Autonomous Drone Flight using Object Avoidance and SLAM Implementation

Brayan Castaneda
Electrical and Computer Engineering
California State University, Fresno
Fresno, U.S.A
skycap@mail.fresnostate.edu

Jonathan Herrera
Electrical and Computer Engineering
California State University, Fresno
Fresno, U.S.A
Jon21herrera@mail.fresnostate.edu

Justin Short
Electrical and Computer Engineering
California State University, Fresno
Fresno, U.S.A
Jdshort11@mail.fresnostate.edu

Abstract - **As technology has continued to improve over the last decade, the continuous approach of Moore's law has allowed onboard computing to reach new milestones. With the ability to perform such high levels of computing, technology has pivoted in a direction seeking utilization of hardware capable of being used dynamically. Adopting this mindset, the concept of using drones for critical situations such as search and rescue, has become a reality for the day-to-day operations of many. While a concept as such could be implemented completely manually, this is not ideal as it proposes many potential errors. For the proposed project, efficiency and reliability were at the forefront of priorities. Autonomous drone flight using object avoidance and SLAM implementation presents a robust system with many use cases. While this technology has been around for some time, modern-day computing, machine learning, and other processing utilities have allowed for new implementations to show the precision and effectiveness they can operate at.**

1. Introduction

Simultaneous Localization and Mapping (SLAM) is a method of estimating the location of a device relative to its surroundings in real time. The data to calculate the position of the device, in this case, a drone, is generated by an array of sensors which must then be statistically analyzed and used for predictive analysis in order to calculate relative position on and future movement. The data generated from the sensors can be used in a variety of ways making the use cases for this system robust and applicable to many situations. Through a combination of hardware and software, the data taken from the sensors becomes an input to the onboard GPU of the Nvidia Jetson TX1. The Nvidia is implemented with the purpose of processing the data and producing a result which is used to calculate the certainty of an object detected, allowing the drone to determine whether or not to avoid the object. While this processing is taking place, the drone will have a known location based on the

analyzed previously analyzed objects. The concept has been around since the late 80's to early 90's but it due to the limitations of hardware was limited in its capabilities. With modern day computing implementing components sequentially smaller than submicron, the computing power needed to integrate SLAM into mobile platforms has become a reality. The proposed project aims to display the power of modern-day computing by producing a system capable of using object detection for object avoidance while utilizing SLAM methods for autonomous drone flight.

II. Existing Technologies

A. Consumer Products

Although a Roomba vacuum cleaner may not be thought of as an autonomous robot, the algorithm used to give the device its functionality is not far off from the SLAM implementation looking to be achieved in this project. The Roomba is able to navigate its way around a room based on the integrated sensors and will sweep the majority of the room without losing its position and going in circles.

B. Autonomous Vehicles

The current market for autonomous vehicles is continuously growing. Whether it is a well-established motor vehicle company such as Ford or a newer company such as Tesla, the race to create a reliable fully autonomous vehicle is on the to-do list of many companies. Tesla has been the front-runner in this market and has designed a vehicle which can conduct autonomous driving but has its limitations. Aside from their vehicles being only semi-autonomous, some of their autonomous features have encountered critical mistakes such as misjudging the distance when parking and shutting a door prematurely and as a result hurting the consumers.

C. Machine Learning Devices and Software

Machine learning has sparked a wave in technology with many still wondering what its ultimate use case is. Machine learning techniques have been used in a variety of different ways with some cases leaving people to ask only the question of “why?”. Whether the techniques are integrated into robots attempting to simulate human nature or used in apps to add a filter around the face of a person, the only conclusion to be drawn is that machine learning can serve many purposes. Many companies have used machine learning to suit their needs but there is still a long way to go before anyone considers themselves a master of the subject.

The standpoint of existing autonomous technologies is still in the infancy stages with many errors being corrected after the product has been released to the public. While there is no current standard to go off of when implementing object detection and avoidance with SLAM, there is a significant amount of research being done in both areas. The current stages of the existing technologies leave many doors open for anybody to propose new solutions and derivatives of the products created to date.

III. Design Objective

The design of the system begins with the Intel Aero drone. The Intel Aero comes equipped with a reprogrammable I/O via Altera Max 10 FPGA, Intel Atom x7-Z8750 processor and an Intel RealSense camera (R200) [1]. Having these components onboard the AERO will allow the data generated from the sensors to be retrieved and exported to the Nvidia TX1 for computation. The Aero has an integrated Inertial Measurement Unit (IMU) and supports DroneCode autopilot and ArduPilot open source software. With the provided open source software and pre-configured sensors, the SLAM algorithm can be configured as needed.

