McNair Scholars Research Journal

Volume 5 Article 4

2018

Feasibility Study for the Application of Open Source UAS Autopilot Simulator

Jaime A. Ramirez Duarte

DOE/NV/25946--3329

Feasibility Study for the Application of Open Source UAS Autopilot Simulator

Jaime A. Ramirez Duarte

Department of Homeland Security Summer Intern

Remote Sensing Laboratory

August 2017

This work was done by National Security Technologies, LLC, under Contract No. DE-AC52-06NA25946 with the U.S. Department of Energy

Table of Contents

List of Figures

List of Tables

Abstract

Constant development in Unmanned Aircraft System (UAS) technologies results in evolving aircraft, instrumentation, and control programs of these systems. An open source simulator was examined as a potential method of testing UAS platforms changes. The open source simulator was used to simulate flights flown by an actual platform. Existing flight data was used as a base line of comparison for the telemetry data of the simulated flight. Patterns in parameters such as vehicle attitude and throttle output were compared between the simulated and actual results to determine if the simulator produced an accurate representation of a physical flight. Analysis of the simulated vehicle exhibited similar behavior to that of the actual vehicle in most of the flight data collected. Simplification in the flight simulator model may be responsible for the deviations in magnitude observed between the two sets of data. Three points of interest in each of the three test cases were chosen as checks for the values in the data. These points in the flight were studied in the various graphs of the parameters graphed and the difference between the simulated and actual values was calculated. The values at these points stayed within an average of 0° and 30° for vehicle attitude and between 0% and 33.3% throttle difference over the three test cases. The average difference in attitude was 5.1° over 93 compared values and 8.1% in throttle over 18 compared values. On average the simulator showed good agreement to existing flight data and therefore will be a good tool to simulate flights prior to a mission. The accuracy of the results may be improved with further development of the simulation model.

Introduction

With the growing number of affordable and commercially available Unmanned Aerial Systems (UAS), an increasing amount of people are joining the UAS community. This growth drives a demand for improvements in the systems, and also has provided a strong community that can work together to improve or develop new software/hardware for use in the Unmanned Aerial Vehicles (UAV). A big player in creating and inspiring users to join the UAS community are do-it-yourself (DIY) kits. DIY kits can be inexpensive and allow the user to build the vehicle in a custom configuration with many optional components. Many hobbyist and even commercial entities develop open source code for these vehicles. Often these developers release the work back to the open source UAS community for use by anyone.

Testing on UAS platforms to characterize vehicle performance, test changes in configurations or parameters, and debug software or flight plans is needed in order to produce better UAS's. There are three main methods for testing vehicles used today and they include physical testing, hardware in the loop (HIL), or software in the loop (SITL). Physical testing offers the closest data to actual flight out of the three because it allows all of the variables present in the actual flight to be present such as environment factors, imbalances in the system, and many more variables that might be present in the vehicle at time of flight. HIL testing is the runner up in terms of the data it produces because it allows the actual hardware to be present without requiring the vehicle to leave the test bed since a virtual vehicle is created using the same code on the physical vehicle. Finally comes SITL testing were no hardware is present, only the simulated vehicle is used in the simulation in order to observe the behavior of the code while flying. SITL testing is a helpful stepping stone when beginning a new development project on a UAV. Combining a SITL simulation with a 3D simulator allows the modeling of components, sensors, and environments which allows the SITL simulations to have even more benefits to testing by allowing more parameters to be tested.

The Remote Sensing laboratory (RSL) has long used conventional aerial platforms to collect radiological data [1]. RSL uses a fixed wing plane and helicopter carrying detectors, capable of collecting radiological data, to perform surveys of areas in an emergency response or prevention standpoint. With the growing interest in UAS programs in private and government agencies, the benefits of employing unmanned systems to collect data in potentially dangerous environments are becoming more prevalent. RSL has created a program to merge UAS platforms with radiological detectors to be used in conjunction with their larger manned aircraft. These platforms can be used to collect data in places where the radiation is too high for humans or in places where access is limited to the large aircraft used. The need to test the systems used before flying or before making changes to them is needed in order to reduce the risk of failure. This study was done in order to verify the feasibility of using open source software to simulate the UAS platforms at RSL. The study was conducted with the focus of showing the capabilities of the simulator and the validity of the simulator. Setup of the simulator was done with documentation on different SITL simulation methods [2] [3].

Simulator Components

A brief overview of the components used in the simulator will be addressed in this section. The configuration of the simulator used is not the only way to perform simulations involving virtual vehicles. An explanation of how the simulator interfaces the different software and the procedures to use the simulator are outside of the scope of this project, but the components of the simulator will be addressed briefly.

