it calculates the predicted belief (state's mean and variance) by incorporating the control action shown in the state transition equation 2.1. For the update step, the algorithm updates the predicted belief by taking in the sensors' measurements according to 2.2. The result will be the new mean and variance of the new state. The algorithm is depicted below

Input ($\mu_{t-1},\ \Sigma_{t-1},\ u_t,\ z_t$)

Prediction Step

$$\begin{aligned} \overline{\mu}_t &= A_t\overline{\mu}_{t-1} + B_t u_t \\ \overline{\Sigma}_t &= A_t\Sigma_{t-1}A_t^T + R_t \end{aligned} \tag{2.3}$$

Update Step

$$\begin{aligned} K_t &= \overline{\Sigma}_t C_t^T (C_t\overline{\Sigma}_t C_t^T + Q_t)^{-1} \\ \mu_t &= \overline{\mu}_t + K_t(z_t - C_t\overline{\mu}_t) \\ \Sigma_t &= (I - K_t C_t)\overline{\Sigma}_t \end{aligned} \tag{2.4}$$

Output ($\mu_t,\ \Sigma_t$)

Where μ and Σ is the mean and variance of the state. R_t and Q_t are the variances of the normal distribution noises of the control action and measurement. K_t is a temporary variable called Kalman gain.

The system can be nonlinear for the Extended Kalman Filter, which is a more general case of the Kalman Filter. It follows the same procedure as in Kalman Filter but linearizes the system beforehand. EKF approximates the nonlinear system using first-order Taylor expansion. Since EKF does not require the system to be linear, the algorithm is more practical in real-world systems such as quadrotors.

2.2 Mapping

To represent the map of an environment in three dimensions (3D), I choose Occupancy Grid (Moravec and Elfes, 1985; Thrun et al., 2005). This technique evenly divides the environment in all dimensions into small and equal spaces forming blocks of cells. Each cell will associate with a random variable representing the state of that space in the environment. The random variables can be binary random variables, whose values will indicate whether the associated spaces are occupied by obstacles or not. The random variables can also be probabilities that show the likelihood of the spaces being occupied. Occupancy grid representation is straightforward. It is easy to understand and visualize humans. One downside is that this representation needs high computation power as the complexity is the cubic of the cell's resolution.

2.3 Optical Flow

The optical flow technique estimates a camera's motion from its sequential images (Horn and Schunck, 1981). The main idea is that neighbor pixels usually have relatively the same motions and intensities. The technique tries to track multiple key points appearing in one image to the next one. Then, it estimates the camera's instantaneous motion from the respective key points' displacements in consecutive images. The integral of camera instantaneous motions is the trajectory of the camera, which at the same time is the trajectory of the drone attached to that camera.

2.4 Scan Matcher

Since we mainly rely on scans from 2D laser for our estimation of the X-Y plan, we need to convert the change from two consecutive scans into the change of the robot's motion. One popular method of scan matching, first introduced by Besl, is the Iterative Closest Point (ICP) algorithm, which iteratively aligns a set of measured points (from the current sensor scan) to a reference set of points (from a previously mapped, known environment, or previous scan) by minimizing the distance between corresponding points (Besl and McKay, 1992). The algorithm estimates the rigid transformation (translation and rotation) that best aligns the two point sets.

Since then, several variations and improvements to the basic ICP algorithm have been proposed, such as using point-to-plane distance metrics, incorporating covariance estimation, or adding robustness to outliers. The Scan Matcher algorithm that we used in this work is proposed by Censi (Censi, 2008). Her Scan Matcher algorithm uses a Point-to-Line Iterative Closest Point technique (PL-ICP). In ICP, we repeatedly estimate the transformation from one scan to another by assuming that point from one scan and its closest point in another scan are the same points after applying the transformation. In PL-ICP, the technique is similar but instead of matching point to point, it matches point from one scan to an estimated line in another scan. This technique performs better as well as more robust to noises.

2.5 Object Detection

Object detection or object recognition is a task where you have to classify and locate objects in images into different classes that interest us. In this work, the classes that we consider are windows and doors which we assume that they are the gateway between indoor and outdoor environments. We decide to use the well-known YOLOv5 or You Only Look Once V5 for its high accuracy and small size (Jocher, 2020). YOLOv5 is a Convolutional Neural Network (CNN) object detection model. CNN is a type of neural network model that uses convolution operation to automatically extract meaningful features from images in multiple levels from small features to larger ones. A more detailed explanation of YOLOv5 can be found in Section 4.6.

Chapter III

LITERATURE REVIEW

The literature review for this thesis focuses on real-time localization for both mobile and aerial robots in an unknown environment with only onboard sensors. Usually, the localization topic is tightly coupled with the mapping topic, so both topics will be discussed here.

3.1 SLAM

SLAM or Simultaneous Localization and Mapping is one of the famous problems in robotics. It is a problem for a robot to navigate in an unknown environment. The robot needs to know the environment map and also at the same time be able to locate itself on the map. Usually, an unmanned aerial vehicle (UAV), uses multiple sensors to solve this problem. The most commonly used sensor is the Inertial Measurement Unit (IMU), which combines an accelerometer and a gyroscope. IMU provides the current acceleration and angular velocity for the robot. Still, IMU is not perfect. Long integration throughout flight trajectory for calculating the robot's position and orientation proves to be too inaccurate and unreliable. Hence, a combination of other types of sensors is used to improve the accuracy. Also, another unanswered question is mapping, since IMU provides no information about the map. Other types of sensors that interact with the environment are needed.

We can categorize SLAM using two criteria either by the environment the robot is in or by the sensors the robot is using. The types of environments are indoor, outdoor, or hybrid, including both indoor and outdoor environments. The sensor types are laser-based sensors, camera-based sensors, or hybrid. We can break down camera-based sensors further into monocular (single camera), stereo (two cameras), or multiple cameras. For the work in this thesis, the environment will be hybrid and the sensors used will also be hybrid.

3.2 SLAM by Sensor

Laser-based SLAM

Generally, there are two approaches to the SLAM problem and each one uses different types of sensors. Many of the early works are laser-based. They use laser scanners that send out a beam array of lasers in a 2D fashion and measure a relatively accurate detail of that planar portion of the map. With multiple scans, we can do iterative closest points (ICP) to determine the change in positions and orientation. Integrating this change over time will give the current position and orientation. Their downsides are their limited range and heavy weight. The affordable laser scanner has roughly about 30-50 meters in range. With the limited range, robots or drones can only move slowly in a small-scale and well-structured environment. Their upsides are their precise measurement of the map and that they have zero concern about lightning conditions in the environment but these also come with high cost.

Some previous works show that laser-based SLAM work well indoor and well-structured outdoor environment. In 2004, Lingemann uses a laser-based approach for tracking the pose of a high-speed mobile robot (Lingemann et al., 2004). In 2005, Diosi improves the scan matching algorithm with polar coordinates and proves that Kalman Filter works well for a robot in a 2D environment (Diosi and Kleeman, 2005). In 2014 Urcola proposes a seamless indoor and outdoor transition for a 2D robot (Urcola et al., 2014). He uses GPS quality to switch between each situation where a robot needs to rely on different sensors. Our work is similar to his work but our domain is in 3D and we separate the environment into three zones: indoor, outdoor, and in-between zone. We also use object detection techniques to combat uncertainty in the in-between zone. In 2018, Dowling manages to get an accurate 3D indoor map with a drone equipped with a laser scanner and an ultrasonic sensor using EKF (Dowling et al., 2018).

Camera-based SLAM

Another approach is to use cameras. The downside of camera-based is the computation power. Processing an image usually require much more computation resource than laser-based. However, computers are becoming more and more powerful, so the camera approach is now computable in real-time. There are three types of camera sensors: monocular, stereo, and multiple cameras. Since batter time or flight time for an aerial vehicle is determined by the power consumption rate which is directly proportional to the weight of that vehicle, all components are desirable to have small sizes, light weights, and minimum power consumption (SWAP) (Bertran and Sànchez-Cerdà, 2016). Thus, monocular cameras are one of the obvious choices.

3.3 SLAM by Technique

Filter-based

Two main techniques for camera-based SLAM are filter-based SLAM and keyframe-based SLAM. The filter-based SLAM is similar to the laser-based one. Instead of doing an ICP on the laser scans to determine the change in the pose of the robot, filter-based SLAM can calculate the change from two consecutive images. In 2003, Davison proposed a novel real-time approach for monocular SLAM. He used extended Kalman filtered (EKF) to estimate the camera pose and mapping (Davison, 2003). However, since EKF has high computational complexity, the selected feature must be sparse. This approach only works on small-size problems. In 2007, Nuchther et al. used particle filters instead of EKF and achieved a 3D outdoor SLAM increasing the environment flight space hugely (Nuchter et al., 2006). In 2009, Huang et al. managed to use Unscented Kalman Filter (UKF) in real-time by creating a new sampling strategy to reduce the number of states it required to compute (Huang et al., 2009).

Keyframe-based

For keyframe-based SLAM, the main idea is to remember the location of each distinguishable landmark. When the next time the robot comes around and sees the same landmark, it can localize itself. The landmarks sometimes are distinct edge features or corner features. The image that contains the landmarks is called a keyframe. In 2007, Klein and Murray designed the first keyframe-based SLAM, PTAM, working for a small indoor workspace environment with sparse mapping (Klein and Murray, 2007). In 2011, Newcombe's work, DTAM, improved PTAM by using a GPU parallel computation (Newcombe et al., 2011). Instead of a sparse feature, he used a dense one (at all pixels). His approach achieved both dense tracking and mapping for indoor environments. In 2015, Mur-Artal created an ORB-SLAM algorithm that works for both indoor and outdoor environments (Mur-Artal and Tardos, 2014). The author uses a semi-dense reconstruction technique.

3.4 Recent Approaches

For stereo and multiple cameras SLAM, the technique used is the same as in the monocular SLAM. They have more information on the depth of the images, so in terms of localization and mapping, they usually perform better. One example is Yang's work (Yang et al., 2017). The author integrated views from multiple cameras on micro aerial vehicles (MAVs) and achieved high accuracy for the robot's pose. Nevertheless, the extra weight of another camera sometimes is undesirable. Many tasks require small drones which can carry only some small weight. This is not the case for this thesis work, so we will use a stereo camera.

Another recent approach is to combine both laser-based SLAM and camera-based SLAM. Laser-based SLAM is good for local navigation and mapping while camera-based is good for the global ones. Newman used this technique for a mobile robot (Newman et al., 2006). The author used the laser to build a simple local environment geometry while using the camera to detect loop closure to improve the global map for navigation tasks. Another example is Bachrach's experiment (Bachrach et al., 2010). His work shows that a micro aerial vehicle (MAV) can achieve a full SLAM using EKF from a laser scanner and stereo camera. For our work, we use both lasers and cameras but we will prioritize using laser-based SLAM. We will cameras only for object detection and depth estimation.