To configure the SLAM algorithm a formula to estimate the Aero’s position must be developed. Using references to a number of SLAM devices online the method of an Extended Kalman Filter (EKF) was determined as a fit for the proposed project.

Considering an assumption of known correspondence, the position of the Aero can be considered as,

$$X_t = (x, y, \theta,\ \ m_{1,x}, m_{1y} \ldots m_{n,x}, m_{ny})$$

Where (x,y, θ) is the Aero’s position and the array of m’s are representative of the relative landmarks.

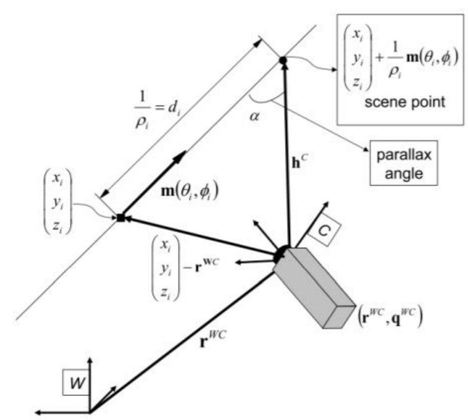


Figure 1. Vector space analyzed to determine the positioning of the Aero [4].

Given the observation of the landmark and the position relative to its surrounding, the EKF computes a probability allowing producing an updated location. The probabilistic calculation can be given by Bayes’ Rule which provides the probability of an event based on the prior conditions given.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

The Aero will begin in its own reference frame which means at time t_0, all relative objects are unknown. As a result, the system must use inference and execute an algorithm of maximization in order to produce the move for the next time step to the highest probability. The error created in the vector space is non-linear and therefore cannot be corrected with complete accuracy. Due to this condition, the simplest adjustment is to estimate the error based on a gaussian method or to utilize the sensors and use the incoming data for inference. The primary data used for inference will be vision provided by the Intel RealSense camera onboard the Aero.

Given that the inference will be conducted by analyzing the data from the RealSense camera, we must construct a layout of the surroundings. In order to construct this layout accurately and provide the Aero with sufficient data for a relatively accurate prediction of the next move, image extraction landmark extraction occurs [2]. Our primary source of landmark extraction will be done through software in OpenCV programmed with Python language. Python's built-in libraries allow for a simple implementation to analyze the data. The data will be analyzed through syntax which will produce a line best fit for the statistical approximation of the current landmark. Our current x and y coordinates are essential to the calculation of the line best fit as it will be dependent upon the mean (μ), standard deviation (σ) and variance (σ^2). Based on the provided locations we can construct a correlation matrix representing the surrounding landmarks relative to the location.

We can represent the correlation between locations as the product-moment correlation coefficient given by,

$$r = \frac{\sum_{i=1}^{n}[(x^{i}-\mu_x)(y^{(i)}-\mu_y)]}{\left(\sum_{i=1}^{n}(x^{(i)}-\mu_x)^2 \sum_{i=1}^{n}(y^{(i)}-\mu_y)^2\right)^{\frac{1}{2}}}$$

The term "r" gives the linear relationship between the two variables x and y.

Through this correlation, a decision can be made as for the progressing step of the Aero. On occasion, there will be an error in positioning and the proposed correction to this is done via software as well.

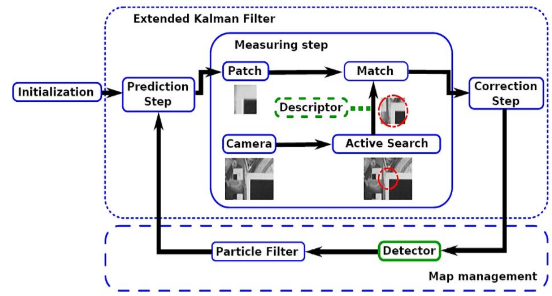


Figure 2. EKF block diagram displaying prediction and step correction.

When necessary correction can be made by running a check sequence between the current landmark and previous landmarks.

- If the landmark is yet to be initialized
  - Initialize the landmark and proceed
- else
  - re-compute the correlation and move in the correct direction

Although this solution will correct as needed, the algorithm must become more efficient than the provided pseudo code as there is a potential that the correction could run in $\Theta(n^2)$ complexity. The TX1 boasting significant computing power for its size can compute these corrections but over time it could pose a hazard if the CPU slows down significantly.

The recognition of landmarks is essential to SLAM implementation but the data used during this process can also be used to detect pre-specified objects. Each and every object has features which are unique, because of this, objects can be classified with accuracy. There a few methods which can be used when attempting to classify objects.

