Software Overview

Year: 2022 Semester: Fall Team: 8 Project: Hermes

Creation Date: 9/6/2022 Last Modified: September 10, 2022

Author: Jonathan Wosiak Email: jwosiak@purdue.edu

Assignment Evaluation:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Score (0-5)** | **Weight** | **Points** | **Notes** |
| **Assignment-Specific Items** | | | | |
| **Software Overview** |  | x2 |  |  |
| **Description of Algorithms** |  | x2 |  |  |
| **Description of Data Structures** |  | x2 |  |  |
| **Program Flowcharts** |  | x3 |  |  |
| **State Machine Diagrams** |  | x3 |  |  |
| **Writing-Specific Items** | | | | |
| **Spelling and Grammar** |  | x2 |  |  |
| **Formatting and Citations** |  | x1 |  |  |
| **Figures and Graphs** |  | x2 |  |  |
| **Technical Writing Style** |  | x3 |  |  |
| **Total Score** |  | | |  |

5: Excellent 4: Good 3: Acceptable 2: Poor 1: Very Poor 0: Not attempted

General Comments:

*Relevant overall comments about the paper will be included here*

1. Software Overview

From a software perspective, the microcontroller will have to deal with reading in sensor data from an accelerometer, gyroscope, magnetometer, and a barometer. These sensors are needed in order to read in all necessary information about the drone’s operation and be used to stabilize the drone.

The microcontroller will also need to transmit the necessary sensor data to the navigation computer. This will make you able to compute motor outputs via PID loops for yaw, pitch, roll, throttle, output motor commands to brushless motor controller, and the actual stabilization of the drone. The navigation computer will need to first, read image frames off stereo camera modules and take those images in order to Compute depth map from compositing the stereo images and calculating the depth from the two different cameras. The depth map will be then processed by the stereo SLAM algorithm to create a 3D point cloud of the surrounding area. This will then be translated to a 2D occupancy grid to pass into the path planning algorithm. This algorithm is D\*Lite which is created to generate and update the path. This will then be decomposed into flight commands for the microcontroller and then send those instructions to the microcontroller via Serial.

2.0 Description of Algorithms

The sensor fusion algorithms take the sensor data and combine them into creating much more comprehensive way to control all aspects of the drone itself. One example of this is a Kalman filter which takes the sensor data given and recursively uses position and speed from gyroscopes and accelerometers to predict both current and future states [8].

The PID Loop algorithm takes the sensor data given as well as results from the sensor fusion algorithm and calculates the proportional, integral and derivative (as the name suggests) responses and summing those values. This then allows for the microcontroller to create the desired output to the motors [5].

The depth map is created by taking two separate images from each of the cameras and comparing them to find the depth of each pixel from the cameras. This is simple as the distance between each of the cameras is known. From that data, you can compare the angle differential between each pixel in the frame and extrapolate that data into a distance using trigonometry that each pixel in the image is from the camera creating the needed depth map [8].

The stereo SLAM algorithm takes the previously created depth map and turns it into a 3D point cloud as well as localizes the drone in space. This is done by taking the depth map and identifying key points on the depth map in order to choose the actual points that go on the point cloud. This is then placed on a grid at the specified distance from as shown in the depth map. As more keyframes from the cameras come in, the key points are compared to their previous locations and the point cloud is then adjusted to show where the drone is in space and where the obstacles around the drone are [9].

The point cloud then gets translated into a 2D occupancy grid by taking the output of the stereo SLAM algorithm and condensing it into a format that the path planning algorithm can handle. Just like the stereo SLAM algorithm, this algorithm chooses key planes on the Z-axis and flattens them into a 2D occupancy grid. This is done by identifying which obstacles are closest in three-dimensional space and prioritizing those planes most crucially and removing duplicate obstacles on the 2D grid [10].

Lastly, the D\* Lite algorithm takes this occupancy grid and generates a path. This is done in reverse to another updating path planning algorithm, LPA\*. This algorithm takes the start and goal nodes on the occupancy grid, and creates a heuristic, which is just the shortest path assuming no obstacles. Then as the algorithm checks nodes along the heuristic from the goal to the start and encounters obstacles, it takes the greediest path to get to the start. This should be the shortest path and as the occupancy grid updates, so does the planned path making for a much more efficient operation as the path doesn’t need to be fully updated every time [6].

3.0 Description of Data Structures

The main data structures that are being used in this project are lookup tables, the STM32 C Library, a priority queue and hash maps. The lookup table is essentially just a way to store data that maps input data into output data to speed up efficiency of retrieving data [1]. The STM32 C Library is all the peripheral data that is needed to create C code to allow for the STM32’s peripheral functions to work [2]. A priority queue is just a stack of data that is not first in last out, nor first in, first out, but instead functions based on a priority that each entrant into the stack is given [3]. Hash maps are very similar to lookup tables as they also function as a means of mapping input data to output data, but the data for a hash map, as well as the means of creating the map to each input and output is completely customizable [4].

