**Graduate Projects**

University of Colorado at Boulder

Aerospace Engineering Sciences

ASEN 5018/6028 –Spring 2015

|  |
| --- |
| **Drones Versus Zombies (DVZ)**  **Localization and Mapping/Sensing Subsystem (LAMSS)**  **Summary/Continuity Document** |

|  |  |  |
| --- | --- | --- |
| **Approvers List** | | |
|  | Title | Name |
| Prepared By | LAMSS Lead | Austin Lillard |
|  | LAMSS Team Member | Tyler King |
|  |  |  |
| Approved By | LAMSS Lead | Austin Lillard |
| Approved By | Systems Engineer | Edward Scott |
| Approved By | Project Manager | Garrett Brown |

# Introduction & Summary

The purpose of the Localization and Mapping/Sensing Subsystem (LAMSS) is to combine measurements from several sensors on the quadrotor to obtain the location of the quadrotor within a pre-mapped environment.

We have selected the following sensors for our system:

* The Pixhawk for its accelerometers, magnetometers, and gyros
* The PX4FLOW for its altitude sonar
* The Hokuyo UST-10LX for its laser range measurements.

The ROS package Adaptive Monte Carlo Localization (AMCL) will be used to provide position and yaw estimates of the quadrotor using the Hokuyo sensor and the odometry frame. The odometry frame will be generated by an extended Kalman filter that combines all measurements to produce an estimate of the quadrotor position and velocity. Finally, the ROS package Hector Simultaneous Localization and Mapping (SLAM) will be used in conjunction with the Hokuyo sensor to create a map of the environment in which the quadrotor will be flying.

# Semester Report

## Objectives and Tasks List

Completed:

1. Integrated all sensors necessary for localization and mapping into the quadrotor and characterized their accuracy and noise
2. Integrated the laser scan matcher package to ROS as an estimate body velocity
3. Corrected the Kalman filter for correct rotation matrices
4. Implemented the quadrotor model to include a linear drag estimate

Incomplete:

1. Fully tested and tuned Kalman filter
2. Integration of Kalman filter with controls
3. Tuning of AMCL parameters

## Issues

1. Difficulties in verifying the assumptions made in the model. Being able to store the data efficiently and determine what the results tell us about our model proved to be difficult. With the amount of “tuning” knobs present in the Kalman Filter, it is difficult to find the exact combination that will yield satisfactory results.
2. Lack of technical knowledge in the area made the initial design of the Kalman filter difficult. As the semester went on, and we learned more in our classes, we were able to gain a much better understanding but were running out of time to implement new ideas.
3. Creating a test environment that accurately replicates the Lockheed Martin Hallway is a difficult task. Actual tests in the hallway prove that it is much more difficult to estimate positions and velocities in the actual hallway than in our test environment.
4. Latency issues between the various systems proved to be a critical issue in the implementation of the Localization subsystem. The quality of the wireless signal had a large effect on the overall performance of the system and is a key consideration for operating in the Lockheed Hallway!
5. The data coming from the pixhawk over Pymavlink is really slow (~15-20 Hz). This makes the linearization assumptions made in the Kalman filter implementation not hold up. Sensors providing acceleration and Euler angle data need to be sampled at a faster rate in order for the Kalman filter assumptions to work properly.

## Lessons Learned

1. Read Probabilistic Robotics (at least the first half). Even after taking a course on Estimation, the pseudo-code provided in Probabilistic Robotics was an incredibly useful resource that we didn’t take advantage of early enough. It is actually an easy read and I even bought a copy of it for myself because I enjoyed it so much!
2. Test in the actual environment much more frequently. Testing in the Lockheed Hallway is key in being able to understand how the system will actually perform. Due to the hallway being hard to replicate, it is worth it to head down to the Lockheed Hallway at least every other week to check that the system is behaving as expected.
3. Designing your own particle filter (or perhaps altering AMCL’s code) might not be a bad idea. The motion model of the particle filter, which is fully described in Probabilistic Robotics, does not match up with quadcopter dynamics exactly. A custom motion model could be built to address these issues and allow for better interconnectivity between the dead reckoning solution and the particle filter.
4. Test the sensors in the actual environment early! For example, the PX4Flow ultrasonic sensor being used for altitude determination does not work in the carpeted section of the Lockheed Martin Hallway, but does work in the RECUV test space. At this point in time, there is no way to have any altitude estimate in the carpeted section of the Lockheed Martin Hallway. We also were shipped a bad Hokuyo laser scanner and did not realize it until too late in the semester. With the shipping and repair times, there was no way we could get the sensor back before the semester ended.

## Procedures

The most detailed information can be found in the Spring 2015 version of the DVZ Design Document.