Since the TX1 does not come preconfigured to recognize any particular set of landmarks as a discrete object, the dataset to do so must be generated by using a training algorithm which must go through many magnitudes of iterations of learning. Although the TX1 does boast great computing ability for its size, training for new object datasets are best done on a desktop computer with powerful graphical computing ability in order to exploit the extreme parallel processing offered in modern graphical processing units (GPUs).

Edge matching: Finding the edges of the object and tracking the overlaps, creating the general shape of the object.

Grayscale matching: The edge pixels are used to compute the distance and intensity between pixels creating a relation to the shape. The same method can be applied using color as a classifier.

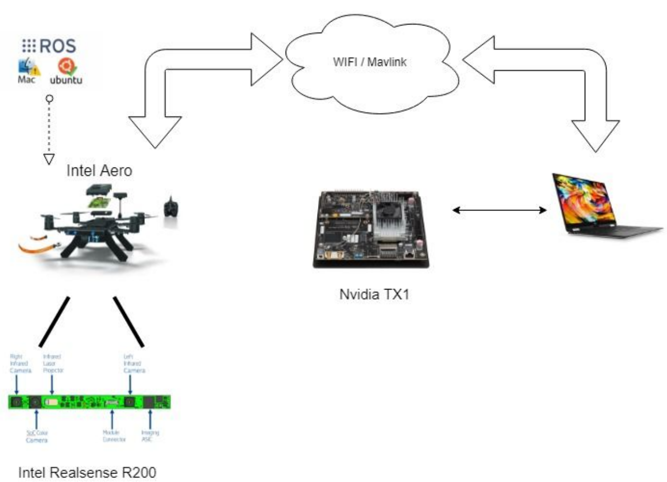
Gradient matching: changes in gradient can be an accurate representation of the depth and shape of an object.

With OpenCV, any of these methods can be used and they can each be tailored to classifying various objects.

By first establishing a method of landmark recognition, representation of landmarks and associated data can be accomplished by using point clouds – the mapping of associated landmarks using discrete points in a 3-dimensional space. Since each point in the point cloud is its own dataset containing a coordinate, color, intensity, and relation to other points, the environment/object in question can be rendered for immediate viewing or saved and exported for further data processing through programs such as MATLAB. Such processing will enable the point cloud dataset to either be calibrated, filtered, or re-rendered for a different environment.

IV. Objectives and Deliverables

The objectives agreed upon by the members of this group are as follows: Autonomous Drone Flight, Data analysis via methods of predictive filters or machine learning software, and SLAM implementation.



The project deliverables are broken down into three primary sections: The Intel Aero, Communications, and data processing.

The tasks to be completed on the intel Aero from the initial point of opening the box consist of many different steps to create an environment that will allow our SLAM algorithm to effectively compute the Aero's movement. The Aero first needs to be configured with the proper settings allowing for a point-to-point communications via WIFI protocol. After creating that access point, the group will be able to utilize the Linux distribution Yocto and the integrated command line tool to update all firmwares and install the software stack Robot Operating System (ROS). The ROS running on the Aero will be the primary source of data feedback from the TX1 resulting in the drones movement from the SLAM data.

The communication portion of the project will be completed using Micro Air Vehicle Link (MAVLink) protocol. The code generated in MAVLink headers will communicate between the data generator, the Intel Aero, and the ground station, the Nvidia TX1. MAVLink protocol utilizes headers written in object oriented languages in order to send data to the intercommunication systems of the two devices. Along with data, MAVLink can be utilized to access the orientation of the devices. This will be significant for the Intel Aero given that the project focus is to determine the Aero's movement as a result of analyzing the data the cameras are creating.

All data processing of this project will be centered around the Nvidia TX1. Since the Intel Aero's integrated compute board is limited to a dual core processor, attempting to integrate SLAM on the Aero itself is likely to cause a number of issue. The TX1 is our solution to minimizing latency of the complete system. By offloading the data via MAVLink protocol onto the Nvidia TX1, the Intel Aero can optimize efficiency by using its total computing power to create the data needed for the TX1 to process. While processing the data for movement decisions, the TX1 will also be used to complete object detection. The group intends to complete data training sets of a couple specific objects in order to provide the project with an example of industry use cases.

The deliverables of this project are all tasks involved in getting the previously discussed sections completed. The work to be completed begins with the drone itself. In order to have anything else work, the drone must first be able to complete autonomous flights. The initial stage is to enable Ardupilot open-source software. From here the drone flight can be evaluated for the remainder of the tasks to be completed. Once autonomous drone flight is achieved the focus can be shifted toward hardware interaction. The drone must properly deliver the data from the cameras and other sensors in order to provide the software with sufficient data for decision making. The ability to retrieve the desired data will allow the project to move forward with software development. The software will aim to train data sets to recognize objects and likewise avoid objects. The final stage of the project will focus on the mapping and localization capabilities of the drone. By having all other portions completed the drone will possess the ability to analyze its surroundings and relate its surroundings to its current location.