4.0 Sources Cited:

[1] “About Lookup Table Blocks - MATLAB & Simulink,” www.mathworks.com. <https://www.mathworks.com/help/simulink/ug/about-lookup-table-blocks.html>

[2] “STM32 Standard Peripheral Libraries - STMicroelectronics,” www.st.com. <https://www.st.com/en/embedded-software/stm32-standard-peripheral-libraries.html>

[3] “Priority Queue | Set 1 (Introduction) - GeeksforGeeks,” GeeksforGeeks, Dec. 03, 2018. <https://www.geeksforgeeks.org/priority-queue-set-1-introduction/>

[4] “unordered\_map in C++ STL,” GeeksforGeeks, Mar. 24, 2016. <https://www.geeksforgeeks.org/unordered_map-in-cpp-stl/>

[5] “PID Theory Explained - National Instruments,” Ni.com, 2019. <https://www.ni.com/en-us/innovations/white-papers/06/pid-theory-explained.html>

[6] S. Koenig and M. Likhachev, “D\* Lite.” [Online]. Available: <http://idm-lab.org/bib/abstracts/papers/aaai02b.pdf>

[7] “Sensor Fusion Algorithms Explained,” Udacity, Aug. 25, 2020. <https://www.udacity.com/blog/2020/08/sensor-fusion-algorithms-explained.html>

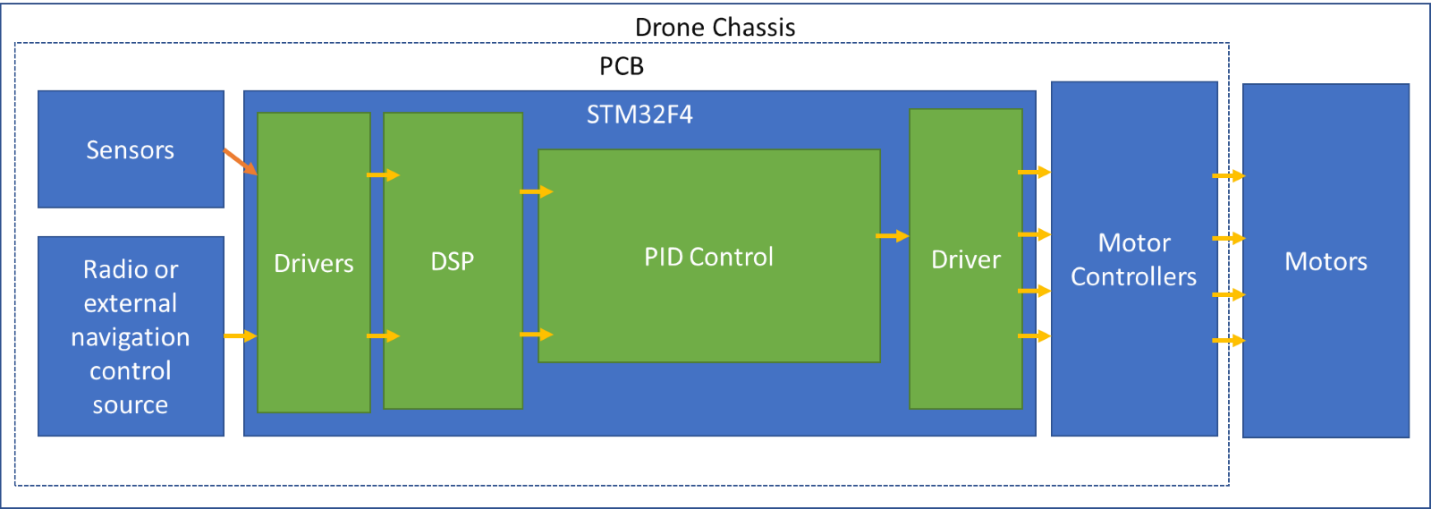
[8] A. Vij, “Stereo Vision: Making a Depth Map from scratch!,” MLearning.ai, Mar. 29, 2021. <https://medium.com/mlearning-ai/stereo-vision-making-a-depth-map-from-scratch-6cd25c82897a>

[9] S. Ye, “Stereo Visual SLAM,” Shangzhou Ye, Mar. 20, 2020. <https://shangzhouye.tech/featured-projects/stereo_slam/>

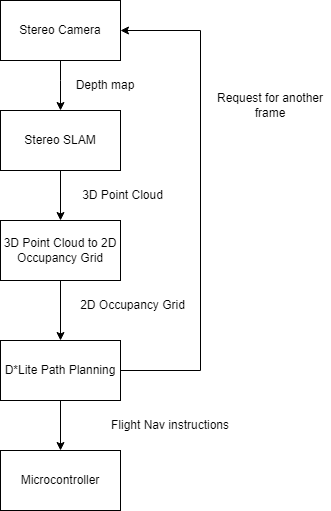
[10] J. Huesman, “Converting 3D Point Cloud Data into 2D Occupancy Grids suitable for Robot Applications,” library.ndsu.edu, 2015. [Online]. Available: <https://library.ndsu.edu/ir/handle/10365/25535>

Appendix 1: Program Flowcharts

Microcontroller Program Flowchart:

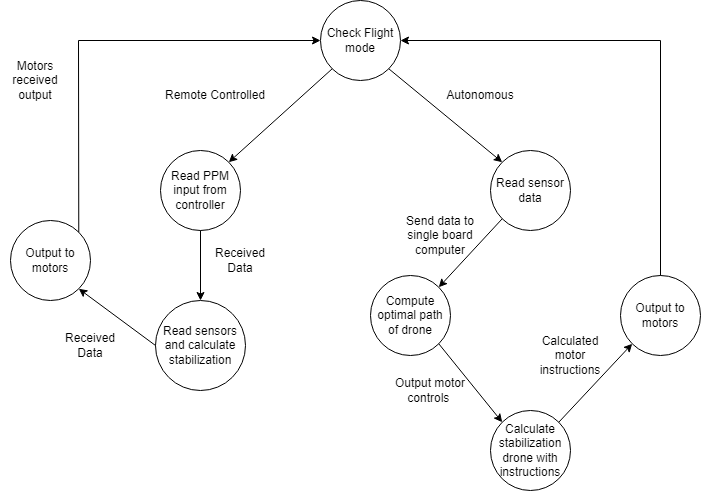


Navigation Computer Program Flowchart:



Appendix 2: State Machine Diagrams

Drone State Machine Diagram:

**