Navigation of Indoor Spaces Using Multiple Quadrotors

Brian M. Katona

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Aeronautics and Astronautics

University of Washington

2018

Reading Committee:

Kristi Morgansen, Chair

Christopher Lum

Program Authorized to Offer Degree:
Aeronautics and Astronautics

University of Washington

Abstract

Navigation of Indoor Spaces Using Multiple Quadrotors

Brian M. Katona

Chair of the Supervisory Committee:
Professor Kristi Morgansen
Department of Aeronautics and Astronautics

The operation of uninhabited aerial vehicles (UAVs) indoors faces many challenges, notably from the ineffectiveness of the sensors typically used on these vehicles for flight in outdoor environments, such as Global Navigation Satellite Systems (GNSS) and magnetometers. However, there are many situations when the ability to use a UAV indoors is desirable, such as for surveying underground infrastructure, performing inventory of a warehouse or assisting in search-and-rescue operations. This thesis addresses three challenges facing the autonomous indoor flight of UAVs. First, it proposes a platform for developing a low-cost UAV system using commercial-off-the-shelf hardware and open-source software. The system consists of one or more quadrotors, an ultrasonic beacon system and a ground control station. Next, it surveys methods for measuring the yaw of a vehicle in an indoor environment, including the magnetometer and computer vision techniques The thesis then proposes a method for using an ultrasonic beacon system to measure yaw. The advantages and disadvantages of each method are noted. Finally, the thesis introduces a technique for coordinating a network of multiple vehicles to provide localization to a vehicle performing a mission by moving the reference beacons of the ultrasonic beacon system throughout the space. The algorithm for choosing waypoints for each vehicle is described, and simulations for the algorithms are presented for multiple cases.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This thesis would not have been possible without the assistance of my advisor, Kristi Morgansen, and my committee member Chris Lum. I would like to express my gratitude to Unsik Lee and John Berg for all of their collaborations on this project. I would like to thank the University of Washington College of Engineering Dean's Endowed Fellowship, the Henry L. Gray Memorial Fellowship, the Louis and Katherine Marsh Fellowship and the Joint Center for Aerospace Technology Innovation for financial support.

Many thanks go to my family for their constant support in everything I do. I would like to thank my friends, both those from the University of Washington and elsewhere, for their kindness and companionship. I am grateful for the other members of the Nonlinear Dynamics and Controls Lab, for making the lab a great place to study and work. I would like to especially thank Itzue Caviedes Solis for being constantly supportive and enduringly caring, and for being the best.

Chapter 1

INTRODUCTION

1.1 Motivation

Since the beginning of the spacefaring era, the governments of multiple countries recognized the civilian, commercial and military benefits of a space-based navigation system. These space-based navigation systems, called Global Navigation Satellite Systems, or GNSS, allow for a vehicle, person or object to be located nearly anywhere on Earth. This facilitates navigation for all types of vehicles, such as automobiles, trucks, ships and airplanes, and tracking of all types of targets, such as cargo, people and even animals. Such widely available and accurate positioning information is especially useful for the operation of autonomous vehicles, such as autonomous cars and uninhabited aerial vehicles (UAVs). GNSS position information is used on autonomous vehicles to locate the vehicle's position and guide it along a desired path to its objective destination. The use of GPS in autonomous systems has become widespread and well understood, and is used on all types of vehicles, from experimental research vehicles to vehicles used by hobbyists. Unfortunately, there are many instances in which GNSS are not effective. Most notably, GNSS signals are relatively weak and cannot penetrate the roofs and walls of buildings reliably. This means that GNSS is typically unavailable indoors. Since GNSS systems require that a line of sight be available between the receiver and the satellites, other environments that may have restricted availability of GNSS service include urban environments with tall buildings, terrain with steep changes in topography or underwater environments. Other possible situations resulting in a loss of GNSS availability includes jamming by rogue actors or operations taking place at high latitudes. These situations in

which GNSS information is not available require that other methods of localizing a vehicle be developed. Despite these challenges, many potential uses for autonomous vehicles in GNSS-denied environment exist. Some examples include surveying of tunnels or other underground infrastructure, inventory in warehouses, or search-and-rescue in disaster situations. As will be discussed in Section 1.2, many methods for performing localization in GNSS-denied environments have been developed. However, many of these methods are experimental, expensive or otherwise unsuitable for use in commercial settings. The goal of this thesis is to use open-source software and commercially-available hardware to develop a system that allows an autonomous vehicle or multiple autonomous vehicles to localize themselves and navigate effectively to accomplish a mission. This thesis addresses three of the challenges associated with indoor flight of autonomous vehicles: the integration of commercial-off-the-shelf hardware and open sources software, the measurement of yaw in a noisy magnetic environment, and coordination of multiple vehicles to reduce the indoor infrastructure needed for indoor flight.

1.2 Background

1.2.1 Vision-based Indoor Navigation

Much of the foundation of early work in indoor navigation for autonomous vehicles used vision-based methods for indoor localization. One popular vision-based method is the use of motion capture systems. Motion capture systems use multiple cameras set around the area in which an object is being tracked. Typically, the object being tracked is equipped with reflective markers that reflect light generated by the camera back to the camera, allowing the system to distinguish and track the target object. This method provides the ability to track both the position and orientation of the target. The use of these systems are widespread, for example in the work of Mellinger and Kumar [22] and Szmuk et al. [35]. Motion capture systems provide millimeter-level localization of targets at high rates. However, the systems are expensive, require the installation of large cameras, and require that the target be within line-of-sight of the camera.

In other forms of vision-based indoor navigation, the cameras are mounted on the vehicle, instead of on the perimeter of the operation area. Collectively these forms of localization and navigation are known as visual odometery. In an early example of visual odometry, Amidi et al. developed an autonomous helicopter that used a visual odometer to estimate its position in flight [3]. This helicopter's computer vision system tracked a visual target in each frame, and estimated the movement of the helicopter based on the change in translation of the camera in the ground navigation frame. Visual odometry has since branched into multiple forms. One form of on-board vision-based indoor navigation is Simultaneous Localization and Mapping (SLAM). In SLAM, one or more cameras on-board the vehicle are used to both build a point-cloud map of the operating environment of the vehicle, as well as to calculate the position and orientation of the vehicle within this constructed map. The techniques now known as SLAM were first outlined by Smith and Cheeseman [32], who described methods for estimating both the relationship and error between coordinate frames when those frames are representing the relative locations of objects. They proposed the usefulness of the technique in robotics, and it was quickly developed into localization algorithms for robotics. Many versions of the technique were developed, such as that of Montemerlo et al. [24] and Weiss et al. [34]. While developing techniques for augmented reality, Klein and Murray [18] made an advancement to the visual odometry field by creating a version of SLAM in which the mapping and localization processes were processed on separate threads. This technique, known as Parallel Tracking and Mapping, or PTAM, also differed from SLAM in its scope, as it was designed to be used in small spaces (like over a desk or table) and therefore created a more detailed, dense map. With advances in processing power, it became possible to use PTAM onboard a vehicle. Presently, SLAM and PTAM have become well developed enough that they are used on mission-ready vehicles [26] and offered as part of commercially available autopilot packages [33].

Nature has been the inspiration for another form of vision-based indoor navigation. The concept of optic flow is inspired by the processing of retinal images undertaken by biological systems, and in relation to flight, especially by insects. Humbert et al. [15] developed models for retinal

image flow and wide-field integration processing in insects, and proposed the use of these models as a basis for autonomous navigation strategies for robots. Around the same time, Ruffier and Franceschini built a proof-of-concept flying vehicle that used an optic flow system called OCTAVE [31] to autonomously navigate a circular path. Optic flow was later shown to be useful for navigation tasks as well, such as detecting collisions [11, 23] and landing [40]. Alaeddini and Morgansen proposed the use of nonlinear observability analysis to estimate the state of a vehicle using a minimal number of optic flow sensors [2]. Optic flow technology is now readily available through commercially-available and open-source optic flow sensors such as the PX4FLOW camera [14, 19]

1.2.2 Beacon-based Indoor Navigation

Localization Methods

Indoor navigation for robots using time-of-arrival and trilateration techniques have been under development at least since the late twentieth century. These localization techniques have the commonality of operating on the basis of the measurement of the time of transmission or signal strength of some type of electromagnetic or acoustic wave. Multiple beacons around an operating area send or receive signals to or from a target. The time-of-arrival of the signal is calculated and used to determine a distance, or position is determined from the relative signal strength of signals from beacons with known positions. By trilaterating the distances between the target and multiple beacons, the position of the target within the network of beacons can be determined. Some examples of the types of signals used include ultrasonic signals, ultra-wideband signals and radio signals such as WiFi or BLT signals. In 1998, Aoyagi et al. proposed an indoor mobile robot navigation system based on ultrasonic sensors. This system, called ”RECS” [4], used ceiling-mounted ultrasonic receivers to track a ground robot equipped with an ultrasonic transmitter, and claimed sub-millimeter accuracy in distance measurements. Hazas and Ward [13] attempted to solve some of the issues inherent with ultrasonic location systems by utilizing broadband transducers.

More recent work includes that of Qi and Liu [28] and Gaulda et al. [12]. Qi and Liu performed additional time synchronization of the beacons using a radio signal along with a WiFi based communication with a central server. Gaulda et al., demonstrated the ability to move through multiple ultrasonic positioning systems, some of which are calibrated and some of which or not. While the robot moves through the system, it calibrates the uncalibrated systems while still being able to effectively navigate through the environment.

Some researchers have demonstrated the utility of using WiFi signals to locate targets, including mobile robots. Bahl and Padmanabhan [5] demonstrated an early version of a local area network (LAN) system that could be used to locate users within the system. Castro et al. used signal strength from wireless networks to locate the position of devices on the network [7]. More recently, Elbasiony and Gomaa created a localization scheme for mobile robots using WiFi signal strength. They noted that while WiFi signals proliferate many buildings and environments, the propagation of the signal is unpredictable based on building geometry. As a result of this and other factors, they achieved accuracies of around one-third of a meter.

Ultra-wideband (UWB) beacon systems are another method of indoor localization that have been explored more recently. These radio frequency-based systems generally can provide a higher level of accuracy than ultrasonic systems, but at a higher cost. One example developed at the University of Michigan by Kempke et al. is PolyPoint [17], which is a UWB system based on a commercially available indoor localization system, and used to provide localization for quadrotors. In 2017, Tiemann and Wietfeld [37] introduced another UWB-based system for locating a quadrotor in an indoor space, and demonstrate the tracking performance of the quadrotor system relative to a motion capture-based system. The authors also demonstrated the system's usability in user-defined tasks, such as navigating a precise trajectory to find a package on a shelf.

One final example of localization systems used in GNSS-denied environments are the long baseline navigation systems used in underwater autonomous vehicle operations. Long baseline navigation systems provide underwater ranging and localization using acoustic signals. They in-

clude beacons that can be deployed from ships or installed on the sea floor, and perform ranging and trilateration using time of transmission of acoustic signals. One example is the Woods Hole Oceanographic Institution Micro-Modem developed by Freitag et al.[10]. In bench testing these showed an accuracy of 10 cm, and in practice an accuracy of 1.5 m. Similar systems have been been deployed in operational missions, such as the Benthos acoustic beacons used in the Artic GAkkel Vents Expedition to study sub-ice hydrothermal vents by Jakuba et al.[38].

Beacon Placement

One of the most important characteristics contributing to a beacon-based localization system's accuracy is the placement of the beacons around the operational area. The geometry of this set-up determines the quality of signal that beacon may receive, the possibilities for interference of signals and the presence of noise and echoes. Many researchers have investigated a variety of techniques that can be used to create sufficient or even optimal beacon configurations for localization systems. Matrinez and Bullo [20] proposed a sensor placement method that is optimal with respect to the Fisher Information matrix, and includes strategies for moving the beacons to an optimal placement. A noted benefit of this method was improved performance in a filtered estimate of the target's position. Jourdan and Roy developed RELOCATE [16], an algorithm that optimally places range-based beacons to minimize a metric called the Position Error Bound. Nazaraehi and Savkin also presented a method of self-deploying mobile beacons, but used a geometric approach with a proof of convergence [25]. Rajagopal et al. [30] developed a toolchain that can be used to compare beacon configurations using metrics of localizability and accuracy based on the GDOP, and demonstrated the ability to reduce the number of sensors needed using knowledge of the geometry of the operational area. One special consideration facing underwater long-baseline systems is uncertainty in the placement of the stationary beacons. Thus some methods have been developed to simultaneously estimate the position of the stationary beacons as well as the trajectory of the mobile autonomous vehicle using those beacons. Teixeira et al. as well as Olson et al. use an

Extended Kalman Filter-based example of this method, with Teixeira et al. defining an optimal vehicle trajectory for performing position estimation of the beacons' locations [36, 27]. Vaganay et al. developed and tested [39] and Curio revised and tested [8] a method for providing higher accuracy location estimates from vehicles with high accuracy navgiation systems to vehicles with low accuracy navigation systems using a moving network of long baseline equipped vehicles. Finally, Bahr et al. developed and tested [6] a method for coordinating the movement of a network of vehicles equipped with long baseline communication and navigation aids in order to provide a relative localization of all vehicles in the network.