Chapter IV

INDOOR AND OUTDOOR LOCALIZATION WITH WINDOW DETECTION

In this chapter, we will discuss our approach to localizing a drone in an unknown environment transitioning between indoor and outdoor using door/window detection. One major assumption is that the drone has to go through an opening like a door or a window to move from an indoor environment to an outdoor environment. The proposed method is to separate the environment into three groups: indoor with high structure data for laser-based method, outdoor with reliable GPS signals, and outdoor near buildings with unreliable GPS. First, we will talk about the procedure of the thesis, then reasoning, and finally our method.

4.1 Procedure

1. Build a drone: Build a self-created hexacopter that can fly manually. We use this drone for experiments in this work as well as other projects that I am a part of.

2. Sensors combination: Add single-point laser, 2D laser, and IMU to the drone.

3. Indoor and Outdoor Localization research: Research various techniques for indoor and outdoor localization.

4. Indoor Localization: Integrate all the sensors (single-point laser, 2D laser, and IMU) into Extended Kalman Filter (EKF).

5. External sensor integration: Integrate ZED2 stereo camera into our system and use RGB and depth image for opening detection.

6. Window/Door detection: Build and train window/door detection model using YOLOv5.
7. GPS integration: Integrate GPS data into the EKF.
8. Field Testing: Design and collect data from handheld tests of our system.
9. Analyze the result of the experiment.

4.2 Reasoning

The main reason for this separation is that from the past literature, robots can localize themselves extremely well in both indoor and outdoor environments but not between. This approach will still use state-of-the-art for those two types of environments. For outdoor, GPS will be the main localization module. For indoors, since the environment provides enough structure data for localization, the main core will use a 2D planar laser with a scan-matcher algorithm. However, for the transition between environments, these two techniques are inapplicable. On one hand, the GPS signal is unreliable because of a multipath signal from the surrounding buildings or the lower strength signal. On the other hand, the 2D planer laser does not work well because there is not enough structure for localization. Hence, we will exploit the assumption that there is a window between indoors and outdoors. We will use object detection to detect the window and approximate the distance and size of the window for localization.

4.3 Mode Switching

We partition the environment into three zones: indoor, outdoor, and in-between zone, where the in-between zone is an outdoor area near a building where the GPS signal is unreliable. For each zone of the environments, we use different algorithms for localization and call them 'mode'. This section here shows how we switch from one mode to another. The diagram in Figure 4.1 depicts the criteria for

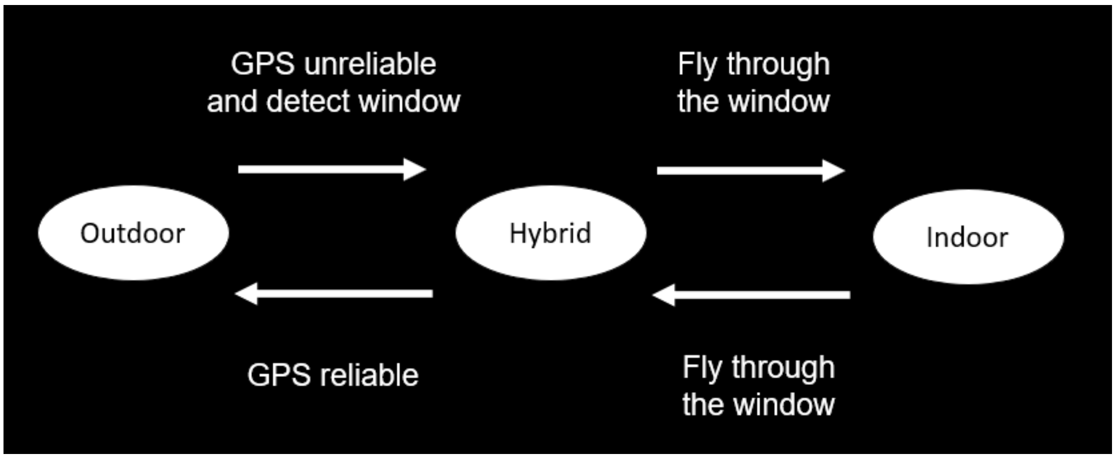

Figure 4.1: Mode switching diagram

mode switching. First, we start in an outdoor mode. We will mainly use GPS as our localizer. When the drone flies closer to the indoor environment and can detect a window using the stereo camera on the drone. It will start approaching the window and the GPS will become less and less reliable. When the GPS is unreliable, then the drone will change to hybrid mode. This mode will manage the localization part using the distance and the size of the window as a landmark similar to the approach in keyframe-based SLAM. The technical idea will be discussed in a later section. When the robot flies through the window, it will switch the mode to indoor mode. In this mode, the robot uses a laser scanner to localize itself similar to a laser-based SLAM.

When the robot decides to leave the indoor environment, it will either explore around to find a window or fly back to the original window it enters. When it detects the window and flies through it and the EKF will start becoming unreliable, the mode will switch back to hybrid mode. During this part, the drone needs to rely on EKF and the unstable GPS since it no longer detects the window behind itself. We can easily remedy this problem by setting up another camera on the back of the drone. However, flying outside briefly until the GPS signal improves is less dangerous than flying in through a window, so we omit setting up a second camera. When the GPS signal comes back, the mode will transition back to an outdoor mode. This mode-switching cycle is an overview of the whole process.

Outdoor mode

For outdoor mode, we use GPS, IMU, and a downward-facing single-point LIDAR as prime sources for localization. We use EKF to smooth and filter out the jumps in GPS. For the position data in the X-Y plane, we directly use GPS data. For height, we use the single-point laser as it is more reliable than the altitude of the GPS. For orientation data, we use both IMU and the change in GPS for the heading. To convert GPS latitude and longitude to our coordinates in the X-Y plane, first, we measure the true heading by adding the magnetic heading at the start of the experiment facing forward and the magnetic declination that we look up at our experiment location (National Centers for Environmental Information, 2018). Then, we convert GPS latitude and longitude into Universal Transverse Mercator (UTM). For each GPS point, we subtract the origin point and then transform them from the UTM reference frame to the experiment reference frame. The UTM reference frame uses true north for Y-axis, while our experiment reference frame uses a forward direction at the beginning of the experiment as the X-axis and uses the start location as an origin.

Indoor mode

For indoor mode, we directly use indoor EKF laser-based SLAM. We use three main sensors: IMU, a 2D laser scanner, and a downward-facing single-point LIDAR. We feed all the processed measurements to EKF for localization. IMU gives information about the roll, pitch, and yaw of the vehicle. We install the single-point LIDAR pointing downward from the vehicle, so it can measure the height. For the laser scanner, we use Scan Matcher. Scan Matcher is a technique for computing the change between two laser scans. In this case, we do PL-ICP on consecutive laser scans to calculate the change in the X-Y direction. With these three sensors, we manage to measure all six degrees of freedom of the drone. The EKF algorithm then will update the pose of the robot in both position and heading.

Hybrid mode

Our hybrid mode is similar to a local planner for the drone to fly through the window or small spaces. Since global localization and mapping usually suffer from the drift of the IMU over time, it is difficult to navigate the drone through the window that requires an exact maneuver. The local planner or our hybrid mode is needed to compute the current pose of the robot in the surrounding environment. For hybrid mode, we use a stereo camera, an IMU, a 2D laser scanner, and a downward-facing single-point LIDAR on the EKF algorithm. Similar to Indoor mode, we use IMU to update the roll, pitch, and yaw of the robot and use downward-facing LIDAR for measuring the height. However, we use a stereo camera and a laser scan to calculate the changes in the X-Y plane.

After detecting a window using YOLOv5 object detection on the rectified RGB images from the stereo camera, the first step is to calculate the distance and the size of the window. We can effectively detect the edges of a door or window by leveraging a depth sensor, which registers a substantial depth measurement when scanning through the doorway, whereas it detects a much smaller depth when encountering the rim of the edge. With simple geometric formulas, we can estimate the distance and the size of the window in Figure 4.2, where the angle can be calculated directly from the ratio of the object size in pixels to the image size multiplied by the field of view (FOV) of the camera and the estimated width and height from the law of cosine.

We then align the window to the previous scan to calculate the change of the robot poses in the X-Y plane respective to the robot reference frame seen in Figure 4.3. Knowing the configuration of the camera and other sensors, where the camera is mounted directly on top of the laser scanner shown in Figure 5.2, we can determine which horizontal line of the image the laser scan is measuring. Calculating this alignment is similar to Scan Matcher in Indoor mode, except we only align the edge of the window to the corresponding measurement in the laser scan. Finally, we update the change in robot pose to the EKF.

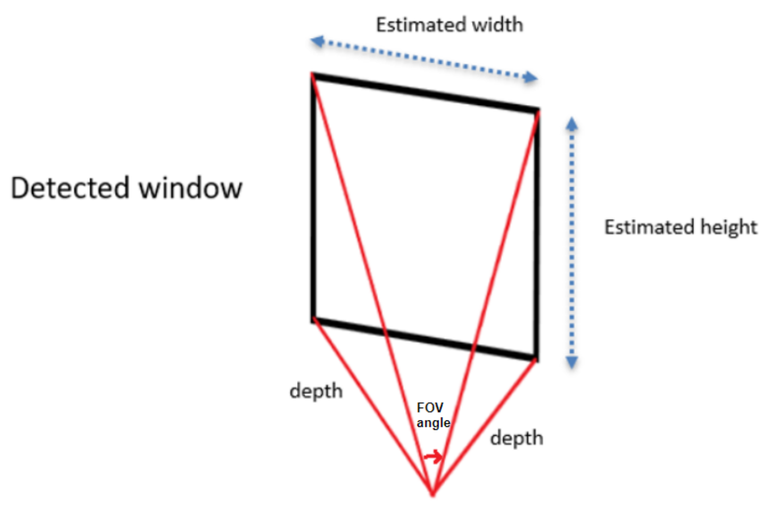

Figure 4.2: Distance from window detection

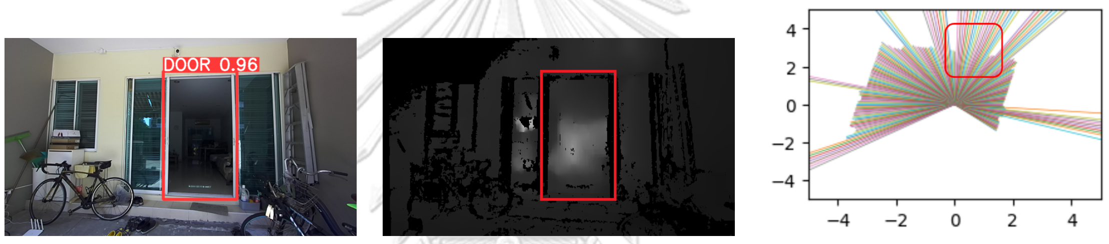

Figure 4.3: Image on the left shows a door detection on a rectified RGB image. The image in the middle shows the associated detected door on the depth registered image. The image on the right shows the associate door (labeled as a red square) in the laser scan.