The intended delivery date for project completion is April 1st, 2018. Structuring all other deliverables based on this date provides a time conscious plan to completion. Given this date is met, this will allow thorough testing to validate all requirements have been met. The initial start date began October 16th, 2017. On this date, the initial setup of the TX1 and some beginning tests were conducted. Through this semester we are aiming to have the Intel Aero in flight and by the winter looking to have the Aero and TX1 in communication. The setup of the hardware is the vital component to the success of this project.

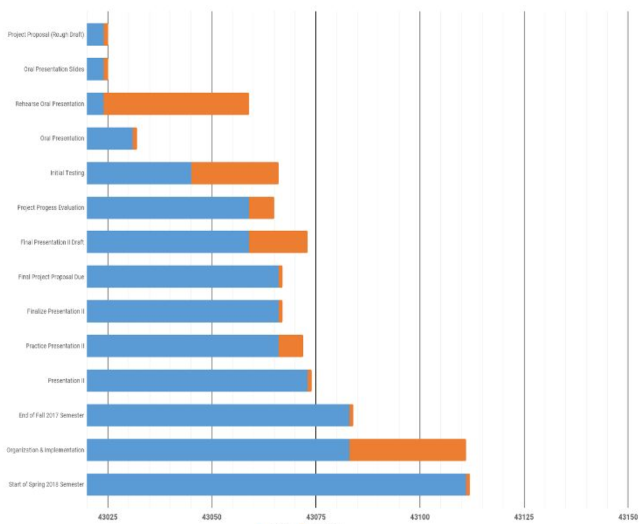


Figure 3: Proposed Gantt chart displaying project milestones

The proposed Gantt chart for project management has been constructed with Agile methods in mind. Each Milestone and deadline aims to have a significant portion of the projected completed at the same time. By closely monitoring checkpoints, the semester has been rationed for each deadline to represent the end of a sprint. Using this method enables an efficient method to monitor success and areas of weakness. Through each sprint, the team will access the progress and adjust the remainder of the project accordingly.

Each team member is to have a breadth of knowledge pertaining to all project related software. Following this will make for efficient programming implementation and troubleshooting. Each week during lecture, all team members must provide a progress update specific to the current position of the project.

V. Budget and Cost Estimates

The budget for this project currently requires no funding. The software necessary can all be found for free through open source resources. Major hardware expenses necessary to complete this project will not need to be purchased by the group. The most significant expenses have been acquired for the term of this project through school organizations USRT and IEEE. We are grateful to have these resources as it will speed up the projects completion by easing the financial cost covered by the group. The remainder of the components needed to complete the project are all peripherals which can be acquired at relatively low costs.

Table 1: Cost of equipment in use

Organization	Equipment	Cost
IEEE	Nvidia TX1	$600
USRT	Intel Aero	$1000
Group funded	mini HDMI connectors	$15
Group funded	Multi-port USB hub	$20

VI. Conclusion

As a group, we wanted to develop a project that would be challenging and require us to work together to while problem-solving. This project will encompass many of the subjects we have covered through our coursework and require us to learn new subjects as well. The topic of machine learning and autonomous vehicles is of high interest in today's market presenting us with an opportunity to develop product applicable to many industries. The supercomputing package of the NVidia's Jetson TX1 coupled with the Intel Aero drone will offer solutions where general mapping is desired in environments that are either unexplored, unmapped, or unreachable. This will be possible by using the Intel Aero drone to travel in a given area, unrestrained by wheeled or legged components and the computing abilities of the TX1 module for image processing and as a machine/deep learning platform. In today's society, the use cases for a project as such are vast. As a group, we feel that this project is ideal given that it will pose a great technical challenge and can be tested in real-world situations once completed.

VII. Acknowledgements

We would like to thank Dr. Kriehn and Dr. Wang, our advisors that will provide essential guidance through the course of the project. Along with our advisors, this project would not be possible without the assistance of USRT and IEEE. Each organization has supplied us with the hardware necessary for the project at no cost to us.

VII. References

[1]

https://click.intel.com/intel-aero-ready-to-fly-drone.html

[2]

https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-412j-cognitive-robotics-spring-2005/projects/1aslam_blas_repo.pdf

[3]

http://www.iri.upc.edu/people/jsola/JoanSola/objectes/curs_SLAM/SLAM2D/SLAM%20course.pdf

[4]

https://earchivo.uc3m.es/bitstream/handle/10016/16099/SLAM%20FOR%20DRONES_final%20for%20revision.pdf%3Bjsessionid=4286977A9925540C823CEE4F4D673D98?sequence=2