1.3 Contributions

This thesis addresses three challenges of indoor flight of autonomous vehicles. The first is the integration of commercially available hardware and open-source software that can be used to facilitate this form of autonomous indoor navigation of quadrotors. The second is the measurement of vehicle yaw when faced with an unpredictable magnetic environment. This is addressed using a method of estimating the yaw of a quadrotor using two ultrasonic beacons. Finally, this thesis presents a novel method of using multiple quadrotors to navigate a large indoor space using ultrasonic beacons. The quadrotors carry the ultrasonic beacons throughout the space to allow for localization coverage of a large space with fewer sensors.

1.4 Thesis Organization

The second chapter of this thesis will introduce the commercially-available hardware and open-source software used in this project to allow a quadrotor to operate autonomously indoors. This chapter will cover the quadrotor used, as well as the associated autopilot software. Also covered will be the ultrasonic beacon system that was implemented with the quadrotor, as well as a computer vision system that was evaluated for use as an indoor navigation aid. The third chapter is a comparison of three methods for measuring the yaw of a quadrotor indoors. The first method

covered is the use of a magnetometer, followed by the use of a computer-vision system. Lastly, Chapter 3 presents a method for using multiple ultrasonic beacons to determine the orientation of a quadrotor. Chapter 4 introduces a method for coordinating the movement of a group of quadrotors throughout an indoor space in which the quadrotors carry ultrasonic beacons that are providing localization coverage for one quadrotor that is executing a desired mission. The final chapter will conclude the thesis and offer directions for future work.

Chapter 2

HARDWARE

The objective of the project is to be able to equip a large indoor space to be navigated at a low cost. Therefore, the project makes use of commercial-off-the-shelf (COTS) components as well as open-source software to maintain as low of a cost as possible. These COTS components are used to move through the space, locate the vehicles in the space, provide vehicle heading information and perform sensing functions.

2.1 Flight Vehicle and Flight Software

2.1.1 Intel Aero Quadrotor

The first step in developing a system to navigate a large indoor space is to choose the vehicle that will be used. In a large indoor space such as a warehouse, the vehicle must have the capacity to reach high heights and operate for a sufficient amount of time to traverse large distances. All this must be accomplished while being agile enough to navigate through shelving or aisles. Given these requirements, a quadrotor fit the necessary profile. Quadrotors are capable of flying at the heights of buildings like warehouses. Additionally, with flight times reaching twenty minutes or more, a quadrotor has sufficient flight time to navigate a large area of an indoor space. Quadorotors are also agile compared to fixed wing aircraft, allowing them to easily navigate the sharp corners of an indoor warehouse-type space.

The vehicle chosen for this project is the Intel Aero Ready to Fly Drone. The Aero is a commercially available drone developed by Intel and built around the Intel Aero Compute board. The Compute board is a developer kit designed to be used for UAVs, and allows the use of a Linux

Figure 2.1: The Intel Aero Drone includes an Intel Atom-based computer, a flight controller with open-source software, and multiple cameras.

operating system through an Intel Atom processor coupled with 4GB of RAM. In addition, the Compute board gives the vehicle access to WiFi, computer vision through multiple sensors, including RGB cameras and depth sensing, as well as input/output for external sensors and actuators. Having the ability to run a standard distribution of Linux is desirable, as it eases development and allows for the use of the Robot Operating System (ROS) [29] (see section 2.1.3) to facilitate data handling. The Aero Compute board is connected through a field-programmable gate array to a separate STM32 microcontroller which hosts the flight software.

2.1.2 PX4 Flight Software

To facilitate the quick development of the system at a low cost, an open source flight software is used. This flight software handles stabilization of the flight vehicle, waypoint navigation, telemetry and many other basic functions of the flight vehicle. The Intel Aero supports two open-source flight stacks, PX4 and Ardupilot. While both provide similar levels of basic functionality, they differ in some of the specific features that are offered. Most notably for this project, PX4 features native support for external vision-based localization such as motion capture systems or visual

odometry. Since in this case the vehicle would be operated exclusively indoors, the option to be able to use external methods for localization is necessary. Therefore, PX4 (Version 1.6.5) is the flight software used on the vehicle. PX4 was conceived out of the Computer Vision and Geometry Group at ETH Zurich [21]. It is a multithreaded autopilot framework that is currently being developed as an open-source project. It can be run on a number of hardware options, from a variety of microcontrollers to the purpose-built Pixhawk open-hardware autopilot. It supports fixed-wing, multicopter and Vertical Takeoff and Landing (VTOL) vehicles. Further expanding its capability is the ability to interface with the Robot Operating System (ROS). This allows for easy interfacing with a companion computer to execute computationally intensive tasks, while safety and time-critical tasks can continue to run in real time on a microcontroller. PX4 is now part of a full UAS software suite called Dronecode. This suite includes PX4, a corresponding communication protocol called MAVLINK and a ground control station called QGroundControl. MAVLINK is a publisher-subscriber based messaging format that allows for communication between flight controllers running PX4 and companion computers or ground control stations, or both simultaneously. Through this protocol, a flight controller using MAVLINK can send telemetry data to another computer, or receive commands from a ground control station. In addition, MAVLINK can also be used to send state information to the flight controller from an external sensing system. QGroundControl [1] is a ground control system designed to work with PX4 using MAVLINK. It allows users to completely manage a flight controller running PX4. Users of QGroundControl can set parameters, calibrate sensors, choose waypoints and start and end missions, among other tasks.

2.1.3 Robot Operating System (ROS)

The Robot Operating System [29] is an open-source set of software used to build robotic applications. It includes libraries, message formats, drivers and tools to facilitate developing complex robotics software. It was developed out of many robotic frameworks at various institutions, and is an open-source collaborative project. In this work, ROS is used to collect data from external

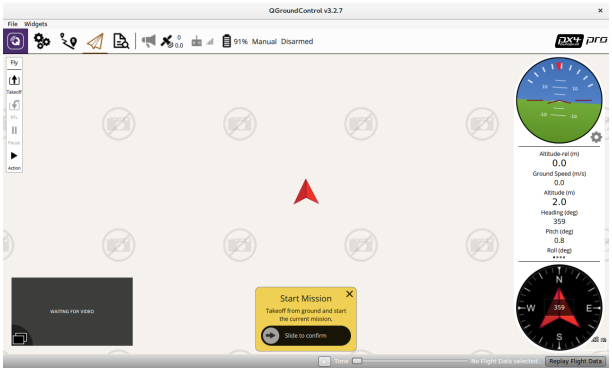

Figure 2.2: PX4 interfaces with QGroundControl, an open-source ground control station

sensing systems, like the ultrasonic beacon system and a computer vision system, and to interface between a companion computer and the flight controller. This makes use of the standard ROS "node" structure and publisher-subscriber message format, as well as more specialized functions such as the mavros library [9]. The mavros library allows for messaging between a companion computer and a PX4-based flight controller using the mavlink message protocol. This is used in this project to pass external state measurements to the flight controller.

2.2 Indoor Localization

The main challenge of operating a quadrotor or other autonomous vehicle indoors is the lack of position from a Global Navigation Satellite System (GNSS) such as GPS. Indoors, the walls and ceiling of the building block GNSS signals causing the inside of the building to be a GPS-denied environment. Therefore, other methods for localization become necessary. As covered in the Introduction, some methods for accomplishing this include ultra-wideband beacon (UWB) systems, beacon-based radio frequency signals like WiFi or BLE, visual odometery, simultaneous localization and mapping (SLAM) and ultrasonic-based beacon systems. These methods differ in accuracy, permeability, cost and power consumption, and these factors must be considered when choosing an indoor localization method. For this application, an ultrasonic beacon-based method is used. It

offers a high level of accuracy, with many systems achieving sub-meter accuracy in positioning. Additionally, ultrasonic beacon systems tend to be relatively low cost. Unlike visual odometry or SLAM, ultrasonic beacon systems do not depend on many environmental variables, like lighting or surface texture. These systems can be set up once and subsequently require little maintenance. While ultrasonic beacon systems do typically produce ultrasonic signals as well as radio signals which could potentially interfere with other signals, in the environments intended for this application it is not expected that there will be signals in frequencies which could lead to interference. And while ultrasonic beacon systems do require a line of sight (LOS) be available between the beacons, this is not a limiting factor in this application. Large indoor spaces like warehouses tend to be regularly structured, facilitating placing the beacons in locations where LOS to multiple beacons can be guaranteed. In addition, as will be discussed in Chapter 4, we will propose a method for dynamically changing the position of the beacons during operations using multiple vehicles.

2.2.1 Marvelmind Robotics Ultrasonic Beacon System

For this project, the Marvelmind Indoor Navigation System is used to provide location data to the quadrotor. The Marvelmind system consists of two types of ultrasonic beacons: stationary beacons and mobile beacons. The stationary beacons are mounted around the perimeter of the operations area, such that at any point in the space the mobile beacon will have a LOS to at least three of the stationary beacons for 3D localization. The mobile beacon is mounted to the object being tracked. All of the beacons can send out ultrasonic signals, and receive ultrasonic signals. Using time-of-flight calculations from these signals, the beacons can calculate relative distances to the other beacons. This information is then communicated to a central modem over a radio frequency. This allows the system to automatically create a map of the space. First, the stationary beacons send and receive ultrasonic signals to each other. This allows the system to calculate the relative distances between all of the beacons, and this information is synthesized into a map of the space by the modem. Once the map has been created, the stationary beacons no longer send ultrasonic

signals. At this point, the mobile beacon begins emitting ultrasonic signals which are received by all of the stationary beacons with LOS. The relative distances between the mobile beacon and all stationary beacons within LOS are then calculated, and using trilateration, the position of the mobile beacon in the space is found. The system provides up to two centimeter-level accuracy for the mobile beacon. The stationary beacons can be placed as far as 30 - 50 meters apart. The beacons can communicate with a modem placed as far as 400 meters away. Update rates can reach 45 Hz depending on the number of beacons used and distance between the mobile beacons and stationary beacons. A minimum of three mobile beacons is required to achieve 3D localization; however, up to 100 beacons can be used in the system. The beacons weigh about 30 grams without a battery, or 50 grams with a battery. They occupy an area similar to a playing card. The system can be monitored and controlled with an included software package. The software features an interface displaying the locations of all the beacons and options for changing the parameters of the entire system. There are multiple ways to obtain data from the system. A companion computer may be connected to any of the beacons or the modem, as these all broadcast the location of the mobile beacons as well as the relative distances of the stationary beacons over UART and SPI. Additionally, a mobile beacon can be configured to broadcast its position as latitude, longitude and altitude information in a local coordinate frame using the NMEA 0183 standard for GPS communication. These position messages can be input directly into a flight controller in place of a GPS signal, and require little to no modification of existing flight software.

2.2.2 Marvelmind Setup

The most important aspect affecting the performance of the Marvelmind system is the placement of the beacons around the room. In the space, the mobile beacon must be in LOS to at least three stationary beacons to ensure the highest levels of accuracy. Additionally, the beacons must be placed either below the lowest altitude the vehicle is expected to operate in (for a quadcopter, on the ground), or above the highest level. Given the potential obstructions of LOS by shelving units

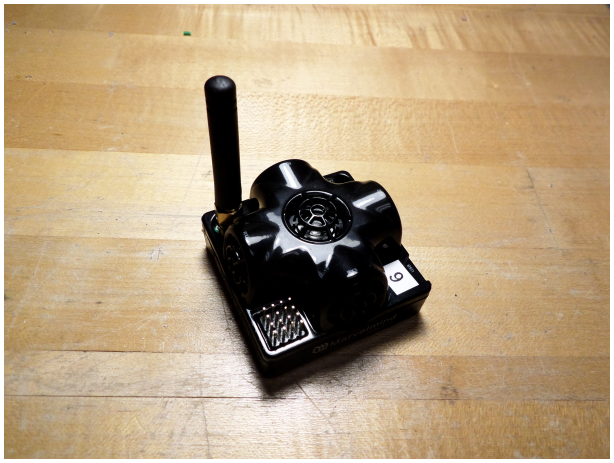

Figure 2.3: The Marvelmind ultrasonic beacon system is composed of up to 100 beacons. Beacons are placed in locations surrounding the operational area as well as on the vehicle or object being tracked.

or other objects in a warehouse-type environment, the best location to place the beacons in this case is on the walls approximately a meter above the highest altitude at which the quadcopter is expected to operate. During the flights described in this work, four stationary beacons are placed around the room, at the corners of a rectangular space. The mobile beacon is mounted on a mast to the top of the vehicle, preventing the beacon from being shadowed by any other parts of the vehicle.