4.4 Reliability

Reliability also plays an important role in mode switching. Here are our main criteria to decide if the GPS or the EKF becomes too unreliable.

For GPS

- Signal strength: The signal strength of the GPS is directly proportional to the reliability of the GPS.
- Number of satellites: We will have a minimum cutoff for the number of visible satellites. If the number of satellites goes below this number, we will classify the GPS as unreliable.

- Position covariance: We can calculate the position covariance from previous GPS data. If the covariance becomes bigger, the reliability becomes lesser.
- Reliability: Some GPSs also report data reliability. However, in our case, we use the GPS Logger application from a phone, so we do not have this data.

For EKF

- Number of laser scans: EKF depends on the structure of the environment. The more structure the better. Hence, we can use the number of laser beams that fall below a certain range as an indicator of EKF reliability.
- Position covariance: Same as in GPS case.

4.5 Evaluation

To evaluate our localization performance, we will compare the trajectory from our proposed method to the other four: approximated ground truth from a ZED2 camera with optimum environmental conditions, GPS only, laser-based only, and estimated path from manual measurement based on ground markers. To compare two trajectories, we compute trajectory alignment between the two. Trajectory alignment is a least squares optimization problem where you want to minimize the summation of the difference the robot poses between two trajectories at each time step. Comparing each trajectory to the ground truth trajectory will give us insight into how well our method performs. We use the RPG Trajectory Evaluation tool from an open-source algorithm which will be explained in detail in Section 4.6.

For the estimated ground truth path from manual measurement, we manually put markers on the ground and measure the distance with measuring tape. The path can be seen in Section 5.3 of Figure 5.3. We review images in each frame of the video to determine the timestamp of important images that contain key point locations such as when we reach a bend, when we stop, and when we make a turn.

We use these timestamps to create an estimated ground truth path assuming we move with a constant velocity and a constant angular velocity.

For trajectory from ZED2, we decide to directly use a localization module from the ZED2 camera. In the experiment, we set up the environment for ZED2 to work optimally and accurately, so that we can use the trajectory as an alternative ground truth. The camera has no direct sunlight and the indoor environment is well-lighted. Please refer to Section 5.2 for more detail.

For GPS-only trajectory, we simply use all the GPS data during the experiment with the timestamps as the trajectory even when the drone is indoors and we might lose some GPS signals. We believe this trajectory will have a higher error in the indoor portion of the path.

Similar to GPS-only trajectory, we compute laser-based only trajectory from purely laser scanner, IMU, and single-point laser. We can see this trajectory as a GPS-denied trajectory. Similarly, we believe this trajectory will have a higher error in the outdoor portion of the trajectory or it might not work at all if the open space is larger than the laser range.

4.6 Open-source Algorithms

Apart from our method described previously, we use other open-source algorithms and systems.

ROS

ROS, which stands for Robot Operating System, is an open-source framework designed to facilitate the development of robotic systems (Stanford Artificial Intelligence Laboratory et al.). It provides a collection of software libraries, tools, and conventions that enable developers to create robust and modular robot applications. ROS has a node-based architecture. It adopts a distributed architecture where in-

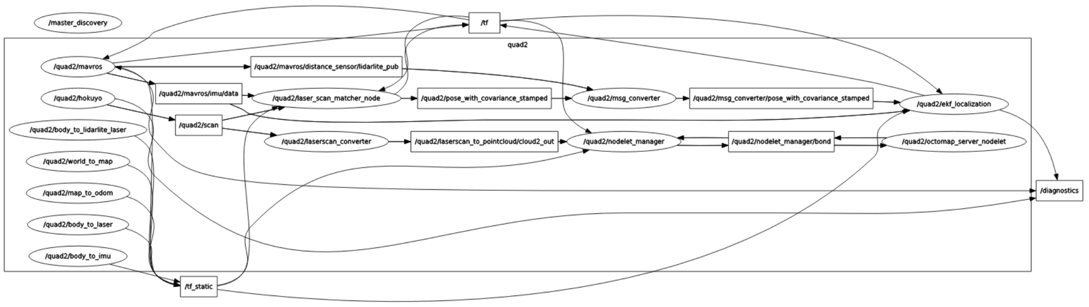
/master_discovery
/tf
quad2
/quad2/mavros
/quad2/mavros/distance_sensor/lidarlite_pub
/quad2/mavros/imu/data
/quad2/hokuyo
/quad2/laser_scan_matcher_node
/quad2/pose_with_covariance_stamped
/quad2/msg_converter
/quad2/msg_converter/pose_with_covariance_stamped
/quad2/ekf_localization
/quad2/body_to_lidarlite_laser
/quad2/scan
/quad2/laserscan_converter
/quad2/laserscan_to_pointcloud/cloud2_out
/quad2/nodelet_manager
/quad2/nodelet_manager/bond
/quad2/octomap_server_nodelet
/quad2/world_to_map
/diagnostics
/quad2/map_to_odom
/quad2/body_to_laser
/quad2/body_to_imu
/tf_static

Figure 4.4: ROS nodes graph of our system

dividual software modules, called nodes, communicate with each other by passing messages. The message passing is the main communication between nodes based on a publish-subscribe model. Nodes can publish messages on a specific topic; other interested nodes can subscribe to receive those messages. This loose coupling enables decoupled and scalable systems suitable for robotic operating systems. Figure 4.4 shows our ROS node without its external sensors.

ROS also organizes software into packages, which are self-contained units that can be easily shared and reused. Packages can contain nodes, libraries, datasets, configuration files, and other resources needed for a specific robot application. ROS provides tools for managing packages, such as building, installing, and versioning. It also provides a rich set of libraries and tools that simplify the development of robotic systems. These include libraries for handling sensor data, robot kinematics, motion planning, perception, visualization, and more. Additionally, ROS has a graphical user interface (GUI) tool called RViz, which we use to visualize sensor data, robot models, and localization and mapping.

Robot localization module

The Robot Localization module (`http://wiki.ros.org/robot_localization`) is a software package within ROS that provides state estimation capabilities for robotic systems for robot localization tasks, determining the robot's position and orientation within its environment. The module utilizes various sensor measurements, such as odometry, GPS, IMU, and other sensor data, to estimate the robot's pose accurately. It combines these sensor inputs using filtering techniques, such as Extended Kalman Filters (EKFs) and Unscented Kalman Filters (UKFs), to generate an optimized estimate of the robot's state. It also uses keyframe-based for processing each scan helping with drifting problem. In our thesis, we use the EKF filtering technique.

YOLOv5

YOLOv5 is an open-source object detection algorithm developed by Ultralytics (Jocher, 2020). It is an evolution of the YOLO (You Only Look Once) model family, known for its real-time object detection capabilities. It builds upon the previous versions by introducing several improvements in terms of accuracy and speed. The YOLOv5 algorithm follows a single-stage object detection approach, meaning it directly predicts bounding boxes and class probabilities in a single pass over the input image. This makes it faster than two-stage detectors like Faster R-CNN, which perform region proposal and object classification in separate steps.

YOLOv5 employs a more streamlined architecture compared to its predecessors. It uses a variant of the Darknet as a backbone network along with feature pyramid networks (FPN) to capture multi-scale information and enhance detection performance. Similar to modern models, it implements a technique called ”model scaling” that allows users to trade-off between accuracy and inference speed by adjusting the model size. The larger the model, the more parameter and the longer the inference time it takes. It also improves detection accuracy by introducing focal loss, which focuses training on unbalance datasets. The model also incorporates various data augmentation techniques during training, such as mosaic augmentation, random-sized cropping, and color distortion. As a result, it achieves better performance and faster inference times compared to the previous model suitable for our work. For this work, we decide to use model size S (small) for fast inference time.

RPG Trajectory Evaluation

Trajectory alignment is a process of comparing and evaluating the similarity between two or more robot trajectories in the field of robotics. We use an open-source tool from the University of Zurich's Robotics and Perception Group (UZH-RPG) (Zhang and Scaramuzza, 2018). The repository provides tools and metrics for both trajectory alignment and evaluation. It offers methods to align trajectories by finding the best translation and rotation that minimizes the difference between them relying on least-squares optimization and iterative closest point (ICP) algorithms. It measures absolute and relative trajectory errors, position errors, rotation errors, scale drift, and other statistical measurements. This alignment process is crucial because it enables a fair and meaningful comparison between different trajectory estimation algorithms or ground truth trajectories which we compare in Section 5.5.

Chapter V

EXPERIMENT AND RESULT

This section shows our experiment and results. First, we will go over the hardware and sensors specification of the custom-built drone for this experiment. Then we will explain our experiment setup and finally the result. The experiment was conducted at Baan Klang Mueang Sathon-Taksin 2, Soi Suan Luang, Bang Kho, Chom Thong, Bangkok 10150 on sunny days with high GPS availability outdoors.

5.1 Hardware

The drone is a 6-axle foldable hexacopter. I manually built it from the ground up with the help of the SR-RcTech shop. The mainframe I used is carbon fiber for its tensile strength and its lightweight. The arms and landing gear are aluminum because they are low-cost and easily replaceable. The drone's general setup is shown in Figure 5.1. A list of each hardware component of the drone can be found below.