* Note that the password to the Laptop is ddvvzz

Table 2.1: Software list

|  |  |  |
| --- | --- | --- |
| Program Name | Version | Purpose |
| Hector SLAM |  | Used to map the environment. This is the pre-mapping step of the implementation of the project. It utilizes just the laser scanner to create an occupancy grid of the environment. It’s launch package can be accessed by typing “roscd hector\_slam\_launch”. |
| map\_server |  | This ROS package allows for a map to be saved and then published at a later time. The saving is done by using “rosrun map\_server map\_saver” while the publishing of a map that has already been saved uses “rosrun map\_server map\_server” |
| AMCL |  | This is the particle filter used for the localization with a laser scanner. It has many different parameters and is described in detail in Probabilistic Robotics. |
| RVIZ |  | This is the viewer that interfaces directly with ROS. This viewer is able to display the occupancy grid, particle locations, frames, and laser scans simultaneously. It is a very useful, necessary, tool for developing the localization algorithm with ROS. |
| Laser Scanmatcher |  | This ROS package is used to generate the body velocities from the laser scanner. This package provides a covariance estimate as well as a dead reckoning position estimate. It does NOT work well in the Lockheed Martin Hallway however. |
| Hokuyo node |  | Allows for the laser scan to be published to a topic named “/scan”. This is used by the laser scanmatcher, hector slam, and AMCL. |
| roscopter |  | The interface with the autopilot that allows for sensor measurements to be obtained. Due to the slow rate, using the “mavros” package instead is recommended. Note that “mavros” does not work with the Arducopter firmware! |
| Px4flow node |  | Publishes the ultrasonic reading to a topic. Also gives optical flow data from the px4flow sensor, but the velocity estimate from this optical flow sensor is not accurate when tested in the actual testing environment. May be possible to increase the accuracy of this optical flow or use the raw image in a more advanced algorithm. The camera on the px4flow sensor also does NOT work with Vicon on. |
| roscore |  | Starts the roscore and enables the computer to be the master. This allows for communication using the ROS interface. |

# Next Semester/Future Expectations

## Prioritized List of Tasks and Objectives

1. Look at the Kalman Filter implementation and see how the system performs with sensors accessed at higher rates (~100 – 200 Hz).
2. Investigate the creation of a custom motion model to be used in AMCL. Possibly write an entirely new particle filter, but it is likely a better idea to look at AMCL’s code and modify it to fit our needs because it already has all of the ROS interfaces that we require for implementation.
3. Look at implementing other sensors to estimate the odometry. A visual based odometry method might not be a bad idea for implementing in the hallway and I believe there are ROS packages that already do this. This should be looked into as a possible way to improve the performance of the state estimation.

## Starting Points

1. The current kalman filter implementation code is found on the team’s computer in the catkin workspace. The fastest way to get to it is to type “roscd odometry”, which will take you to the “odometry” folder in the workspace. In this folder are many different python scripts that have been created for the odometry calculations. The most recent kalman filter implementation is “kalman\_mark\_austinv4.py”. Looking at this file will allow you to see how to interface with ROS properly and how all the matrices were implemented in python.
2. AMCL is a package that is installed and run off of the ODROID. To access this package and the launch files, the command “roscd amcl” can be used. In the examples folder, the launch files for DVZ are present. The launch files form a basis for the various parameters that can be changed in the particle filter. To fully understand these parameters, looking at the AMCL page on the ROS website and reading Probabilistic Robotics will be necessary. The source code for AMCL should be located in the “src” folder of the catkin workspace on the odroid. This source code should contain the motion model of the particle filter, which can be altered to match quadcopter dynamics.
3. Check on the ROS website for current packages available for visual odometry and look at what other research laboratories are doing with the technology. Several universities post their code for their visual SLAM and odometry implementations, so looking around will hopefully yield good results. Unfortunately, I have not had time to investigate the software that is out there or the cameras that might be used in the implementation of the visual odometry on a quadcopter.

## Improvement, Updates, Verification

1. The Kalman filter design is based off of work done at BYU. The basic concept of it makes sense, but the assumptions might break down in the hallway scenario. Also, the current assumption is that the low rates of the incoming sensor data was the problem in the performance of the Kalman filter. This has not been verified, and should therefore be one of the first things investigated and improved upon.
2. We have assumed that a particle filter is the best way to implement the localization algorithm, but there are other ways to localize within a space. Maybe it would be possible to localize without the need for many particles, but this would require a lot of extra work to implement.
3. We have assumed that the map that was created is a good enough representation of the physical space. Without truthing the map, it is difficult to know just how well our mapping package performed, especially in the Lockheed Martin Hallway. Because the map is discretized, the discretization size actually does have an effect on the performance of the localization algorithm. This should be further investigated to ensure that the mapping process is actually not causing problems with the localization algorithm.