The other important factor in ensuring a effective system setup is the thresholding of the ultrasonic signal. High levels of noise in the environment can lead to interference in the signal being received by the stationary beacons from the mobile beacon. By testing the beacons in the environment in which the operations will take place and monitoring the ultrasonic signals using the oscilloscope provided in the system's software, the appropriate minimum threshold at which each stationary beacon recognizes signals from the mobile beacon can be determined. This is a process that depends on many external variables and is best accomplished using trial and error in the operating environment. Additionally, the system accuracy can be improved by tuning the amplifier

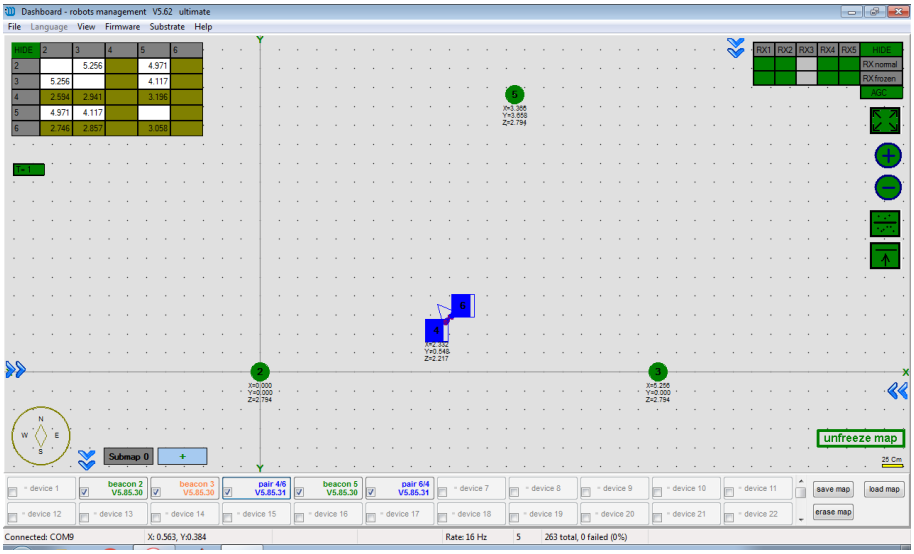

Figure 2.4: The Marvelmind ultrasonic beacon system includes software that can be used to monitor and configure the system. Blue icons represent tracked beacons, green icons represent reference beacons.

gain of each stationary beacon, with the goal of achieving the highest possible signal-to-noise ratio. This can also be done by hand using the oscilloscope tool provided in the software and an iterative process.

2.3 Qualcomm Snapdragon Flight

The Qualcomm Snapdragon Flight board (now known as the Qualcomm Flight) is an aerial robotic development board that focuses on integrating computer vision, flight navigation and communication technology onto one board. On a single board it includes a Snapdragon 801 processor capable of running Linux, and a Qualcomm Hexagon Digital Signal Processor to run flight software. The board also has a forward facing 4K video camera, and a downward-facing optic flow camera. To enable navigation, the board has an on-board IMU as well as support for a barometer and GNSS. For communication, the Qualcomm Flight uses dual-band Wifi as well as Bluetooth.

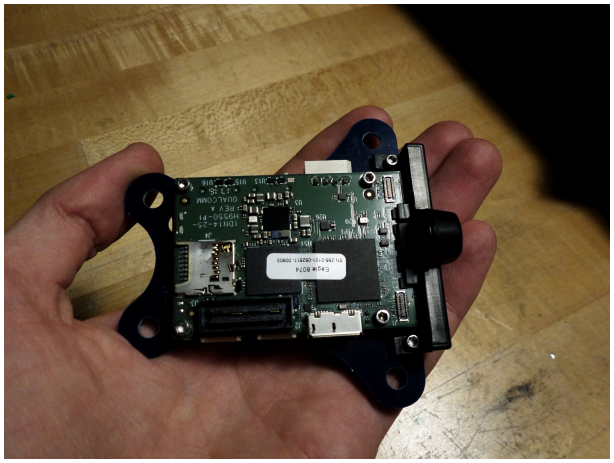

Figure 2.5: The Qualcomm Snapdragon Flight is designed for indoor flight, and features two cameras and inertial sensors to enable visual odometry.

2.3.1 Qualcomm Flight Computer Vision Capabilities

The Qualcomm Flight natively supports the Qualcomm Machine Vision software development kit (SDK). This SDK includes algorithms for SLAM, Depth from Stereo, volumetric mapping and camera calibration tools. Of interest for this project was the built-in SLAM capability. Specifically, this algorithm is referred to as Visual-Inertial Simultaneous Localization and Mapping (VISLAM), as it provides pose (location and orientation) information from a fusion of IMU and camera data in an Extended Kalman Filter (EKF). The Machine Vision SDK and the VISLAM algorithm is proprietary to Qualcomm and not open-source. However, it can be accessed and used on the Qualcomm Flight through the provided ROS interface. This ROS interface allows the VISLAM algorithm to be easily started, stopped and managed through ROS. Additionally, it allows for the pose information produced by the algorithm to be easily passed to other programs such as the flight controller.

Chapter 3

YAW MEASUREMENT

On a typical quadrotor designed for outdoor use, the sensor suite includes a GPS receiver to measure longitude and latitude, a barometer to measure altitude, an IMU to measure linear accelerations and angular rates and a magnetometer to measure heading from the Earth's magnetic field. In indoor environments, three of those sensors become ineffective. GPS signals are relatively weak, and unable to penetrate buildings. In the system presented in this thesis, this is solved by using an ultrasonic indoor positioning system. Generally, barometers are also of little use indoors, as the resolution of most barometers is too low to give an accurate reading of the relatively low range of altitudes a quadrotor can reach indoors. The ultrasonic indoor positioning system also provides altitude information at a higher resolution than most barometers can measure, so it can also be used to measure altitude in addition to the horizontal position of the vehicle. The magnetometer is especially challenging to use in an indoor environment. Large amounts of metal, like the building materials inside buildings, have their own magnetic fields which interfere with the Earth's magnetic field, potentially causing the heading measured by a magnetometer to become unreliable. The magnetometer drifts as it moves through the corrupted magnetic field in the building. One possible solution is to map the changes in the magnetic field throughout the space in which the vehicle will be operated, and apply correcting offsets to the magnetometer readings throughout an operation. However this is a time intensive process that would need to be accomplished at each operation location. While this may be viable for a system that will be used exclusively in one place, it is not an enduring solution for a system that will be used in many locations. Thus, other methods for measuring heading must be considered. This chapter will cover three methods for measuring yaw and compare the performance of these methods.

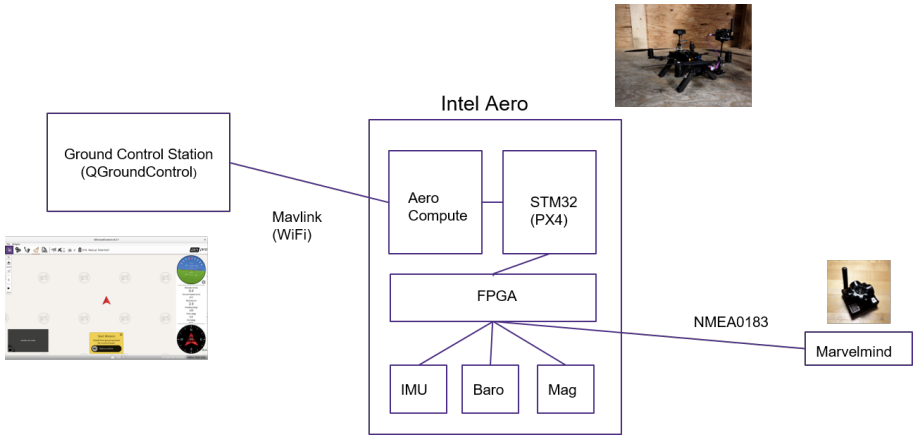

Figure 3.1: The basic setup of the project includes position information sent to the vehicle from the Marvelmind system, and yaw measured using the magnetometer.

3.1 Magnetometer

As mentioned above, magnetometers are standard sensors included on autonomous vehicles and especially UAS. Magnetometers measure the earth's magnetic field, typically in one, two or three axis. The strength of the magnetic field measured by these sensors can then be used to determine the direction of magnetic north in the same way as a compass. The vehicle's heading in a direction relative to north can then be used to navigate between waypoints.

3.1.1 Implementation

The Intel Aero RTF drone uses a Yuneec Typhoon H combined GPS and magnetometer unit. (In this work, since our operations take place indoors, the GPS is disabled by disconnecting the wiring carrying the GPS signal). Measurements from the magnetometer are input into the PX4 flight software over an I2C connection to the Aero Compute board. By default, the PX4 autopilot software calculates yaw from a fusion of IMU data and magnetometer data. In one mode, the magnetic field measured in three axis by the magnetometer is converted to a yaw angle first, which is then used

as an observation in the EKF. In the second mode, all three readings from the magnetometer are used individually as observations to the EKF. The first mode is used in situations when the vehicle is undergoing little movement, such as when it is sitting on the ground. The second mode is used during regular flight; nominally the switch between modes occurs once the vehicle has exceeded 1.5 meters of altitude.

3.1.2 Advantages

The main advantage of the magnetometer is that it is a standard method of measuring yaw, and fully and natively supported in the PX4 flight software. Using it requires no outside software or modifications made to the PX4 flight software. Yaw readings produced from the fusion of the IMU and magnetometer exhibit relatively low levels of noise. Additionally, using the magnetometer does not require the vehicle to have line-of-sight to any external sensor or to have extra computing resources.

3.1.3 Disadvantages

The main disadvantage of using a magnetometer indoors is the drift of the resulting yaw reading observations. This is believed to be due to fluctuations in the magnetic field in the flight area induced by nearby metal building materials or electrical equipment. During flight this manifests itself as a drift in the heading of the vehicle as it moves throughout the flight space. During flight, it can be seen that while the estimator outputs the same heading reading, the vehicle is yawing slowly. This has implications most significantly for navigation, as effectively the drifting of the yaw rotates the body-fixed reference frame relative to the local reference frame in which the vehicle is operating. Thus as it attempts to navigate toward a waypoint, it moves in a direction offset from the actual desired heading. For example, take a vehicle that is initially facing north, then drifts ten degrees toward the east. If it then attempts to head toward a waypoint that is located north of the vehicle in the local reference frame, the vehicle will move toward a point ten degrees

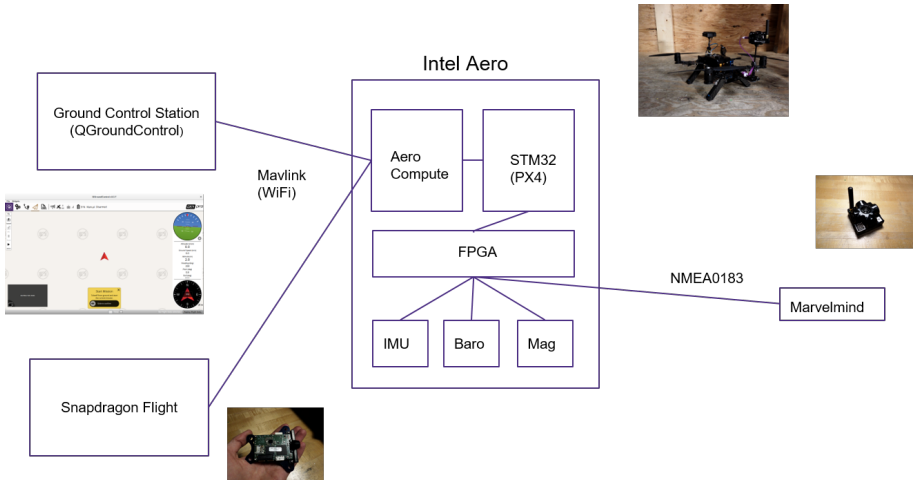

Figure 3.2: The Snapdragon Flight was used to bench-test the use of visual odometry. It was connected to the PX4 software over WiFi using ROS.

toward the east instead. As a result, it will never reach the desired waypoint, but will continue to oscillate its position in an attempt to reach that waypoint. This behavior is undesirable, and if severe enough, can cause a mission to fail completely or result in a crash. As a result of this significant disadvantage, it was determined that another method of measuring yaw is needed.