Drone Overview

1. Total Weight: 2.5 kg.
2. Wingspan : 92.71 cm.
3. Number of propellers: 6 propellers
4. Onboard computer: NUC Intel i5
5. Propeller Size: 27.94 cm. (11 inches)
6. Height: 30.48 cm.
7. Motor: Gartt Motor ML3508 580kV
8. Speed: Hobbywing 40A, 2-6S Lipo

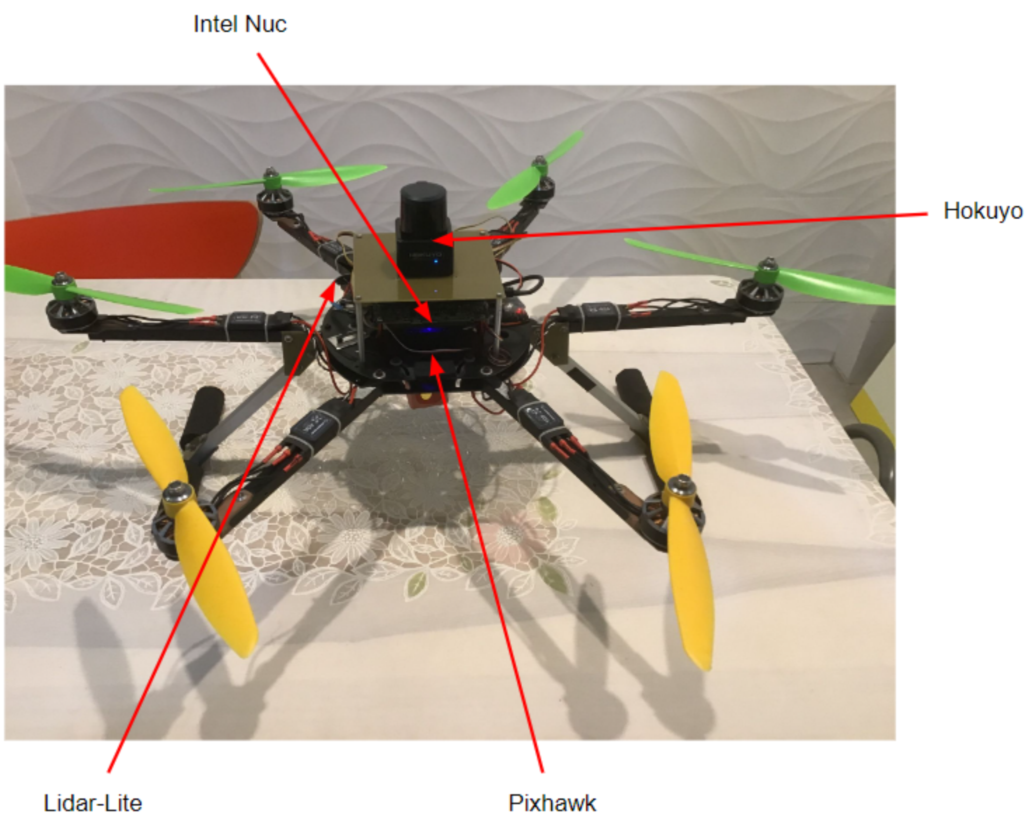

Figure 5.1: The drone's general setup

9. Battery: Dupu 2600 mAh, 14.8V

5.2 Sensors

We have three onboard sensors on the drone and two external sensors that we used specifically for this experiment. The three onboard sensors are IMU, single point laser, and a planar laser. The two external sensors are a stereo camera and GPS. Specifications of each sensor are specified below.

IMU

Inertial Measurement Unit or IMU is a combination of an accelerometer and gyroscope. It measures both the acceleration and angular velocity of the robot. We directly use the IMU from Pixhawk which is an auto-pilot module that we only use for sending PWM commands to motors. The Pixhawk is at the center of the drone shown in Figure 5.1. The IMU on Pixhawk is MPU-6000 giving a reading at 50 Hz. The sensitivity error of acceleration measurement is $\pm 3\%$ while the sensitivity error of angular velocity is $\pm 2\%$. For the EKF algorithm, we set the process noise variance for acceleration on X and Y-axis to 0.1 m^2/s^4 and on Z-axis to 0.015 m^2/s^4 and for angular velocity on Roll and Pitch-axis to 0.025 $1/s^2$ and on Yaw-axis to 0.04 $1/s^2$. We assume no correlation between different variables.

Hokuyo

The planar laser that we use is Hokuyo UST 20LX. It's range is 30 m with an accuracy of ±40 mm and it operates at 40 Hz. It has a scan angle of 270 degrees with a resolution of 0.25 degrees. We set the Hokuyo facing the same direction as the drone's forward, so its 90-degree blind spot is directly behind the drone shown in Figure 5.1. We mainly use this sensor for calculating the change in a pose in X and Y-axis using the Scan Matcher technique. The process noise variance that we set for X and Y is 0.05 m^2.

LIDAR-Lite

For height measurement, We use LIDAR-Lite v3, a single-point laser. We mounted it downward so the laser will point toward the ground in Figure 5.1. We get the measurement at 10 Hz. The range is at a maximum of 40 m and an accuracy of ± 2.5cm when the range is under 5 meters and an accuracy of ± 10cm when it is over 5 meters. The resolution of the reading is ± 1cm%. Since our operating height for the experiment is always under 5 meters, we set process noise variance for Z to be relatively small at 0.05 m^2.

GPS

Global Positioning System (GPS) is the most prevalent Global Navigation Satellite System (GNNS) today, so we decide to use an already available external source GPS from my Redmi 5G Android phone. We attached the phone on the right arm of the drone to log GPS data through GPS Logger, a free application on Google Play Store shown in Figure 5.2. Another reason is that we do not want to modify the drone too much as it is also needed for another project as well as the fact that we are already required to use an external camera which is explained in the next subsection. Even though the data we gathered from this application is at 1 Hz, we also receive the number of satellites available. We use this number to determine when the drone enters a GPS-unstable zone when the number reaches two fixed points. We determine these exact points by recording the number of satellites present in each of the three zones for five minutes varying between standing still and moving. Then, we find the best cut between each zone by modeling the number of satellites in each zone as a separated Gaussian distribution.

ZED2

For object detection, we also use an already available stereo camera ZED2. It offers RGB and depth (RGBD) images as well as other various measurements, e.g., real-time synchronized inertial, elevation, and magnetic field. It operates at 15 Hz and has a field of view (FOV) of 110° x 70°. ZED2 also provides its precise localization which we will only use as an alternative ground truth for comparison in this experiment. For our algorithm, we will only use its RGBD images for object detection. It is important to note that this sensor by itself might already seem to solve our indoor and outdoor localization because but actually, it does not. The sensor does not perform well in direct sunlight or low-light environments. In contrast, our GPS is unaffected by sunlight and our laser can work in a completely dark environment. We set up our experiment environment to be in a way that ZED2 will work optimally and accurately so that we can use both its RGBD images as input and its localization as alternative ground truth.

There is another challenge working with ZEDS. It requires NVIDIA graphic cards for it to work which our onboard computer does not have. Therefore, we need

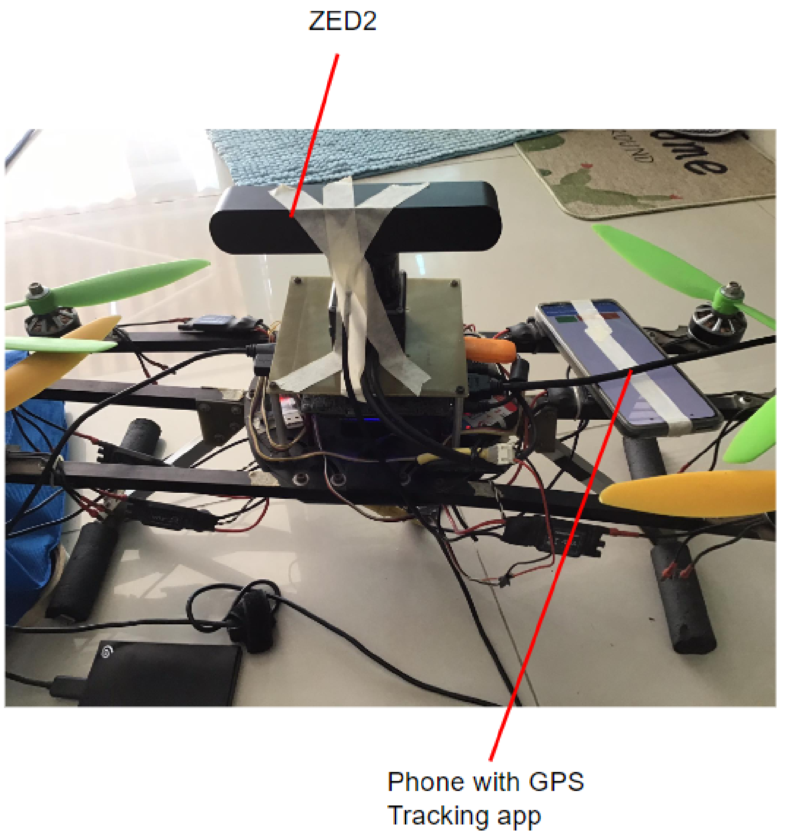

Figure 5.2: Drone setup with external sensors of ZED2 and a GPS on a phone

an external laptop with NVIDIA graphic card for running the sensor. We attached the sensor on top of the drone directly above Hokuyo so there is no obstruction in the view of the stereo camera shown in Figure 5.2. Then, we connect it to the external laptop that we carry along during the experiment. We log the sensor data in the external laptop and then combine all the logs from the onboard computer, external laptop, and my phone before processing the data using the timestamp in each log entry. Hence, there might be some error introduced in this experiment contributing to the time synchronization of the sensor.

5.3 Experiment Setup

We experiment at Baan Klang Mueang Sathon-Taksin 2 and we select sunny days with high GPS availability outdoors. Our objective of this experiment is to prove that our method can be used to safely localize a robot when transitioning between indoor and outdoor environments if we can detect a gateway between the two environments and use it as an anchor for the robot. The drone setup with external sensors is shown in Figure 5.2. We hand-carry the drone for the experiment as we cannot fly it due to having two external sensors as well as the need to carry an external laptop.

The experiment path is relatively simple and is outlined in Figure 5.3. The path starts outdoors then walk indoors and comes back out on the same path. We start from the Start marker outdoors and walk forward along the ground markers for 14.6 meters. Then, we turn left 90 degrees and continually walk forward for another 15.2 meters passing the door at the 9.4 meters mark. Finally, we turn 180 degrees and walk back along the same path stopping at the Start marker. We put markers on the ground along the predefined path every two meters outdoors and every one meter indoors manually measuring the distance between markers with measurement tape with examples in Figure 5.4. An example video of the experiment setup can be found at `https://drive.google.com/drive/folders/1E-6_Sj9hDVjgmg2AAtAbkFFFuWMaCCzo?usp=sharing`. For the indoor path, We also use the tile line so that our estimated path in the result comparison is even more accurate. The GPS unreliable zone is the area right in front of the house and the area inside right next to the door.

Since the laser scanner that we use is Hokuyo UST 20LX and it has a range of 30 meters, but the experiment location does not have that much range of open space, we decide to limit the range of the laser scanner. If the laser range is over 5 meters, we will report the max range for that laser indicating that the laser does not hit anything.

We conduct the same experiment a total of five times. Each time, we record all the data using rosbag, a logging package available in ROS, for onboard computers and also for external laptops while the GPS data on my phone is recorded from the

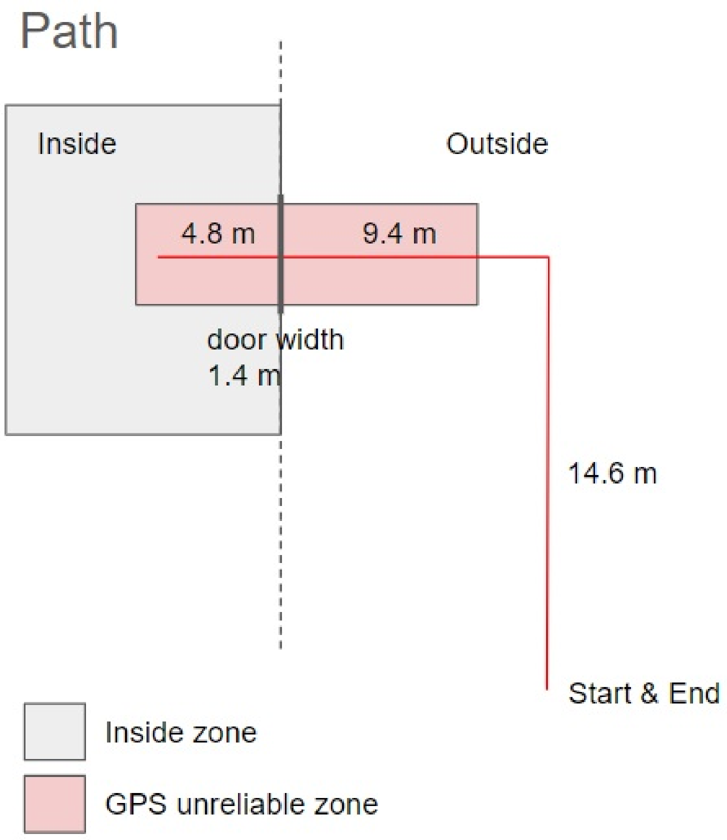

Figure 5.3: Experiment setup path

GPS Logger application written to a text file. Then, we combine all data sources and process them together to compute the localization path with our method. We have five localization paths for comparison: our method, estimated ground truth, GPS only, Laser-based only, and finally estimated ground truth from the predefined path.

5.4 Door Detection

We use the YOLOv5 object detection model to detect a door which from our assumption represents a gateway between indoor and outdoor environments (Jocher, 2020). To train the model, we collected a total of 500 RGB images of the door and the environment of our experiment from multiple perspectives using our ZED2 stereo camera on the same day as the experiment. The image size is 640x360 pix-

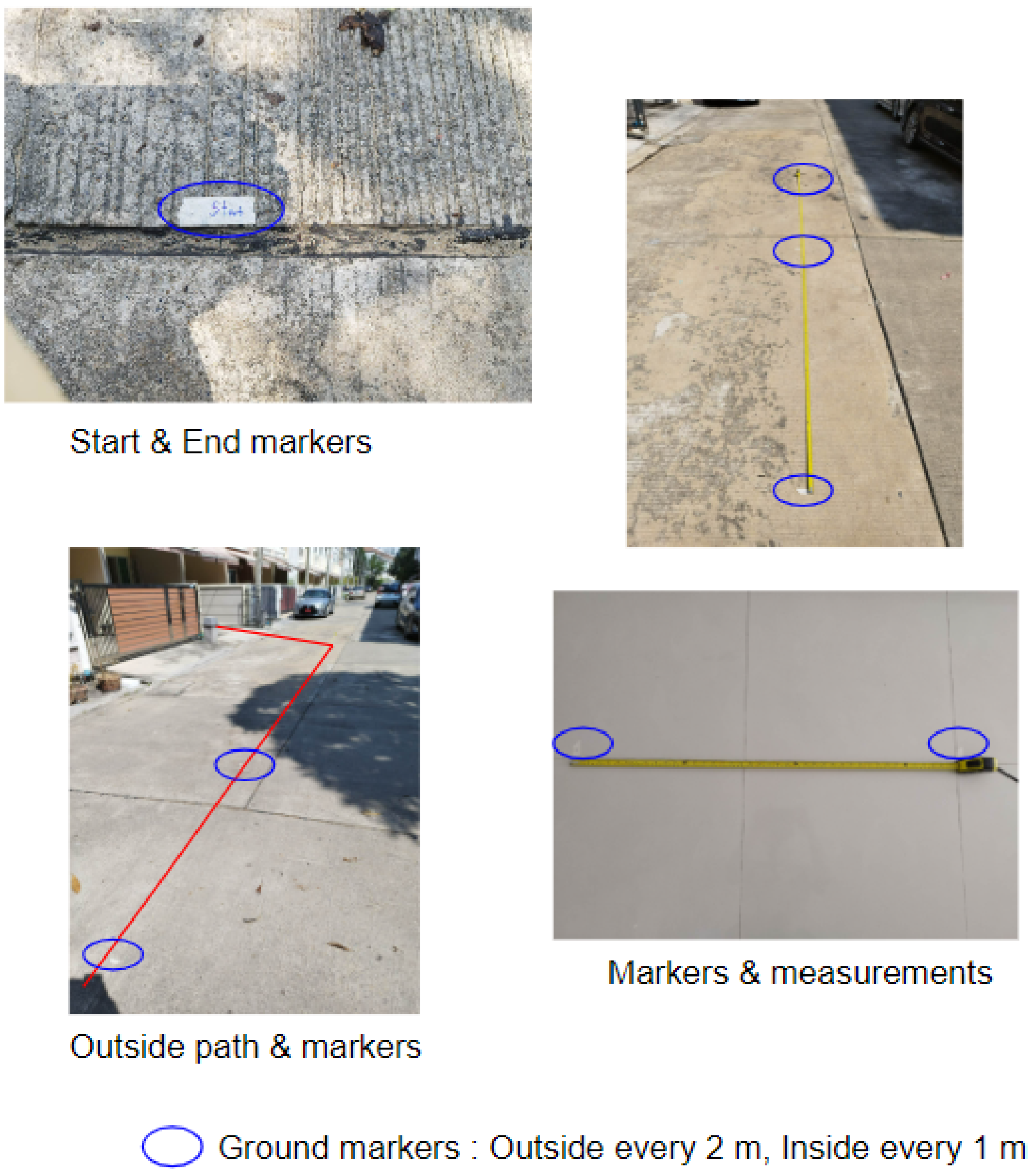

Figure 5.4: Ground markers

els. All the images we use have been rectified internally into the common reference frame from the left and the right camera images. In our dataset, there are a total of 174 images containing the door that partition between indoor and outdoor environment while the rest are images of the environment. We ignore all other windows and doors. We split the dataset into a train set, a validation set, and a test set at a ratio of 70:25:5 respectively. We use an open-source Label Studio tool for labeling the data (Tkachenko et al., 2020-2022). Figure 5.5 shows some of our training images.

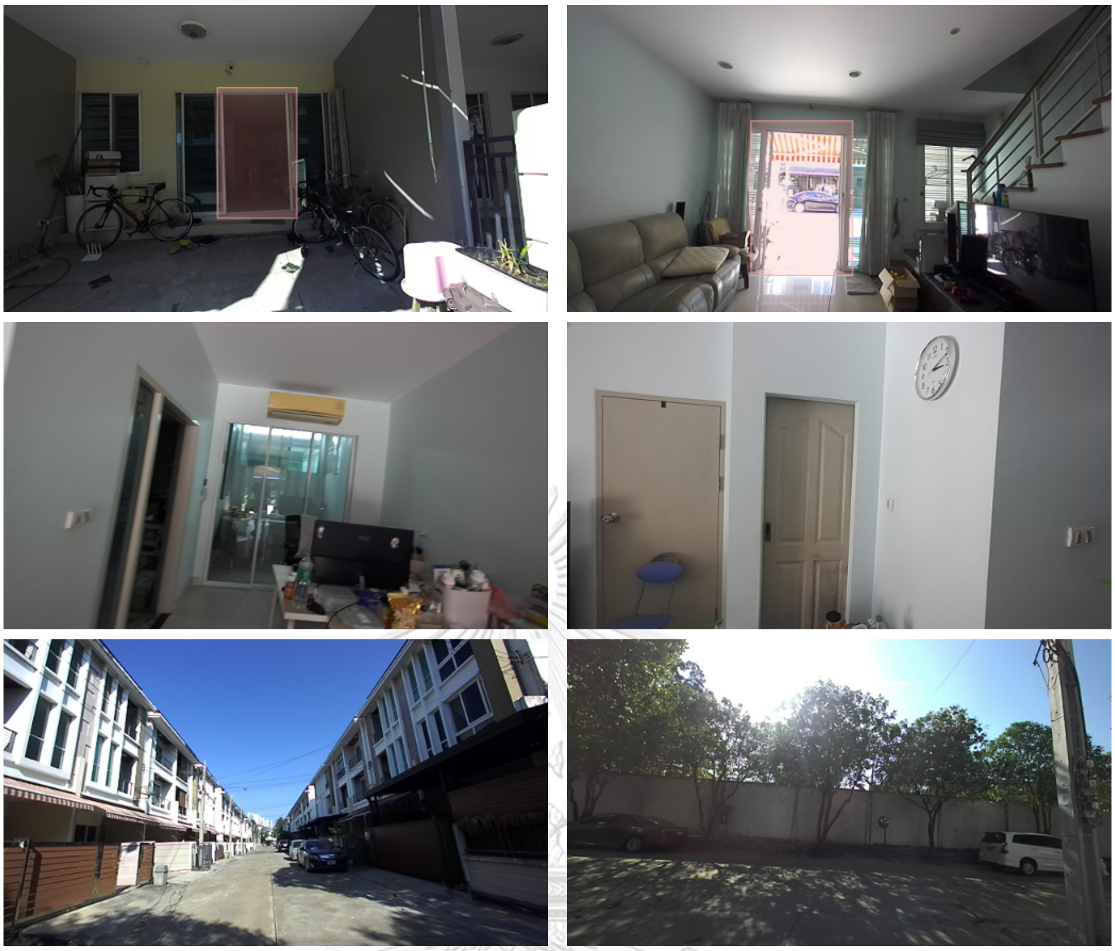

Figure 5.5: Example images that we use for training the YOLOv5 model collected from the ZED2 camera. The top two images contain our interested door; the pink bounding boxes are the door labels. The middle two images are environment images with doors that we ignore. The bottom two images are environment images.