3.2 Visual-Inertial Simultaneous Location and Mapping (VI-SLAM)

Visual-Inertial Simultaneous Location and Mapping (VI-SLAM) is a method of using both IMU data as well as image data from cameras to localize a vehicle in a space and determine its orientation. The IMU data includes acceleration information from accelerometers and angular rate information from gyroscopes. The images produced by the camera are processed to identify features that appear in sequential frames. The movement of these ”feature points” between subsequent frames can then be sued to determine how much the vehicle has moved. Separately, these data are difficult to use. Location data derived from IMUs have a tendency to demonstrate drift, as they have no external reference, and errors aggregate accordingly. Data from the camera is also subject

to drift, as feature points may not be mapped exactly as the camera moves. However, by fusing the data from the IMU and the camera, VI-SLAM avoids the challenges presented by each of these sensors and can allow for the calculation of the pose of a vehicle.

3.2.1 Implementation

VI-SLAM is implemented using the Qualcomm Snapdragon Flight avionics board. As covered in Chapter 2, the Flight includes an IMU as well as a forward-facing camera and a downward-facing camera. The acceleration and rotational rate data, as well as the images from each camera, are fused to calculate a pose for the vehicle. The software used to perform this fusion is a combination of a proprietary machine vision library developed by Qualcomm as well as an open-source ROS package used to interface with the Qualcomm machine vision library. The output of the ROS node is a ROS message containing position information and orientation information. The position information is in x-y-z coordinates relative to the point at which the VI-SLAM algorithm was initialized. The orientation information is in the form of a quaternion. The message can be published as a `mavros/mocap/pose` message. It is then passed to the PX4 autopilot software by MAVROS in the form of a mavlink message. PX4 interprets this as a pose estimate from an external vision system. While position and orientation information is available in this message, in this work only the yaw information was needed. A parameter in the flight software allows only the heading information from the vision system to be used. The flight software then extracts the yaw information from the quaternion provided by the VI-SLAM algorithm and uses it as a measurement of the heading of the vehicle through a Kalman filter.

3.2.2 Advantages

The VI-SLAM algorithm is designed to be used indoors and has been demonstrated to be stable and effective indoors. It does not depend on offboard sensors, beacons or computational resources. The sensors are relatively lightweight, compact and consume low power.

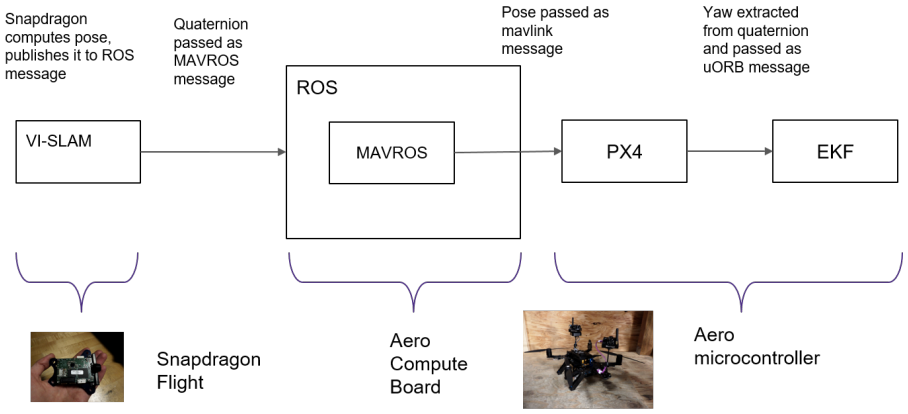

Figure 3.3: The SLAM algorithm producing yaw measurements from the Snapdragon Flight is proprietary, but the measurements can be obtained and passed to the flight controller using ROS.

3.2.3 Disadvantages

Since VI-SLAM is a visual navigation scheme, it is highly dependent on the quality of the images captured by the cameras. This means that lighting is very important. Inconsistent lighting and shadows can result in poor performance of the algorithm. This is especially true when the system is initialized. If the initial images taken by the cameras are too dark, then the position and orientation calculation will not initialize. This means that the system must be initialized with the cameras mounted high enough so that they can take well-lighted images even when the vehicle is on the ground. Texture in the images is also important. As mentioned in the ”Implementation” subsection, the algorithm relies on the ability to identify unique feature points in each image. If the image does not contain enough detail or variation to find these feature points, it will become less accurate or fail completely. In addition, when compared to the other methods in this section, the algorithm is more complex and requires more computational power.

Figure 3.4: When equipped with two beacons, a vehicle's orientation can be obtained from the ultrasonic beacon system.

3.3 Yaw Calculation using Two Ultrasonic Beacons

Given the local coordinates of two points on a vehicle, the orientation of the vehicle can be determined relative to some initial orientation. In this work, the Marvelmind ultrasonic beacon system is already being used to find the location of one point on the vehicle in order to locate the entire vehicle in the room. By placing an additional ultrasonic beacon on the vehicle, the heading can be calculated. This can be done by calculating a vector between the two beacons, and tracking how that vector changes. From the changes in this vector, a quaternion representation of the orientation of the vehicle can be obtained.

3.3.1 Implementation

As described in Chapter 2, the Marvelmind Ultrasonic Beacon system consists of stationary beacons and mobile beacons. The stationary beacons are placed on the perimeter of a flying space and receive signals from the mobile beacon in order to trilaterate the location of the mobile beacon. The mobile beacons are located on the vehicle and during operations send out ultrasonic signals to

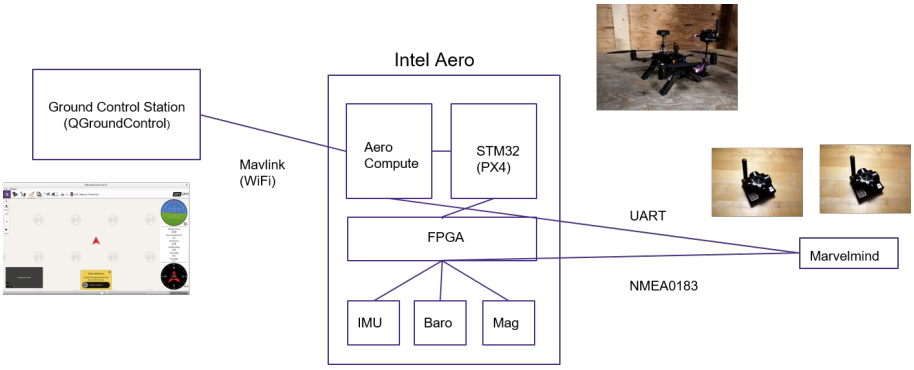

Figure 3.5: When using two Marvelmind beacons, one is connected to the Aero's GPS port to provide position information while the other is connected over UART to provide the individual positions of both beacons.

be received by the stationary beacons. The system is capable of tracking multiple mobile beacons. In order to implement yaw calculation using two ultrasonic beacons, a second mobile beacon is added to the vehicle. The two beacons are mounted about 10 cm apart, one extending off the front of the vehicle and one at the back of the vehicle. Their positions are used in the calculations below to find measure the change in orientation of the vehicle.

Quaternion Calculation

Let the position of the beacon mounted to the rear of the vehicle be designated p_1. Let the position of the beacon mounted to the front of the vehicle be designated p_2. The vector from the rear beacon to the front beacon is then $x_b = p_2 - p_1$. To measure changes in the orientation of this vehicle, a reference vector x_0 is needed. Since the system is using only two points of reference on the vehicle instead of three, there is the possibility of a rotation being unobservable if it occurs around x_0. So if x_0 is chosen to be around the roll, pitch or yaw axis of the vehicle, the orientation of the vehicle may be unreliably measured. Thus x_0 is initialized as a random unit vector so the vehicle is unlikely to rotate about x_0. A quaternion representation of orientation can be computed from a rotation axis and a rotation angle about that axis. In this case, the axis can be found from the cross

product of x_0 with x_b:

$$r_q = \frac{x_0 \times x_b}{|x_0 \times x_b|} \tag{3.1}$$

The angle of rotation can be found from the dot product of x_0 and x_b:

$$\theta_q = \arccos x_0 \cdot |x_b| \tag{3.2}$$

Then the quaternion representing the rotation of vehicle can be found as:

$$q = \begin{bmatrix} r_{q_x} \sin \frac{\theta_q}{2} \\ r_{q_y} \sin \frac{\theta_q}{2} \\ r_{q_z} \sin \frac{\theta_q}{2} \\ \cos \frac{\theta_q}{2} \end{bmatrix} \tag{3.3}$$

Data Handling

The position of each beacon, p_1 and p_2 are received from the ultrasonic beacon system by a ROS node, which publishes the positions to a ROS topic. Another ROS node subscribes to the stream of position information, and uses those measurements to perform the quaternion calculation. This node then publishes the quaternion to MAVROS using the `/mavros/mocap/pose` message. As in section 3.2 it is then passed to the PX4 autopilot software by MAVROS in the form of a MAVLINK message. PX4 interprets this as a pose estimate from an external vision system. While position and orientation information is available in this message, again only the yaw information was needed, so the ”external yaw fusion” parameter is used. The flight software then extracts the yaw information from the quaternion provided by the VI-SLAM algorithm and uses as a measurement of the heading of the vehicle through a Kalman filter (called the Local Position Estimator [LPE]) in PX4.

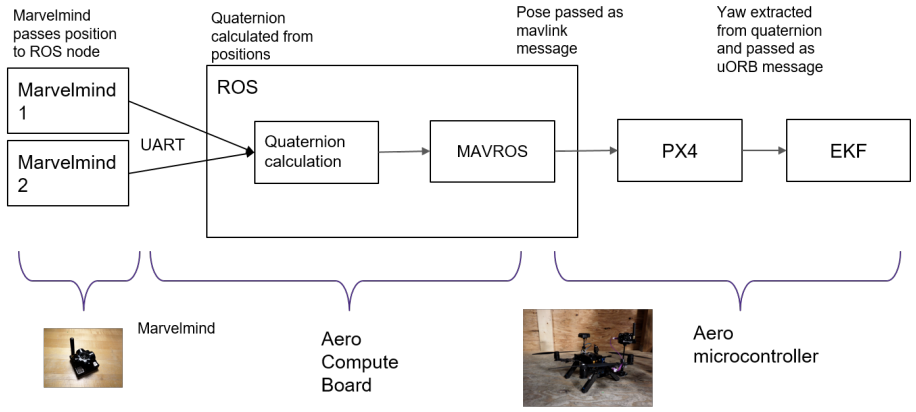

Figure 3.6: ROS was used to obtain individual position information from each Marvelmlind beacon and calculate a rotation relative to some initial orientation.

3.3.2 Advantages

The two-beacon yaw calculation benefits from making use of a system already being used. The only addition to the vehicle is the second Marvelmind beacon. This means that this additional method of calculating yaw incurs little additional complexity, especially compared to the VI-SLAM method, which requires two cameras and additional processing power. As long as the ultrasonic-beacon system is properly set up and operating with high levels of accuracy, the resulting yaw calculation is also highly accurate. The beacons are also relatively low cost. Unlike the magnetometer, if needed, the complete orientation of the vehicle (not just heading) is available through this method.

3.3.3 Disadvantages

The dual beacon system adds more weight to the vehicle than the other two methods described here. The beacons each weigh about 54 grams with batteries, introducing a total of 108 grams to the vehicle. While all of the methods described here are mostly platform independent, this relatively large weight addition requires that a larger vehicle be used that can carry the weight.

This is not an issue for the Intel Aero RTF used in this project, but smaller quadrotors, fixed wing aircraft or lighter-than-air vehicle may not be able to handle the added weight. In addition, using the ultrasonic beacon system for heading information means that both the position information of the vehicle and the heading is dependent on the ultrasonic beacon system. If the system fails or experiences degraded accuracy, then the vehicle would lose information about both its position and orientation at the same time. This could make the consequences of such a failure more drastic or more difficult to recover from.

Chapter 4

MULTIVEHICLE COORDINATION FOR INDOOR NAVIGATION WITH ULTRASONIC BEACONS

The ultrasonic beacon system described in Chapter 2 is relatively low cost. A single small space, such as a research lab, may be easily outfitted with less than ten beacons, especially if it is regularly shaped and there are few structures preventing line of sight between the stationary beacons and the mobile beacons. However, in larger spaces, the number of beacons required increases, and accordingly, so do the system's cost and complexity. In a large operational space such as a warehouse, with many obstacles such as shelving preventing line of sight between the beacons, the number of beacons could rise very quickly to the point where the cost and complexity of the system is burdensome. Thus a method for reducing the number of required beacons is needed. The solution developed in this is work is a multivehicle approach. In this approach, beacons are placed on a set number of quadrotors. One of the quadrotors serves as the "working" quadrotor, moving throughout the space on a set trajectory and performing some objective. The rest of the quadrotors carry the stationary beacons. These quadrotors nominally are landed on the ground or on a perch in places where the stationary beacons mounted on them can provide sufficient coverage to the mobile beacon on the working quadrotor. As the mobile quadrotor moves along its mission-based trajectory, the quadrotors with the stationary beacon also move throughout the space, one at a time, to continually provide the mobile beacon with sufficient coverage, with "sufficient" being determined by some metrics of localizability and accuracy of the localization system. This chapter will introduce an algorithm that can be used to plan a set of waypoints for each vehicle in the system in order to navigate throughout a large indoor space with only a small number of beacons.

4.1 Cost Comparison

The primary driver for creating a system of moving beacons is to reduce the number of beacons needed, and therefore reduce the overall cost of the system. By evaluating the cost of each system modality, it can be seen that the cost can be reduced by fixing the number of needed beacons using a mobile network of beacons. Consider the case of a warehouse composed of multiple aisles. Assuming that a certain number of beacons can be used to localize a vehicle in each aisle, the cost of outfitting the entire warehouse with beacons in each aisle can be compared with the cost of using multiple quadrotors to create a mobile network of beacons. The values used in this case study can be seen in Table 4.1. From Figure 4.1 it can be seen that for any warehouse larger than 18 aisles, it would be more cost-effective to use a system in which the beacons are placed on quadcopters rather than outfitting every aisle with beacons. The cost is calculated as:

$$C_{singlevehicle} = m * c_m + a * c_a + 2 * a * c_b + n_r * b_r * c_b \tag{4.1}$$

where m is the number of ultrasonic modems, c_m is the cost of a modem, a is the number of UAVs, c_a is the cost of a UAV, n_r is the number of rows, b_r is the number of beacons per row and c_b is the cost of a beacon. When using a network of UAVs carrying the reference beacons, the cost is no longer dependent on the number of rows and can be calculated as:

$$C_{multivehicle} = m * c_m + a * c_a + 2 * a * c_b \tag{4.2}$$

4.1.1 Beacon Configurations with Charging Stations

Given the current limitations of battery life of UAVs, when using UAVs in a large indoor space it will be necessary to charge the UAVs during operations. To ensure that charging can be quickly and conveniently accomplished during a mission, charging stations could be placed throughout the area. However, this will have an impact on cost. In this example, charging stations could be placed in the aisles of a warehouse. In the case of using a single vehicle to navigate the space using

Parameter	Single Vehicle	Multivehicle
Modem Cost	$70.00	$70.00
Beacon Cost	$70.00	$70.00
UAV Cost	$1099.00	$1099.00
Number of modems	1	1
Number of UAVs	1	5
Beacons per aisle	4	0

Table 4.1: The cost of a beacon setup for a the case of a warehouse is related to the number of beacons needed.

Parameter	Single Vehicle	Multivehicle
Modem Cost	$70.00	$70.00
Beacon Cost	$70.00	$70.00
UAV Cost	$1099.00	$1099.00
Charging Station Cost	$1000.00	$1000.00
Number of modems	1	1
Number of UAVs	1	5
Beacons per aisle	4	0
Charging stations per aisle	1	2

Table 4.2: The cost of a beacon setup for a the case of a warehouse is related to the number of beacons needed and the number of charging stations needed.

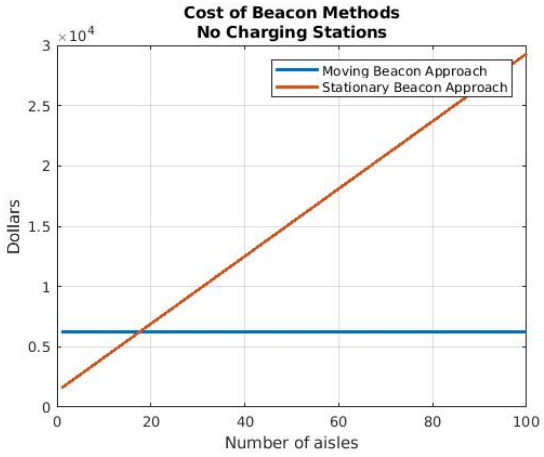

Figure 4.1: As the number of aisles increases, so does the cost for a fixed beacon configuration. When a mobile network of beacons is created, the cost remains constant with the number of aisles.

beacons placed throughout the space, this could mean the addition of one charging station per operational area. For the case of using a network UAVs to move the reference beacons throughout the space, more than one charging station may be necessary in an operational area. The costs of the single vehicle method becomes:

$$C_{singlevehicle} = m * c_m + a * c_a + 2 * a * c_b + n_r * b_r * c_b + n_r * n_c * c_c \tag{4.3}$$

where n_c is the number of chargers per aisle and c_c is the cost of a charging station.

For the multivehicle method, the cost becomes:

$$C_{multivehicle} = m * c_m + a * c_a + 2 * a * c_b + n_r * n_c * c_c \tag{4.4}$$

Given the current high cost of charging stations, this significantly increases the cost of the system. As can be seen in Figure 4.1.1, created using the parameters in Table 4.2 when using 2 charging stations per operation area for a multivehicle setup, this increases the cost such that the multivehicle scenario is actually more expensive than outfitting an entire large indoor space with beacons. Since the cost of charging infrastructure will likely reduce in the coming years, and the

number of needed charging stations for a certain area is still unknown and relatively untested, it is still important to explore how to use a network of UAVs to provide localization in an indoor environment.

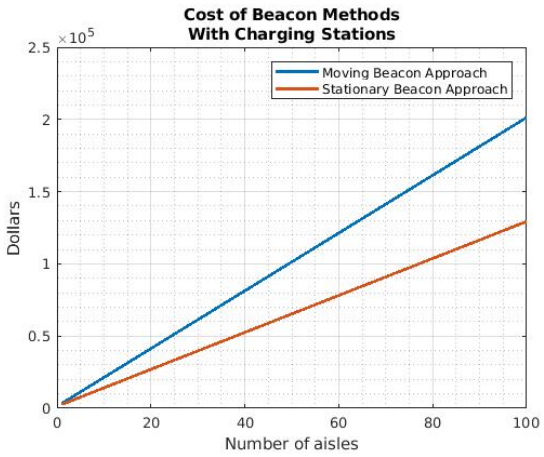

Figure 4.2: When charging stations are used, the cost for each additional aisle increases for both the single vehicle scenario and the networked multivehicle scenario.

4.2 Beacon Configuration Evaluation

In order to determine if a certain configuration of beacons is suitable, we must determine which characteristics of the system are important, and then choose metrics that measure those characteristics. Two important characteristics for an indoor localization system configuration are coverage and accuracy. Coverage is the amount of a space that is covered by the localization system. One metric that can be used to evaluate coverage is localizability. Localizability is a binary metric which tells whether or not a target at a certain location can be localized by the system. By evaluating each point in a space, the percentage of the area that is localizable can be found. Accuracy can be measured using a metric called the Geometric Dilution of Precision, which estimates the accuracy of a measurement of a localization system based on the geometry between the beacons

and target. This section will cover how these metrics are calculated. These calculations are taken from Rajogopal et al. [30].

4.2.1 Localizability

In general, localizability is simply a binary function defined at each space in an operation area that describes whether or not a target at that location can be localized by the system with a certain beacon configuration. Localizability depends heavily on they type of localization system being used. Some systems are severely limited by the range of their signals, and thus distance between the target and beacon can be important. Other system rely on having a line of sight between the target and the beacon. This work is specifically focused on ultrasonic beacon systems. Since the range of the ultrasonic beacons used here is much larger than the distance the target is expected to be from the stationary beacons, we can assume that range does not play into localizability for this system. However, ultrasonic beacon systems do require that the target be within line of sight of at least three beacons. Line of sight means that no object, like walls or shelving, can be between the target and the beacon. We can define a binary function LOS that represents whether or not a target is within line of sight of a beacon. Let X_i be the coordinates of the i-th target and B_j be the coordinates of the j-th beacon. Then:

$$LOS(X_i, B_j) = \begin{cases} 1 & \text{line segment between } X_i \text{and } B_j \text{ does not intersect objects} \\ 0 & \text{line segment between } X_i \text{and } B_j \text{ intersects objects} \end{cases} \tag{4.5}$$

A MATLAB-based line segment intersection solver is used to determine whether or not the line segments intersect.

In addition, the beacons being used in this work have a limited field of view based on the arrangement of sensors. This field of view is approximated to be a hemisphere emanating from the front of the beacon. Thus the mobile beacon must fall within this hemisphere in order to be recognized. Let r be the vector from the point X_i to the beacon B_j. Let o be a unit vector extending from the front of the beacon. Let θ be the angle between r and o, defined as:

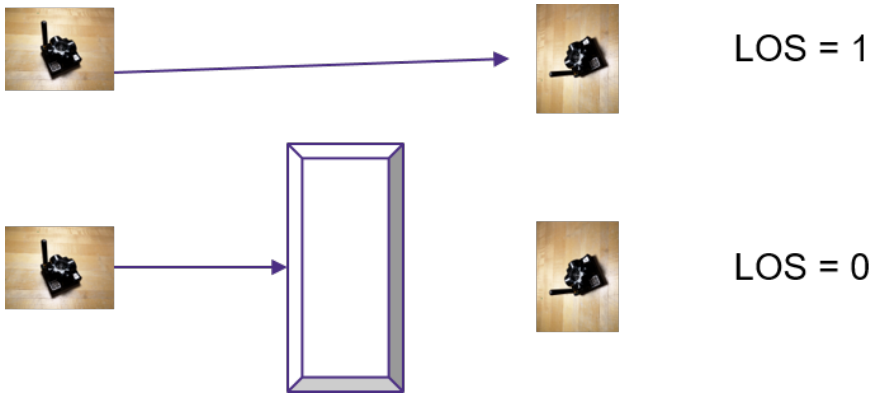

Figure 4.3: Two ultrasonic beacons have line of sight if there are no obstructions between them.

$$\theta = \arctan \frac{||r \times o||}{r \cdot o} \tag{4.6}$$

In order to fall within the field of view of the beacon, this angle must be less than π radians. Thus we can define:

$$FOV(\theta) = \begin{cases} 1 & |\theta| < \pi \\ 0 & |\theta| > \pi \end{cases} \tag{4.7}$$

A target can be considered within coverage of a beacon if both 4.5 and 4.7 are true.

$$C(X_i, B_j) = \begin{cases} 1 & LOS(X_i, B_j) = 1 \text{ and } FOV(\theta) = 1 \\ 0 & LOS(X_i, B_j) = 0 \text{ or } FOV(\theta) = 0 \end{cases} \tag{4.8}$$

Finally, to be localizable, a target must be within coverage of at least three beacons. Let $G(X_i)$ be the set of beacons providing coverage to the target X_i. Let $\#$ represent the cardinality of a set. Then a function for localizabilty can be written:

$$L(X_i) = \begin{cases} 1 & \#G(X_i) \geq 3 \\ 0 & \#G(X_i) < 3 \end{cases} \tag{4.9}$$

This function describes whether or not one target location is localizable. In order to evaluate the beacon configuration over an entire operational area, the percentage of the operational area that is localizable can be determined:

$$Q_L(X, B) = \frac{N_L}{N_X} \times 100 \tag{4.10}$$

where N_L is the number of points in X that are localizable and N_X is the total number of points in X.

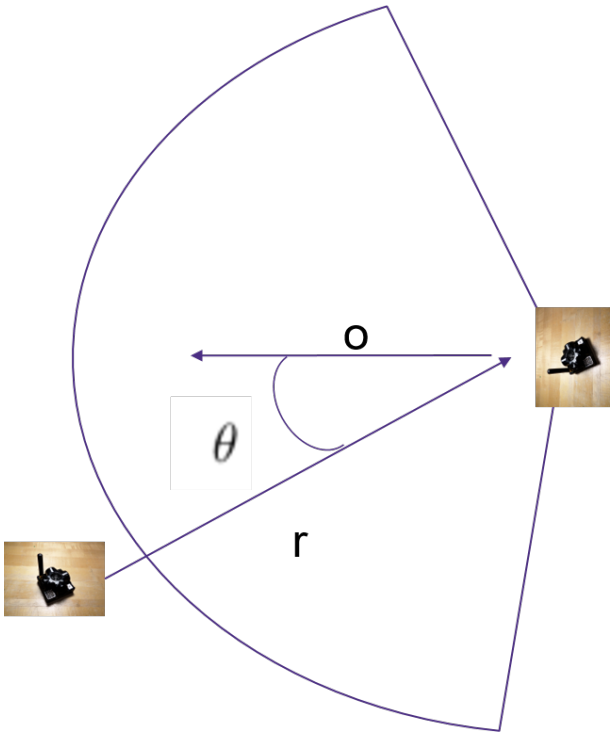

Figure 4.4: A target is within field of view of a reference beacon if the angle between an incoming ray of the ultrasonic signal and the centerline of the reference beacon does not exceed a threshold.