We use YOLOv5 size S (small) as our architecture and use their default hyperparameters for training. We train the model using AWS (Amazon Web Service) on g4dn.12xlarge multi-GPU instance type from Amazon EC2 (Amazon Elastic Compute Cloud) and we train it for a total of 700 epochs in under 30 minutes. The model weight file and its train result can be seen in `https://drive.google.com/drive/folders/1cs0OfxypP47wowqG7WrW3MRE2yVLnNhz?usp=sharing`. From the result, we can see that it correctly detects our interested door with almost 100 percent accuracy in both the validation set and the test set, which implies that the model might overfit the dataset. Therefore, this model might not generalize well in our test experiment. However, when we use this model to predict each frame in our experiment videos layout in Section 5.3, it turns out to perform extremely well.

In the five videos of our experiment, we have a total of 13911 images. We use our model to predict each image using only CPU computation from an Intel(R) Core(TM) i5-8300H laptop. It achieves a runtime of five frames per second which is still not suitable for a real-time flight but we can increase the inference time with GPU acceleration as well as letting it only processes some frames instead of every frame. Nevertheless, trying to achieve an object detection module with a high frame rate is not in the scope of this work, so we simply use this model to process the data offline with our current laptop. The videos of our prediction can be viewed in `https://drive.google.com/drive/folders/1f_FlEHHUUDDOc_oAIMFqBHtbAJ5IjZQx?usp=sharing`. Some example experiment images can be seen in Figure 5.6. We can see that there are fewer than ten false positives and false negatives in all videos. From this result, we decide to use this model for this work. Please note that this model is likely to be overfitted to only our interested door, so for any other doors, windows, or entrances, the model will not recognize them. We will discuss how to combat this problem in future work in Section 6.2.

5.5 Result

Trajectories

This section presents a comparison of four different methods against estimated ground truth from manual measurement. We experimented a total of five times. Our experiment reference frame is that the X-axis is a forward direction at the beginning of the experiment and the origin is at the start marker. Figure 5.7 shows trajectories from each experiment and from each method. For GPS-only trajectory we convert latitude and longitude in a similar way to the outdoor mode layout in Section 4.3.

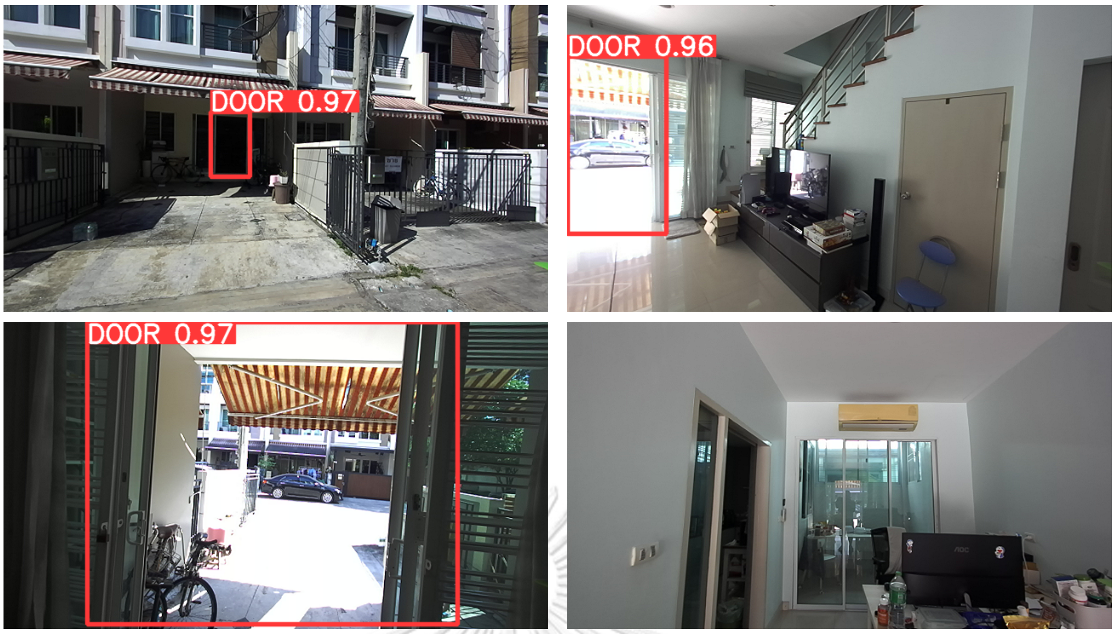

Figure 5.6: Example images from the experiment. The first three images show how the model can correctly detect our interested door. The last image shows that the model can ignore other doors. In each image, the small green blobs on the bottom right corner of the image are the propeller of the drone and it does not affect detection accuracy.

For the Laser-based only trajectory, since the range of the laser scanner is 30 meters and our experiment location does not have that much open space, we limit the laser scan range to 5 meters. From this limitation, there is a big drift in the outdoor part of the trajectory since the Scan Matcher algorithm does not when there are a few structures in the environment. For cleaner comparison, we omit this drift from the trajectory and start the trajectory right before the indoor part. We rotate and translate the trajectory so that the indoor location are aligned.

We can see that our method performs better in the outdoor part of the trajectory than that of the laser-based method as the laser-based does not have enough structure for the Scan Matcher algorithm and it cannot create a sensible outdoor trajectory. Similarly, our method is more accurate than the GPS-only method in the indoor part as the GPS signal is less accurate indoors. When compared with the manually estimated path and ZED2 path with optimum operation conditions, our result performs worse but the result is relatively similar given that ZED2 will not

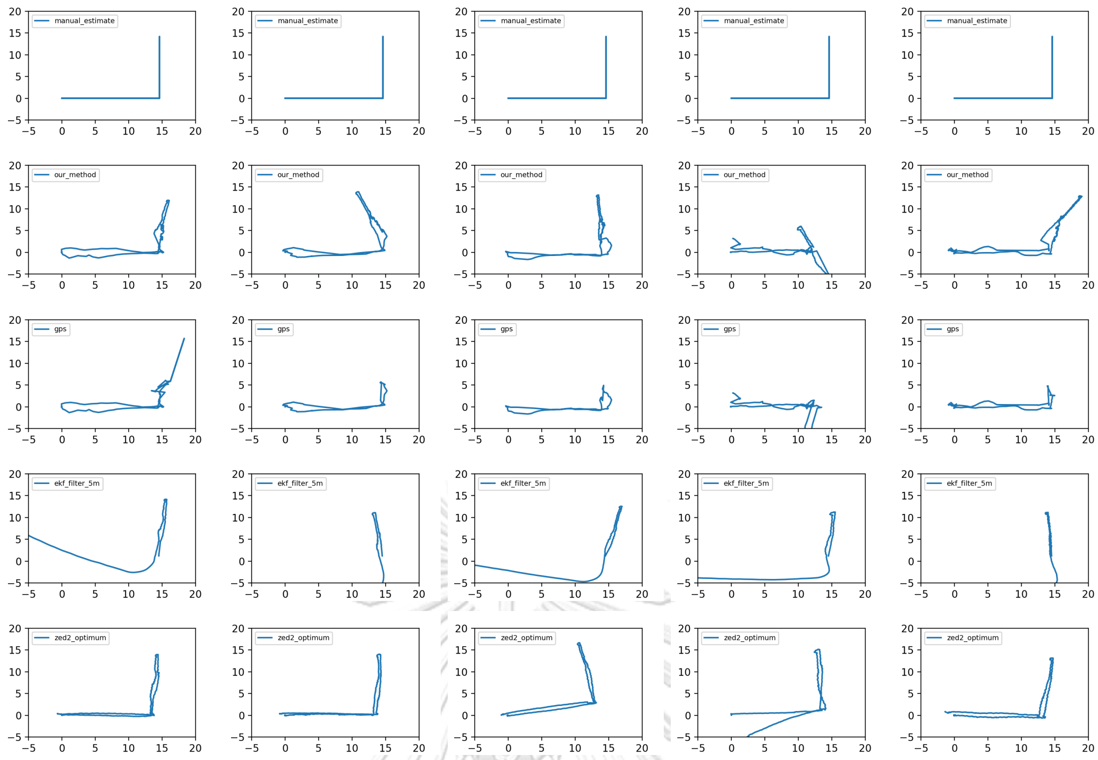

Figure 5.7: List of figures showing trajectories from each experiment (column) and each method (row). Columns from left to right are test number one to test number five. Rows from top to bottom are estimated ground truth from manual measurement, our method, GPS only, Laser-based only, and finally ZED2 optimum condition.

work in the dark or have direct sunlight, but our method can. Hence, our method still gives a good localizer for a robot in an indoor and outdoor environment.

Problems with our method

Figure 5.8 shows a better resolution of our method trajectories in the five experiments. There are three problems with our method. First, in experiment number four, the path right after GPS mode is wrong as it almost does not have a hybrid mode. The problem comes from the fact that by counting from the number of satellites the GPS receives the GPS signals report that they are reliable until the drone almost reaches the threshold of the doorway. Therefore, the trajectory is only a GPS-only path and a Laser-based only path combined. To solve this problem, we might need to use doorway detection to help verify GPS quality.

Second, in experiment number two and number five, the indoor path has a big error in rotation. This error arises when we switch from the hybrid mode to the indoor mode with the end of the hybrid path having a rotation error. When we start the indoor mode, the Scan Matcher algorithm will not know if the heading at the start is biased or not. To solve this problem, we need to share collected data between operation modes so that we can use keyframes from camera sensors for detecting loop closure when we come out from the building.

Third, in experiment number three and number five, there is a pose jump when switching from hybrid mode to outdoor mode shown in Figure 5.9. Since hybrid mode on the way out only relies on unstable GPS and unstable laser-based EKF, there is a drift in the trajectory, when switching to GPS mode and receiving an absolute pose update, the jump occurs. To remedy this problem, we need just need to do a loop closure at the start of the GPS pose update, and the whole drift with be accounted for. This loop closure will also alleviate our second problem.

Trajectory alignment

For the RPG Trajectory Evaluation tool, figure 5.10 depicts the comparison of trajectories of each method to estimated ground truth from the manual measurement of test number one. Table 5.1 also shows the differences as errors from the estimated ground truth trajectory using the RPG Trajectory Evaluation tool. The error that we report is the root mean square error on the trajectory alignment error with the unit in the meter. We can see that our method outperforms in every test for both GPS-only and laser-based methods as expected. However, with the problems mentions in the previous subsection, it performs worse than that of the ZED2 with optimum operating conditions.