4.2.2 Geometric Dilution of Precision

Once it is known whether or not a space is localizable, the next step is to determine how accurate the localization is at each of those localizable spots. As discussed in Section 1.2.2, many methods of evaluating the accuracy of a trilateration calculation have been developed. For example, there are methods based on probability distribution, the Cramér-Rao bound and the Fisher Information Matrix. For some systems, however, it is sufficient to estimate the accuracy using the geometry of the setup. One such method is the Geometric Dilution of Precision (GDOP). GDOP can be used to describe the accuracy of a multitude of time of arrival navigation and localization methods. It is a measure of the ratio between difference in the location estimation of the target to some change in geometry of the system. Ideally, a small change in the geometry of the system does not cause a large change in the position estimate of the target. In this case, this poor performance would be reflected in a high value of GDOP. Thus an objective of designing a configuration of a beacon-based navigation system is to have a low value of GDOP. There are many formulations and variations of GDOP. The formulation used here is useful for two-dimensional cases like the one being considered. It can be defined as:

$$GDOP(X_i, G(X_i)) = \sqrt{\frac{N_b}{\sum_{n=1}^{N_b-1}\sum_{m=n+1}^{N_b} A_{nm}}} \tag{4.11}$$

where N_b is the number of beacons in G and A_{nm} is the angle between beacons G_n and G_m.

This function gives a measure of the accuracy of the localization system at one point in space. In order to evaluate the beacon configuration over an entire operational area, an empirical cumulative distribution function (CDF) can be used. A lower GDOP is preferable for a certain beacon configuration. The empirical cumulative distribution function measures what proportion of a set of values falls below a certain value. it can be defined as:

$$F(y) = N_X^{-1}\sum_{i=1}^{N_X} I(GDOP(X_i)) \tag{4.12}$$

where I is a binary function describing whether or not the value of the GDOP at the point falls below the target value.

$$I(GDOP(X_i)) = \begin{cases} 1 & GDOP(X_i) \leq y \\ 0 & GDOP(X_i) > y \end{cases} \tag{4.13}$$

Using the set of GDOP values for the desired operational area and the associated empirical cumulative distribution function, the accuracy of the entire space can be evaluated by choosing a maximum desired level of GDOP for the whole space, and determining what proportion of the space it is desired to fall within that performance bound. For example, it may be that in order to have satisfactory performance, a GDOP of less than 3 is desired. Then to evaluate a beacon configuration, the empirical CDF can be used to find the proportion of the space that has a GDOP of less than 3. Alternatively, the area under the empirical CDF curve can be used to evaluate the accuracy of a beacon configuration. In this case, an upper bound on the GDOP, $GDOP_{max}$ must be chosen. Any target and beacon configuration that exhibits a GDOP over this value will be considered not localizable. Once this value has been chosen, the integral of the empirical CDF curve can be taken as a measure of the accuracy of the beacon configuration.

$$Q_{GDOP} = \int_0^{GDOP_{max}} F(y)dy \tag{4.14}$$

4.3 Multivehicle Coordination Algorithms

4.3.1 Prioritizing Localizability

This algorithm can be found written in an algorithmic format in the Appendix (A.1).

Let P be the set of all possible beacon locations. Let $B \subseteq P$ be a candidate beacon configuration. Let $S = P \setminus B$ be the set of all open beacon locations in P. Let W be the set of waypoints the vehicle will follow on its path through the space. Let V be the set of reference vehicles, the vehicles that will carry the reference beacons around the perimeter of the space. Start with some

initial beacon configuration B_1 for waypoint $w_1 \in W$. To determine the next beacon configuration, choose a vehicle $v \in V$. For every open beacon location $s \in S$, calculate the resulting Q_L for the beacon configuration B_j in which v moves to s. Record which beacon location s_{max} produces the highest value of Q_L for vehicle v. Repeat for each v in V. Of all $v \in V$, determine which vehicle v_{max} produces the highest value of Q_L. Move v_{max} to its s_{max}.

It is possible that multiple vehicles and candidate locations will produce the same value of Q_L. In this case, the GDOP can be used to disambiguate the best choice for the next beacon configuration. For each of the beacon configurations with the same Q_L, Q_{GDOP} can be evaluated. Then from the set of beacons with the same value of Q_L, the beacon configuration with the lowest Q_{GDOP} can be chosen.

4.3.2 Prioritizing Accuracy

This algorithm can be found written in an algorithmic format in the Appendix (A.2).

Let P, B, S, W and V be defined as in section 4.3.1. Start with some initial beacon configuration B_1 for waypoint $w_1 \in W$. To determine the next beacon configuration, choose a vehicle $v \in V$. For every open beacon location $s \in S$, calculate the resulting Q_L for the beacon configuration B_j in which v moves to s. Ensure that this value is above some desired threshold for localizability. Then, for every open beacon location $s \in S$, calculate the resulting Q_{GDOP} for the beacon configuration B_j in which v moves to s. Record which beacon location s_{max} produces the lowest value of Q_{GDOP} for vehicle v. Repeat for all v in V. Of all $v \in V$, determine which vehicle v_{max} produces the lowest value of Q_{GDOP}. Move v_{max} to its s_{max}.

4.4 Simulation Results

Simulation of each algorithm was carried out using Matlab R2017b. The simulation takes as input a set of operation areas defined by waypoints, the locations of walls or other obstacles, and initial beacon configuration, orientations for each beacon and a maximum acceptable GDOP. It produces

as output the locations for each vehicle at each waypoint. Simulations presented here were run for two scenarios: one in which the area being navigated had only exterior walls, and one in which the area being navigated also has interior walls. Both scenarios were run for the case when localizability was prioritized and when accuracy was prioritized.

4.4.1 Prioritizing Localization

Without Interior Walls

When prioritizing localizability, the beacons move to provide the most coverage in the area surrounding the waypoint. In this example, the beacons move to keep the operational area that the waypoint is in within line of sight. It can be seen in Figure 4.4.1 that in the final configuration the beacons are aligned along the wall. This seems to conflict with experience with working with the beacon systems, in which having beacons arranged in convex shapes is important. The behavior can likely be improved to match the physical system by updating the model of the beacon system in the simulation to more closely represent the physical system. Some adjustments that can be made are accounting for the field of view of each sensor of the beacon (there are 5) and accounting for interference and echoing.

With Interior Walls

In Figure 4.4.1, the effect of interior walls can be seen. The beacons move to provide coverage on one side of the wall, then the other side of the wall, which can be seen most prominently from the movement of vehicle D and C.

4.4.2 Prioritizing Accuracy

Without Interior Walls

The simulation results when prioritizing accuracy were very similar to those for when prioritizing localization. This was likely do to the fact that accuracy and localizability are closely related. For example, if a beacon cannot localize a point in the area, that beacon location is also resulting in a decrease in the accuracy metric. Some differences did result from prioritizing accuracy, demonstrating that some trade-off did still exist between localizability and accuracy. This can be seen in Figure 4.4.2

With Interior Walls

As in the example with prioritizing localization when there are interior walls, Figure 4.4.2 demonstrates how a wall can reduce the line of sight of the beacons and affect both localizability and accuracy of certain configurations. The differences between the vehicle positions in 4.4.2 and 4.4.2 demonstrates how the vehicles move to provide accurate coverage from either side of an interior wall.

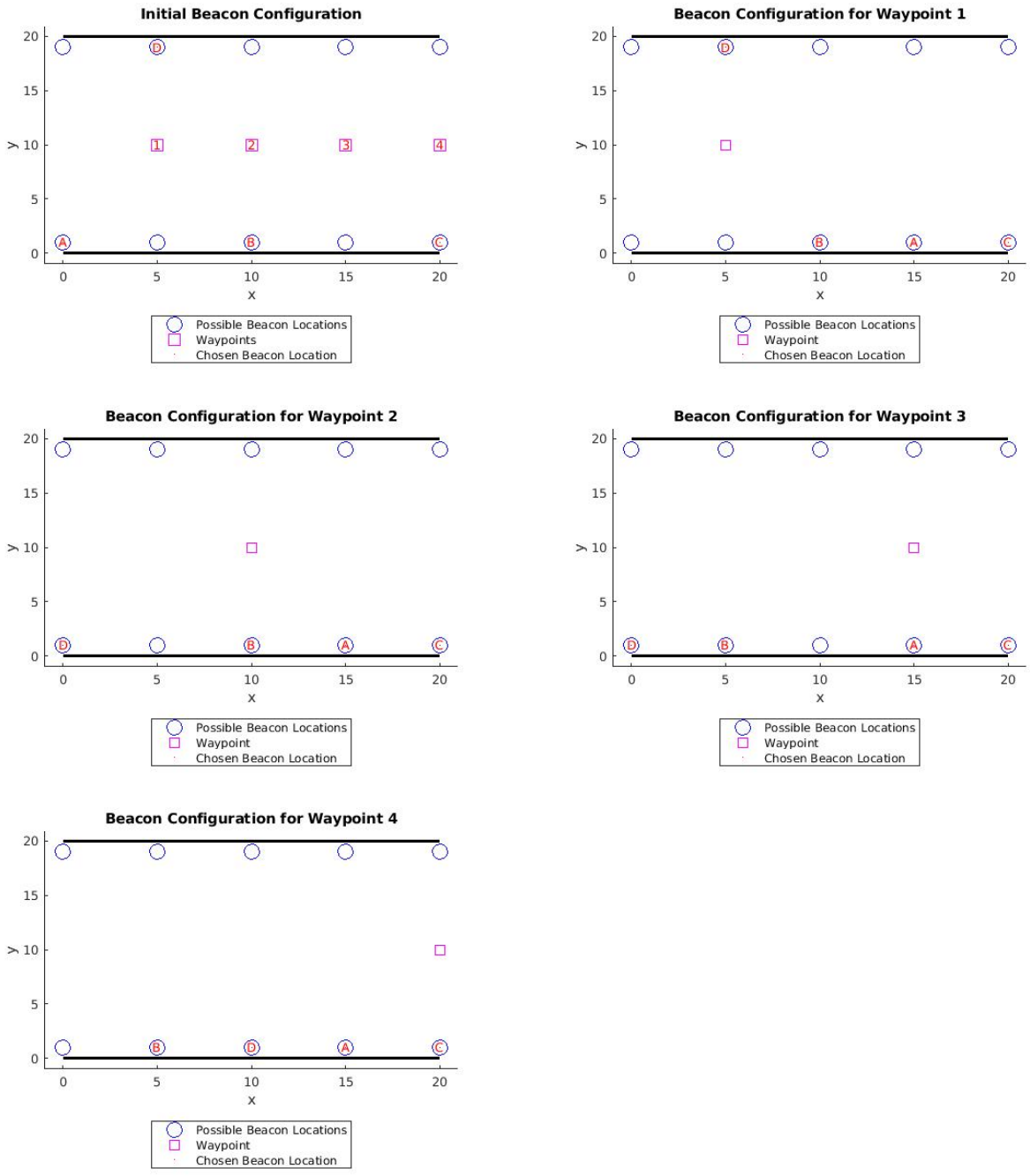

Figure 4.5: This series of plots shows an example of multivehicle coordination for moving beacon locations when prioritizing localizability, and there are no internal walls in the operational area.

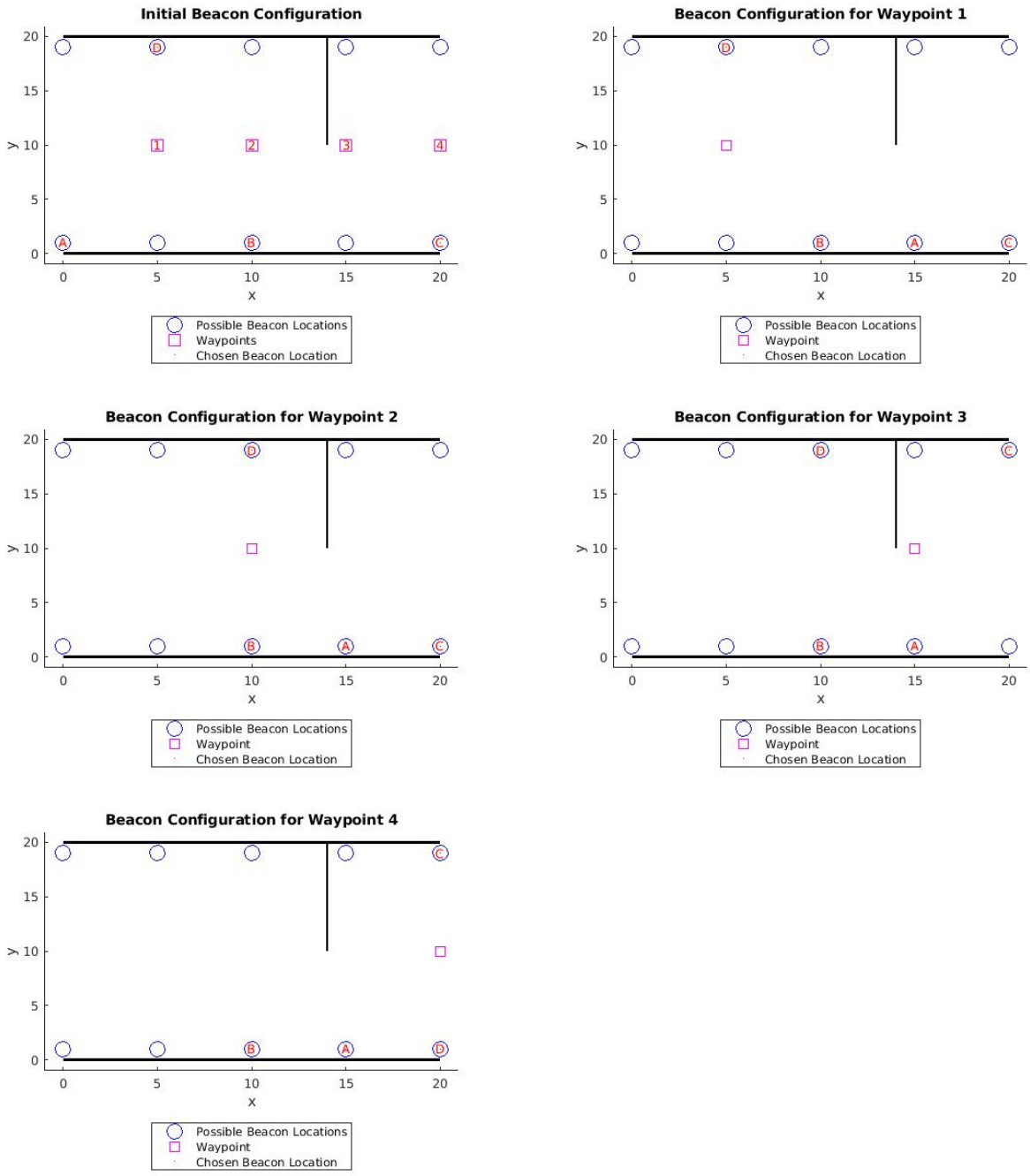

Figure 4.6: This series of plots shows an example of multivehicle coordination for moving beacon locations when prioritizing localizability, and there are internal walls in the operational area.

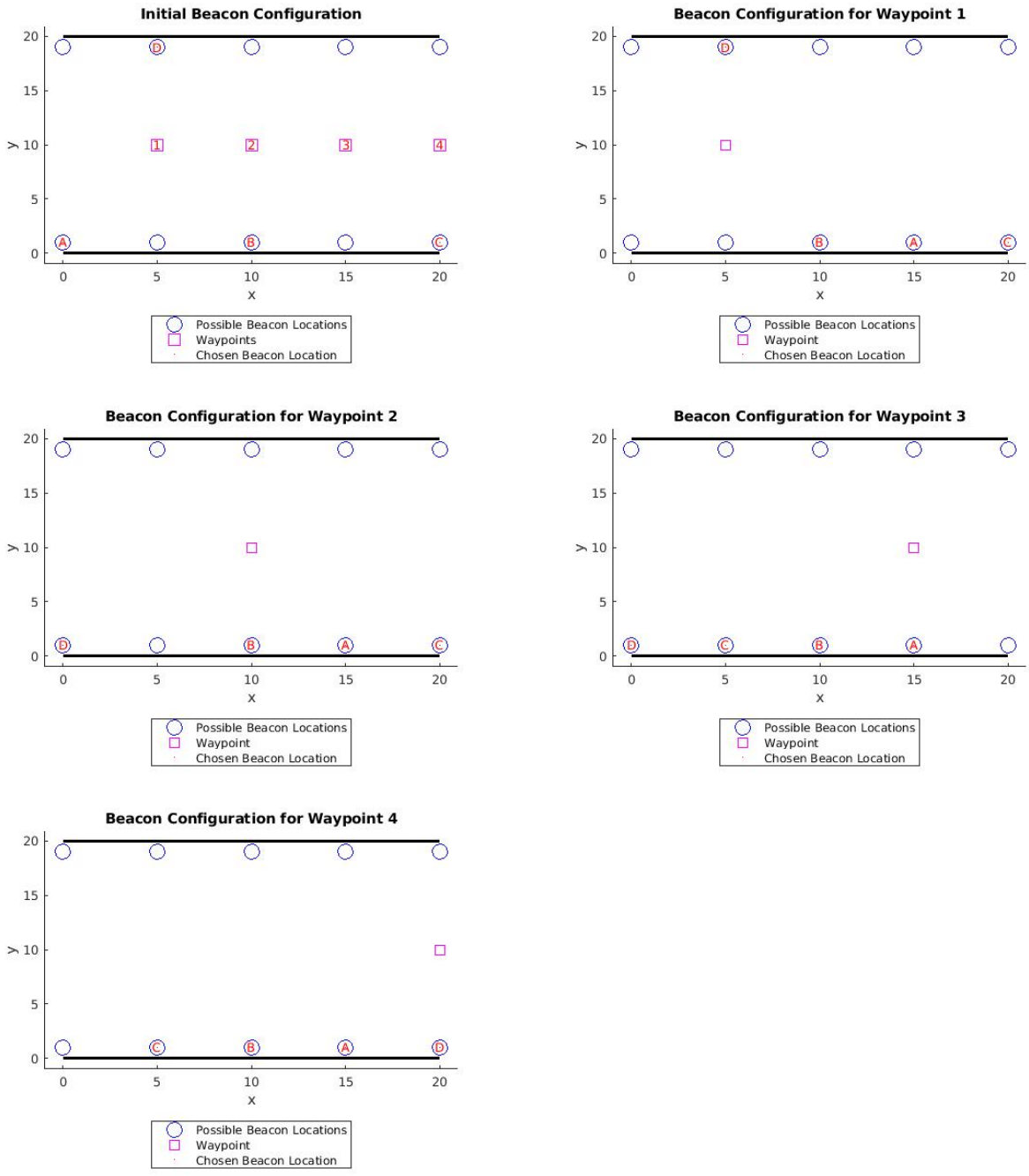

Figure 4.7: This series of plots shows an example of multivehicle coordination for moving beacon locations when prioritizing accuracy, and there are no internal walls in the operational area.

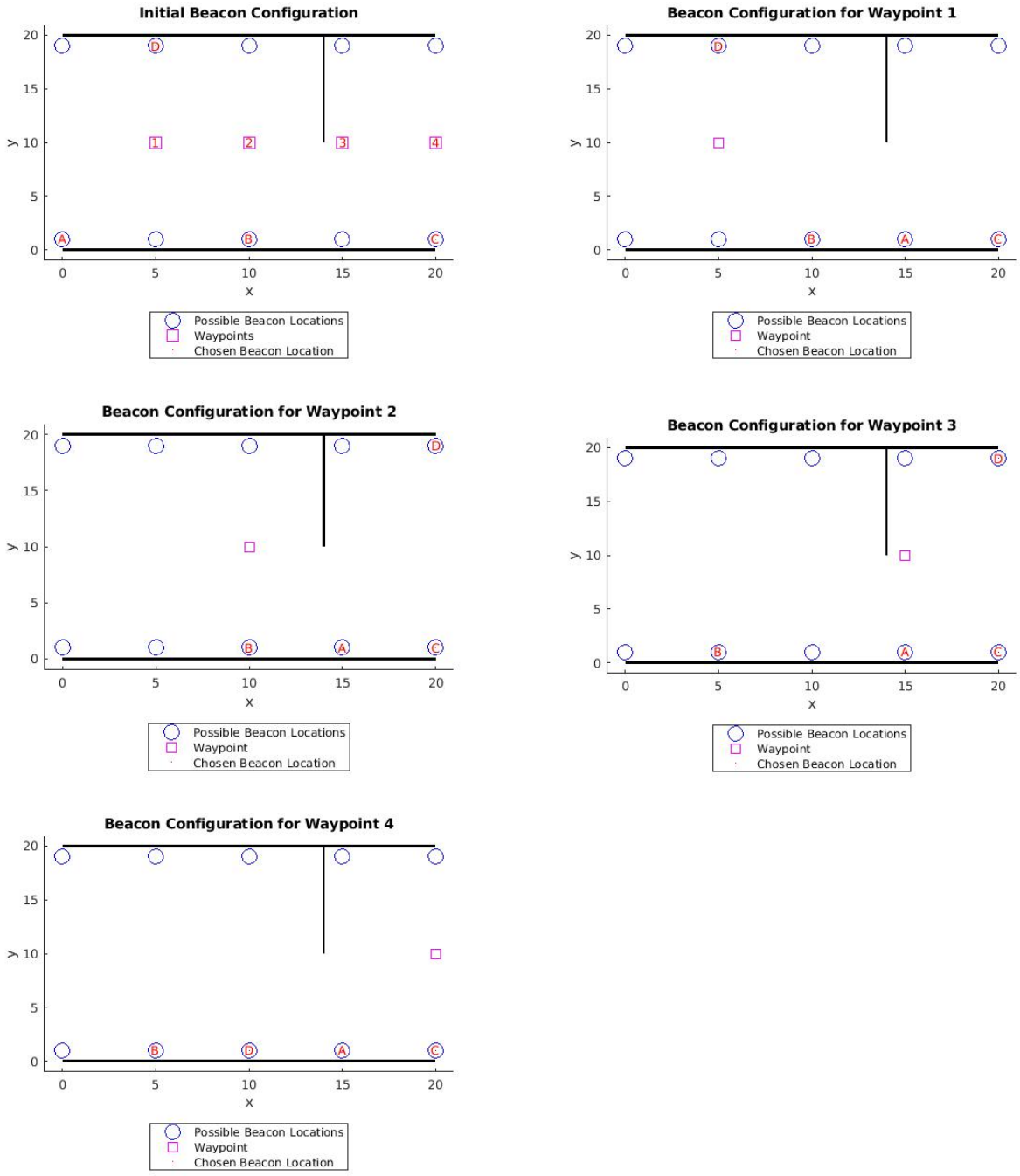

Figure 4.8: This series of plots shows an example of multivehicle coordination for moving beacon locations when prioritizing localizability, and there are internal walls in the operational area.

Chapter 5
CONCLUSIONS

5.1 Summary

This thesis addresses three challenges facing the use of autonomous vehicles in indoor spaces: low cost hardware and software integration, accurate yaw measurement using ultrasonic beacons and multivehicle coordination for navigating indoor environments. First, it proposes a framework for integrating commercial-off-the-shelf hardware and open-source software to obtain a low-cost quadrotor platform for indoor flight. This platform consists of an Intel Aero RTF drone, the Marvelmind ultrasonic beacon system, PX4 flight software, the Robot Operating System (ROS) and the QGroundControl ground control system. The Marvelmind system provides position information to the Intel Aero. In conjunction with ROS, the Marvelmind system is also used to provide yaw information for the vehicle. Mission planning, parameter adjustment and calibration are conducted using QGroundControl. The `mavros` package is used to communicate between ROS and PX4 as well as between PX4 and QGroundcontrol. The resulting system is capable of autonomously navigating an indoor space to accomplish a mission.

Second, the thesis develops a methodology for using an ultrasonic beacon system to measure the yaw of a quadrotor. This methodology includes equipping the vehicle with two ultrasonic beacons and tracking the changes in the positions of these two beacons to measure the change in yaw of the vehicle. This system avoids the drift in yaw measurements that occur with a magnetometer-based yaw measurement system or the dependency of vision-based systems on environmental lighting and surface texture. The advantages and disadvantages of this system are compared with those of using a magnetometer or using vision-based systems.

Third, the thesis proposes using a network of vehicles in order to reduce the number of beacons

needed in an ultrasonic beacon system to navigate a large indoor area. In this method, the stationary reference beacons of the ultrasonic system are placed on quadrotors, which then move around the space in order to continually provide localization coverage to the vehicle performing the mission. Simulations were conducted to demonstrate how the paths of each vehicle in the system can be predetermined to provide either maximal coverage or maximal accuracy within different room configurations.