Mode switching

Mode switching depends on GPS reliability and door detection. For GPS reliability, we decide to use the cut-off for GPS unreliable when the

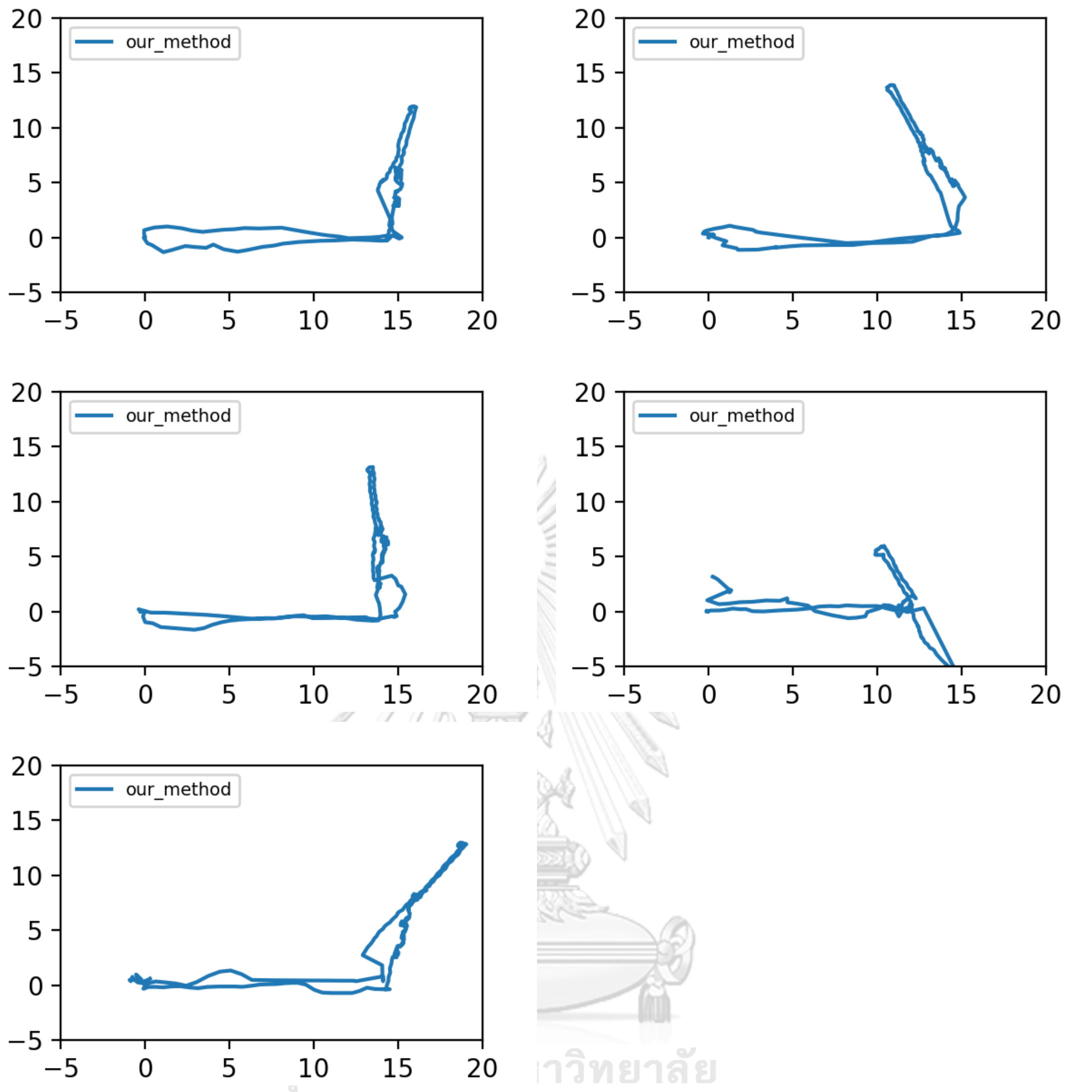

Figure 5.8: Trajectories of our method from the five experiments order from left to right and from top to bottom.

number of satellites is less than 30 satellites for three consecutive times. The GPS data can be seen in `https://drive.google.com/drive/folders/1Ib3uzv2hIX7kXRhHyBg8SwIrNMj_f9YS?usp=sharing`. Figure 5.11 shows segments of the path where the GPS is unreliable as well as the associated path (one row for the way in and another for the way out) in the estimated ground truth. We can see how in experiment number four the associated ground truth for the way in the GPS falsely reports reliability until the drone almost reaches the door. Furthermore, in all experiments, the GPS becomes unreliable closer to the building than on the way out.

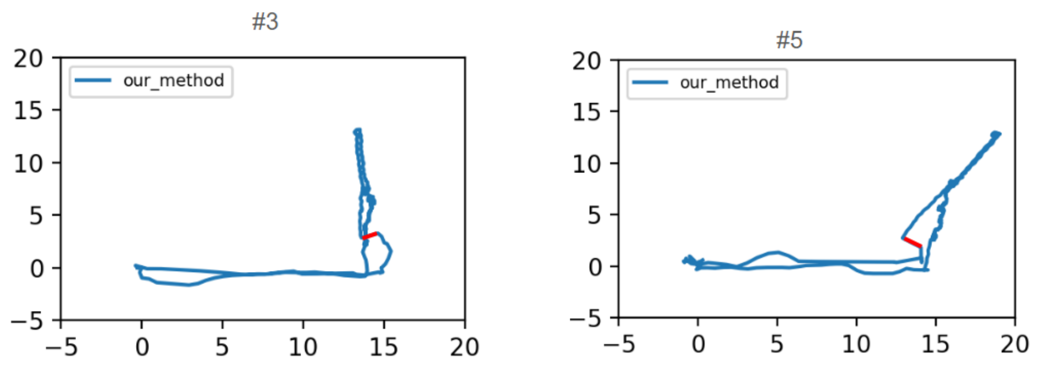

Figure 5.9: Pose jumps when switching from hybrid mode to outdoor mode indicated in red.

Our hybrid mode operation zone for the way in is the zone when the GPS is unreliable and the drone can detect the gateway which can be seen in Figure 5.12. We can see that in all experiments we can detect the door right after turning in the predefined path, so the operation mode gets switched from outdoor mode to hybrid mode only when the GPS is unreliable and switched from hybrid mode to indoor door when it can no longer detect the door.

Method \ Test	# 1	# 2	# 3	# 4	# 5
Proposed method	1.45	2.11	1.73	3.16∗	1.93
GPS only	2.53	3.39	4.32	4.21	3.88
Laser-based only	3.89	3.80	2.80	3.71	3.11
ZED2 optimum condition	0.69	0.89	0.72	1.86	0.65

Table 5.1: Comparison of four different methods against estimated ground truth from manual measurement. The unit in the table is in meters.
* Test No. 4 of our method fails because the GPS reports that it is reliable until the drone reaches the doorway when in truth the GPS is not reliable.

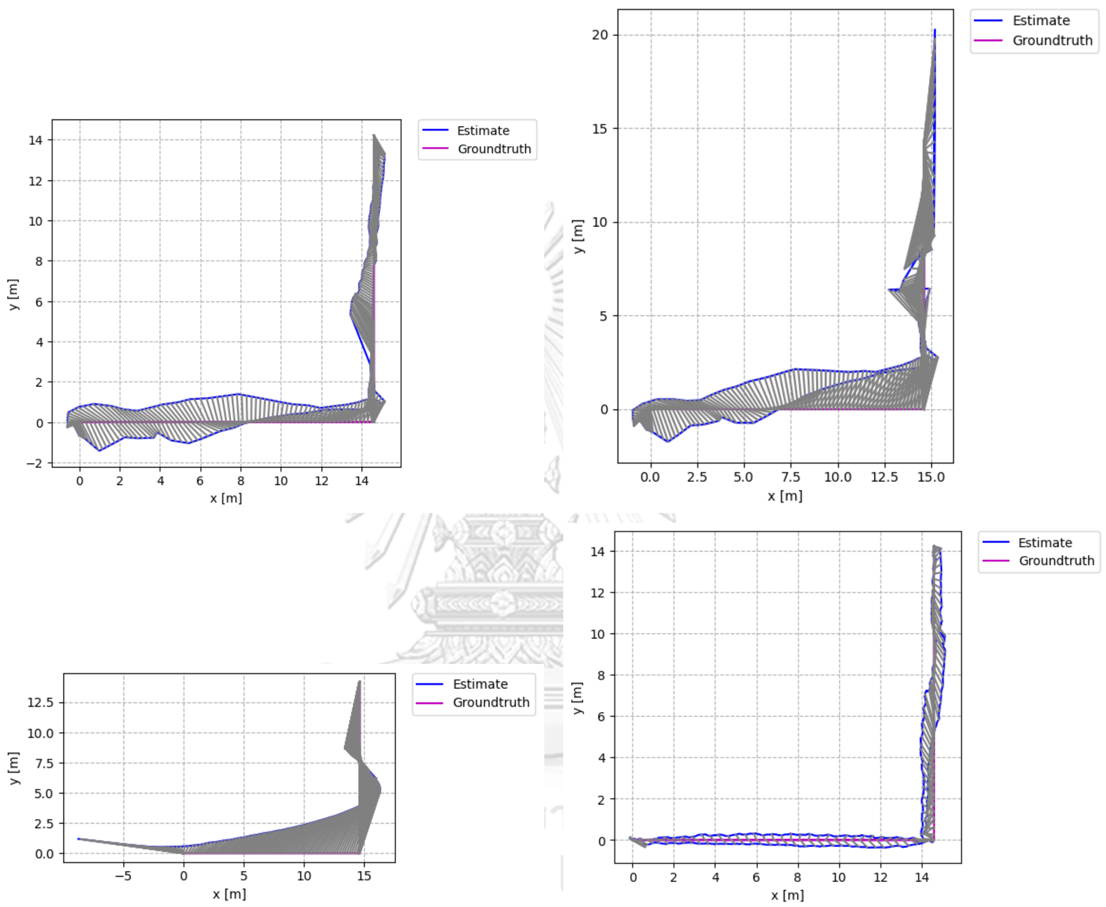

Figure 5.10: Trajectories alignment of each method again estimated ground truth from the manually estimated predefined path from experiment number one. From left to right and from top to bottom, the alignments are our method, GPS only, Laser-based only, and finally ZED2 optimum condition. The purple line is the estimated ground truth, the blue line is each method we compare, and the grey lines are matching in both trajectories according to pose and timestamp.

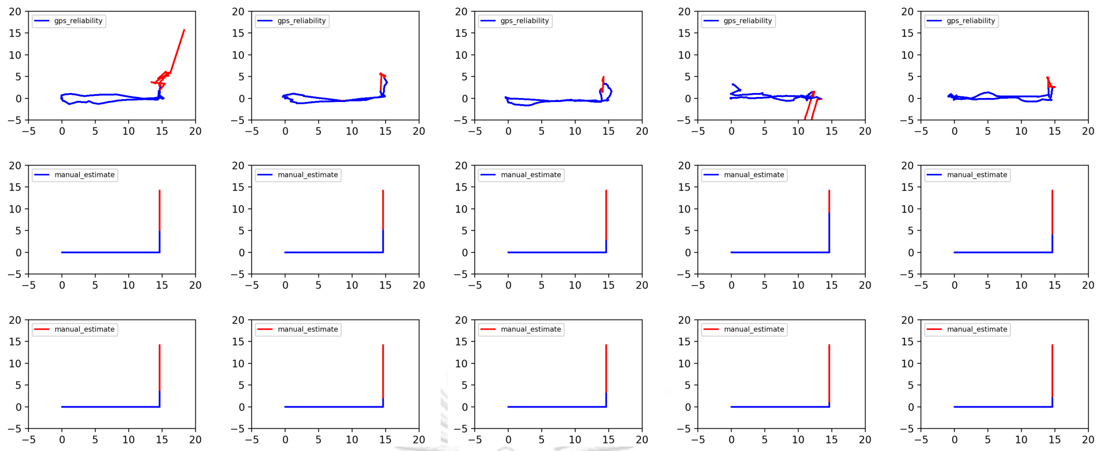

Figure 5.11: List of figures showing GPS reliability of each experiment (by columns). The top row is the GPS path. The second and third rows are associated with the ground truth path on the way in and on the way out respectively. The red indicates the GPS is unreliable.

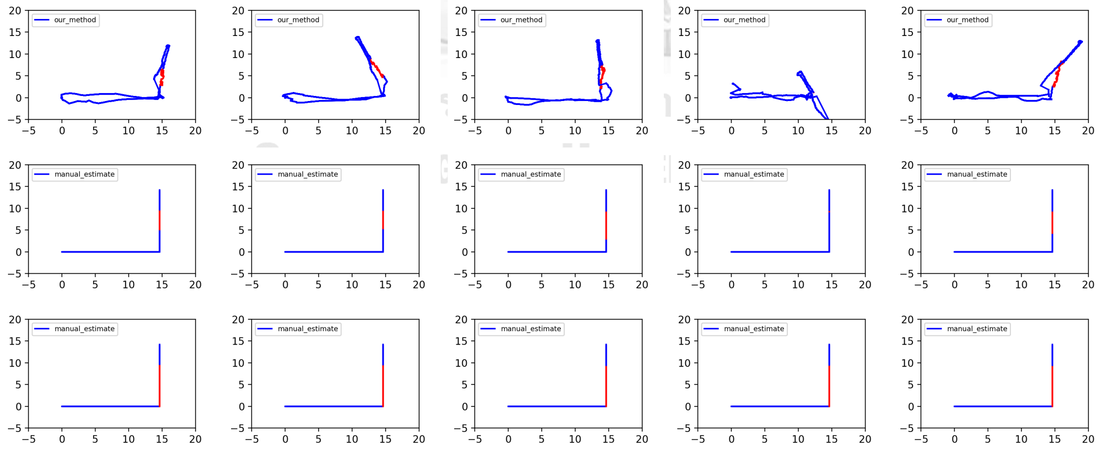

Figure 5.12: List of figures showing hybrid mode operation zone for each experiment (by columns) on the way in. The first row is our method path. The second row is the associated ground truth path on the way in. The third row is the path where object detection detects the door. The red indicates the hybrid mode operation zone.

Chapter VI

CONTRIBUTION AND DISCUSSION

6.1 Contribution and Discussion

This research provides a guideline system for autonomous UAVs to localize in such an environment with both 2D-laser scanners and cameras when only vision-based or only ranged-based fails to do so. The work aims to build a self-created hexacopter that can localize itself indoors and outdoors in an unknown environment. Even though we have not achieved a test flight due to hardware limitations, we still have proven multiple points.

Firstly, we have proven that in transitioning between outdoor and indoor environments, we can use an opening between those two environments to safely localize the robot. In the experiment, the opening is a doorway, but it can be anything like windows, archways, or even a cave entrance.

Secondly, in an area with minimal structure and an unstable GPS signal, we can find a distinguishable landmark for localization. We can use that landmark as an anchor for the robot for a short period until it finds a new anchor or move out of that environment safely.

Thirdly, this work can be further adapted and improved for numerous applications such as autonomous inspection in challenging environments like orchards, subways, and drainage systems. It could also improve search and rescue operations in thick forests and cave systems.

6.2 Future Work

In the experiment, we had to use external sensors since our onboard computer does not have an NVIDIA graphic card for processing the sensor mentioned in Section 5.2. Hence, we cannot run the whole system in real-time. Even though most of the algorithms we used processed during the experiment time, our object detection from YOLOv5 did not. Despite using a small-size model for faster inference time, YOLOv5 could potentially take too much time to process in real-time. To combat this problem, we will need to select only some frames for object detection which will result in a fewer frame rate and could affect the localization. Finding a balance between frame rate and model size for real-time processing is our next challenge.

The other problem with carrying external sensors is that we cannot fly the drone. Therefore, in our experiment, the sensors' measurement and images from the camera have less vibration than flying. We are still not certain if vibration will affect our localization method or not. Moreover, we need to find out if our trajectory alignment result from our experiment will be similar to the result from flying the drone. The speed of the drone will not be similar to walking and the drone will need to pitch forward more which might cause the laser scanner to be not usable. Nevertheless, we will leave these further studies as future work.

As we have mentioned in Section 5.4, right now our model can only detect the door in our experiment. For general use, we need to detect any arbitrary entrances between indoor and outdoor environments and if the robot can pass through those entrances safely. One simple fix is to detect all doors, windows, and entrances and then reject the ones that are unavailable from depth images. An available entrance means that the entrance has to be large enough for the robot and it must be open. We can calculate the entrance size directly from the object's size in the depth map. To detect its open state we can also check from the depth map that it must have much greater depth in the center of the object.

6.3 Open Problems

This thesis only focuses on localization tasks, but for a robot to be fully autonomous, it still needs many other systems to operate coherently. The robot needs to have a robust control system to navigate a complex environment. It also needs both local and global mapping modules as well as a navigation system. Hence, there are still many open problems to solve before we can truly have fully autonomous robots perform missions in such environments.

REFERENCES

Bachrach, A., de Winter, A., He, R., Hemann, G., Prentice, S., and Roy, N. 2010. Range - robust autonomous navigation in gps-denied environments. In 2010 IEEE International Conference on Robotics and Automation, pp. 1096–1097. :

Bertran, E. and Sànchez-Cerdà, A. 2016. On the tradeoff between electrical power consumption and flight performance in fixed-wing uav autopilots. IEEE Transactions on Vehicular Technology 65.11 (2016): 8832–8840.

Besl, P. and McKay, N. D. 1992. A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14.2 (1992): 239–256.

Censi, A. 2008. An icp variant using a point-to-line metric. In 2008 IEEE International Conference on Robotics and Automation, pp. 19–25. :

Davison. 2003. Real-time simultaneous localisation and mapping with a single camera. In Proceedings Ninth IEEE International Conference on Computer Vision, pp. 1403–1410 vol.2. :

Diosi, A. and Kleeman, L. 2005. Laser scan matching in polar coordinates with application to slam. In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3317–3322. :

Dowling, L., Poblete, T., Hook, I., Tang, H., Tan, Y., Glenn, W., and Unnithan, R. R. 2018. Accurate indoor mapping using an autonomous unmanned aerial vehicle (uav). arXiv preprint arXiv:1808.01940 (2018):

Horn, B. K. and Schunck, B. G. 1981. Determining optical flow. Artificial Intelligence 17.1 (1981): 185–203.

Huang, G. P., Mourikis, A. I., and Roumeliotis, S. I. 2009. On the complexity and consistency of ukf-based slam. In 2009 IEEE International Conference on Robotics and Automation, pp. 4401–4408. :

Jocher, G. 2020. YOLOv5 by Ultralytics [Online]. Available from: https://github.com/ultralytics/yolov5 [2020,May].

Klein, G. and Murray, D. 2007. Parallel tracking and mapping for small ar workspaces. In 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234. :

Lingemann, K., Surmann, H., Nuchter, A., and Hertzberg, J. 2004. Indoor and outdoor localization for fast mobile robots. pp. 2185 – 2190 vol.3. :

Moravec, H. and Elfes, A. 1985. High resolution maps from wide angle sonar. In Proceedings. 1985 IEEE International Conference on Robotics and Automation, volume 2, pp. 116–121. :

Mur-Artal, R. and Tardos, J. 2014. Orb-slam: Tracking and mapping recognizable features. :

National Centers for Environmental Information, N. 2018. Ncei geomagnetic calculators [Online]. Available from: https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml [2018,Dec].

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. 2011. Dtam: Dense tracking and mapping in real-time. In 2011 International Conference on Computer Vision, pp. 2320–2327. :

Newman, P., Cole, D., and Ho, K. 2006. Outdoor slam using visual appearance and laser ranging. In Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pp. 1180–1187. :

Nuchter, A., Lingemann, K., Hertzberg, J., and Surmann, H. 2006. 6d slam - 3d mapping outdoor environments. Fraunhofer IAIS 24 (11 2006):

Stanford Artificial Intelligence Laboratory et al. Robotic operating system [Online]. Available from: https://www.ros.org [,].

Thrun, S., Burgard, W., and Fox, D. 2005. Probabilistic robotics. MIT Press, Cambridge, Mass. ISBN 0262201623 9780262201629. Available from: http://www.amazon.de/gp/product/0262201623/102-8479661-9831324?v=glance&n=283155&n=507846&s=books&v=glance .

Tkachenko, M., Malyuk, M., Holmanyuk, A., and Liubimov, N. 2020-2022. Label Studio: Data labeling software [Online]. Available from: https://github.com/heartexlabs/label-studio [2020-2022,].

Urcola, P., Lorente, M., Villarroel, J., and Montano, L. 2014. Seamless Indoor-Outdoor Robust Localization for Robots, volume 253, pp. 275–287. ISBN 9783319036526. doi: 10.1007/978-3-319-03653-3_21.

Yang, S., Scherer, S., Yi, X., and Zell, A. 2017. Multi-camera visual slam for autonomous navigation of micro aerial vehicles. Robotics and Autonomous Systems 93 (03 2017):

Zhang, Z. and Scaramuzza, D. 2018. A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7244–7251. :

Biography

Laphonchai Jirachuphun was born in Bangkok on 12 July 1992. For his elementary school, he studied at Stree Manda Pitak School, Chanthaburi. Then, he graduated from Patumwan Demonstration School in 2008 with a lower secondary school diploma and from Triam Udom Suksa School with an upper secondary school diploma in 2011. In the same year, he won a bronze medal at the 23rd International Olympiad in Informatics was held in Pattaya, Thailand. After that, he went to the Massachusetts Institute of Technology, Cambridge, United States of America to study Computer Science and Computer Engineering. He complete his bachelor's degree in 2016.