5.2 Future Work

5.2.1 Yaw Measurement

The first step to improving the yaw measurement using the ultrasonic beacon system is to conduct further testing of the system. As of the writing of this thesis, the system has only been tested on the ground; no flight tests have been conducted. Flight tests will allow the performance to be further evaluated and determine what changes can be made to allow the system to perform better. In addition, the method for inputing the yaw information into the PX4 flight software can be improved by further modifying the PX4 software. Currently the yaw information is input into the software in the format used for external vision information, and it is thus treated as camera data. This leads to operational challenges, as the software needs to be modified to be used without a compass, and the characteristics of the yaw information from the ultrasonic beacon system is different from traditional external vision information. One possible alternative is to modify the PX4 code to allow the yaw information from the ultrasonic beacon system to be incorporated in the format of compass data, which would reduce some of the other modifications that need to be made currently in the software. Finally, the effects of using the ultrasonic beacon-based yaw measurement system together with other methods of measuring yaw, like using computer vision, can be investigated. This would necessitate a method of integrating a camera into the system and fusing the data from the beacons and from the camera. This would allow the system to obtain improved yaw measurements, and provide secondary methods of obtaining yaw information if one

of the systems becomes unavailable. For example. this could occur when there is insufficient beacon coverage, or poor lighting for the camera system.

5.2.2 Multivehicle Coordination for Indoor Navigation with Ultrasonic Beacons

The primary improvement that can be made to the multivehicle coordination simulation is to expand the simulation to three dimensions. This is especially important to increasing the fidelity of the simulation because in practice the moving reference beacons will operate above the vehicle performing the mission objectives in order to provide altitude information to that vehicle. The accuracy of the model of the ultrasonic beacon signals can also be improved. This includes taking into account the field of view of each individual sensor on the beacon, the effects of echoes and multipath effects. The algorithm determining the waypoints for each vehicle could also be improved in many ways. Instead of evaluating the coverage and accuracy of an entire area around the desired waypoints of the mission vehicle, the algorithm could be improved to evaluate the localizability and accuracy along some desired flight path of the mission vehicle. In addition, the algorithm could be improved by using optimization techniques such as convex programming or optimal control methods to choose an optimal beacon configuration. Once the simulation has been improved, the next steps would be to test the algorithm experimentally, first on ground vehicles then using aerial vehicles such as quadrotors.

BIBLIOGRAPHY

[1] Qgroundcontrol v3.2.7, December 2017. https://github.com/mavlink/qgroundcontrol.

[2] A. Alaeddini and K. A. Morgansen. Autonomous state estimation using optic flow sensing. In *2013 American Control Conference*, pages 585–590, June 2013.

[3] Omead Amidi, Takeo Kanade, and Keisuke Fujita. A visual odometer for autonomous helicopter flight. *Robotics and Autonomous Systems*, 28(2):185 – 193, 1999. Intelligent Autonomous Systems (IAS-5).

[4] Seiji Aoyagi, Masaharu Takano, and Hajime Noto. Development of indoor mobile robot navigation system using ultrasonic sensors. *IFAC Proceedings Volumes*, 31(2):305 – 309, 1998. IFAC Workshop on Intelligent Components for Vehicles (ICV'98), Seville, Spain, 23-24 March.

[5] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, volume 2, pages 775–784 vol.2, 2000.

[6] Alexander Bahr, John J. Leonard, and Maurice F. Fallon. Cooperative localization for autonomous underwater vehicles. *The International Journal of Robotics Research*, 28(6):714–728, 2009.

[7] Paul Castro, Patrick Chiu, Ted Kremenek, and Richard R. Muntz. A probabilistic room location service for wireless networked environments. In *Proceedings of the 3rd International Conference on Ubiquitous Computing*, UbiComp '01, pages 18–34, Berlin, Heidelberg, 2001. Springer-Verlag.

[8] J. Curcio, J. Leonard, J. Vaganay, A. Patrikalakis, A. Bahr, D. Battle, H. Schmidt, and M. Grund. Experiments in moving baseline navigation using autonomous surface craft. In *Proceedings of OCEANS 2005 MTS/IEEE*, pages 730–735 Vol. 1, Sept 2005.

[9] Vladamir Ermakov. mavros kinetic. https://github.com/mavlink/qgroundcontrol.

[10] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball. The whoi micro-modem: an acoustic communications and navigation system for multiple platforms. In *Proceedings of OCEANS 2005 MTS/IEEE*, pages 1086–1092 Vol. 2, Sept 2005.

[11] W. E. Green and P. Y. Oh. Optic-flow-based collision avoidance. *IEEE Robotics Automation Magazine*, 15(1):96–103, March 2008.

[12] David Gualda, Jess Urea, Juan C. Garca, Enrique Garca, and Jos Alcal. Simultaneous calibration and navigation (scan) of multiple ultrasonic local positioning systems. *Information Fusion*, 45:53 – 65, 2019.

[13] Mike Hazas and Andy Ward. A novel broadband ultrasonic location system. In *Proceedings of the 4th International Conference on Ubiquitous Computing*, UbiComp '02, pages 264–280, London, UK, UK, 2002. Springer-Verlag.

[14] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys. An open source and open hardware embedded metric optical flow cmos camera for indoor and outdoor applications. In *2013 IEEE International Conference on Robotics and Automation*, pages 1736–1741, May 2013.

[15] J. S. Humbert, R. M. Murray, and M. H. Dickinson. A control-oriented analysis of bio-inspired visuomotor convergence. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 245–250, Dec 2005.

[16] D. B. Jourdan and N. Roy. Optimal sensor placement for agent localization. In *2006 IEEE/ION Position, Location, And Navigation Symposium*, pages 128–139, April 2006.

[17] Benjamin Kempke, Pat Pannuto, and Prabal Dutta. Polypoint: Guiding indoor quadrotors with ultra-wideband localization. In *Proceedings of the 2Nd International Workshop on Hot Topics in Wireless*, HotWireless '15, pages 16–20, New York, NY, USA, 2015. ACM.

[18] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, Nov 2007.

[19] Qi Lu, Beibei Ren, Siva Parameswaran, and Qing-Chang Zhong. Uncertainty and disturbance estimator-based robust trajectory tracking control for a quadrotor in a global positioning system-denied environment. *Journal of Dynamic Systems, Measurement, and Control*, 140(3):031001–031001–15, Nov 2017.

[20] Sonia MartíNez and Francesco Bullo. Optimal sensor placement and motion coordination for target tracking. *Automatica*, 42(4):661–668, April 2006.

[21] L. Meier, D. Honegger, and M. Pollefeys. Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6235–6240, May 2015.

[22] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525, May 2011.

[23] M. B. Milde, O. J. N. Bertrand, R. Benosmanz, M. Egelhaaf, and E. Chicca. Bioinspired event-driven collision avoidance algorithm based on optic flow. In *2015 International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, pages 1–7, June 2015.

[24] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI International Joint Conference on Artificial Intelligence*, pages 1151–1156, 2003. Cited By :614.

[25] Vali Nazarzehi and Andrey V. Savkin. Distributed self-deployment of mobile wireless 3d robotic sensor networks for complete sensing coverage and forming specific shapes. *Robotica*, 36(1):118, 2018.

[26] James H. Neilan, Josh Eddy, Loc Tran, Benjamin Kelley, Andrew K. McQuarry, Matthew Vaughan, Ralph Williams, and Bonnie D. Allen. chapter Field Testing Visual Odometry: Results from Benchtop to Flight for Autonomous Science Mission Needs. AIAA AVIATION Forum. American Institute of Aeronautics and Astronautics, Jun 2017. 0.

[27] E. Olson, J. Leonard, and S. Teller. Robust range-only beacon localization. In *2004 IEEE/OES Autonomous Underwater Vehicles (IEEE Cat. No.04CH37578)*, pages 66–75, June 2004.

[28] J. Qi and G.-P Liu. A robust high-accuracy ultrasound indoor positioning system based on a wireless sensor network. *Sensors (Switzerland)*, 17(11), 2017. cited By 0.

[29] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. Ros: an open-source robot operating system. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

[30] N. Rajagopal, S. Chayapathy, B. Sinopoli, and A. Rowe. Beacon placement for range-based indoor localization. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, Oct 2016.

[31] Franck Ruffier and Nicolas Franceschini. Optic flow regulation: the key to aircraft automatic guidance. *Robotics and Autonomous Systems*, 50(4):177 – 194, 2005. Biomimetic Robotics.

[32] Randall C. Smith and Peter. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986.

[33] E. Solomon, C. Vorwald, V. Hrishikeshavan, and I. Chopra. Visual odometry onboard a micro air vehicle using snapdragon flight. In *7th AHS Technical Meeting on VTOL Unmanned Aircraft Systems and Autonomy*, 2017.

[34] Weiss Stephan, Scaramuzza Davide, and Siegwart Roland. Monocularslambased navigation for autonomous micro helicopters in gpsdenied environments. *Journal of Field Robotics*, 28(6):854–874.

[35] M. Szmuk, C. A. Pascucci, D. Dueri, and B. Aikmee. Convexification and real-time on-board optimization for agile quad-rotor maneuvering and obstacle avoidance. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4862–4868, Sept 2017.

[36] P. V. Teixeira, M. B. Nogueira, and J. B. Sousa. Long baseline beacon position estimation. In *OCEANS 2011 IEEE - Spain*, pages 1–7, June 2011.

[37] J. Tiemann and C. Wietfeld. Scalable and precise multi-uav indoor navigation using tdoa-based uwb localization. In *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–7, Sept 2017.

[38] Jakuba Michael V., Roman Christopher N., Singh Hanumant, Murphy Christopher, Kunz Clayton, Willis Claire, Sato Taichi, and Sohn Robert A. Longbaseline acoustic navigation for underice autonomous underwater vehicle operations. *Journal of Field Robotics*, 25(1112):861–879.

[39] J. Vaganay, J. J. Leonard, J. A. Curcio, and J. S. Willcox. Experimental validation of the moving long base-line navigation concept. In *2004 IEEE/OES Autonomous Underwater Vehicles (IEEE Cat. No.04CH37578)*, pages 59–65, June 2004.

[40] Floris van Breugel, Kristi Morgansen, and Michael H Dickinson. Monocular distance estimation from optic flow during active landing maneuvers. *Bioinspiration & Biomimetics*, 9(2):025002, 2014.

Appendix A

A.1 Algorithm 1 - Prioritizing Localizability

for all vehicles v in V **do**
  for all new beacon location p in P **do**
    for all point i in operating area **do**
      for all beacon j in current beacon configuration **do**
        Check line of sight between point i and beacon j
        if i is in LOS **then**
          $LOS(i, j) = 1$
        else
          $LOS(i, j) = 0$
        end if
        initialize count of localizable points to 0
        for all row k in LOS **do**
          $inrange(k) =$ number of non-zero entries in k
          if inrange(k) > 2 **then**
            add 1 to count of localizable points
          end if
        end for
        if $inrange(i) \leq 2$ **then**
          GDOP = NaN
        else

calculate GDOP
end if
end for
end for
Calculate CDF of GDOP
Integrate CDF from 0 to maximum desired GDOP
Localizable percentage =count of localizable points / number of points in area $*100$
if Localizable percentage $>$ maximum localizable percentage **then**
optimal beacon = p
optimal vehicle = v
else if Localizable percentage = maximum localizable percentage **then**
if integral of GDOP CDF $>$ maximum integral of GDOP CDF **then**
optimal beacon = p
optimal vehicle = v
end if
end if
end for
end for
return optimal beacon
return optimal vehicle

A.2 Algorithm 2 - Prioritizing Accuracy

for all vehicles v in V **do**
  for all new beacon location p in P **do**
    for all point i in operating area **do**
      for all beacon j in current beacon configuration **do**
        Check line of sight between point i and beacon j
        if i is in LOS **then**
          $LOS(i, j) = 1$
        else
          $LOS(i, j) = 0$
        end if
        initialize count of localizable points to 0
        for all row k in LOS **do**
          $inrange(k) =$ number of non-zero entries in k
          if inrange(k) > 2 **then**
            add 1 to count of localizable points
          end if
        end for
        if $inrange(i) \leq 2$ **then**
          GDOP = NaN
        else
          calculate GDOP
        end if
      end for
    end for
    Calculate CDF of GDOP

Integrate CDF from 0 to maximum desired GDOP

Localizable percentage =count of localizable points / number of points in area $*100$

if Localizable percentage $>$ threshhold **then**

if integral of GDOP CDF $>$ maximum integral of GDOP CDF **then**

optimal beacon = p

optimal vehicle = v

end if

end if

end for

end for

return optimal beacon

return optimal vehicle