ArduPilot

The open source professional grade flight controller software, ArduPilot, was the controller chosen to be the focus of the simulator. The ArduPilot project released the first version of the software in 2009 created by Jordi Munoz and Chris Anderson [4]. The code has since been worked on by many individuals around the globe, which have provided maintenance and development for the control software. Many individuals contribute ideas and fixes to the code through GitHub, and the development team tests the code regularly to find bugs and areas that need improvement. ArduPilot is mainly written in C++ with support from Python utilities. RSL uses multiple 3DR Solo platforms to conduct missions that employ this controller, which is why this study focuses around it. Other benefits from using this software are open source allows easy development, and the code supports SITL simulations fairly easily.

MAVProxy

MAVProxy is a ground control station (GCS) software package created by the CanberraUAV team in Australia for a UAV competition. The GCS is written in Python allowing it to be run on multiple operating systems and allowing for easy development. Many modules have been developed for use in UAS operations including a real time graphing tool, a log analyzer, gimbal support, controller support, and many more. MAVProxy runs with any flight controller software using the MAVLink protocol and features a simple command line interface. Benefits to using MAVProxy as the main GCS in the simulator include developed modules, opening ports to communicate with other software, portability, and good documentation on SITL procedures.

Mission Planner

Developed by Michael Oborne to be used with the ArduPilot Mega (APM) flight controller hardware, Mission Planner (MP) is a popular GCS used in UAS operations that can now interface with multiple different hardware. MP is an easy to use GCS that offers many tools to prepare, plan, and execute missions using a graphical user interface (GUI). There are multiple uses for MP such as easy flight planning, changing parameters on the vehicle, running a SITL simulation of flight plans, and monitoring of vehicles in flight. MP was used in this simulator for the ease of creating flight plans using its Flight Planner tab, the built in log analyzer, and the GUI interface for monitoring flights.

Gazebo

Gazebo is a robotics simulator created by Dr. Andrew Howard in 2002 at the University of Southern California [5] used by many robotics teams across the country and was even used by the Open Source Robotics Foundation (OSRF) in the DARPA Robotics challenge of 2013. Gazebo offers the ability to model whole environments and robots for use in the simulations. It supports four robust physics engines, high-quality graphics, support for sensor modeling, development for plugins to interface with the application program interface, and many more features. Recently a plugin and vehicle model were developed for interfacing Gazebo with an ArduPilot SITL using TCP messaging. The vehicle model was of an iris quadcopter with a camera mount which is very similar in size and geometry to the 3DR Solo employed by RSL. The plugin allowed the full integration of Gazebo in the SITL simulation process to model the virtual vehicle in 3D and the environment. The physics engines used by Gazebo also allow the possibilities of modeling wind, collisions, and sensor movement controlled by SITL.

Experiment

In order to test the accuracy and validity of the simulator, three test cases of actual flights conducted by RSL were chosen to be simulated. The chosen cases all had telemetry logs available, and each of the missions used basic commands such as Takeoff, go to Waypoint, Loiter, Loiter Turn, and Return to Launch (RTL). These cases also used various altitudes throughout the flight, which allowed the opportunity to simulate climb and descent of the vehicle.

Before simulations were conducted, the world model was created in gazebo as well as modifying the vehicle model to fix some minor issues. The quadcopter model had minor flaws in the collision box, used by Gazebo to determine the point of contact with another model, and the weight property used was slightly different than the weight of the 3DR Solo. These parameters were altered in the model file for use in any gazebo worlds. The world model consisted of a simple flat plane with roads, grass, telephone poles, and some buildings. The GPS coordinates of an open area outside of Las Vegas was used as a reference frame for planning flight plans and initiating the virtual vehicle. These coordinates were overlaid in the world model to be used as the spherical reference frame for flight. An aerial view of the world is shown in Figure 1.

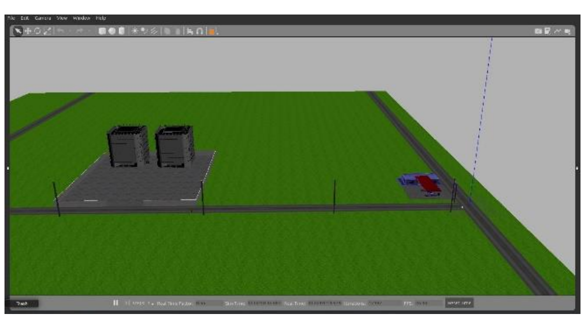

Figure 1: Simulated world used in Gazebo.

The next step in the experiment was making the flight plans using the MP Flight Plan tab. This task was done by extracting the waypoints from the telemetry logs and transposing the points to the coordinates that matched the simulator coordinates used in the world model. In one case where the telemetry logs did not contain waypoints, a Keyhole Markup

Language (KML) file was created in order to have a guideline on where the vehicle flew. From the KML and telemetry data, the waypoints were created by hand in the MP flight plan tab. The flight plan for test case one is shown below in Figure 2 while the other two flight plans can be found in Appendix A.

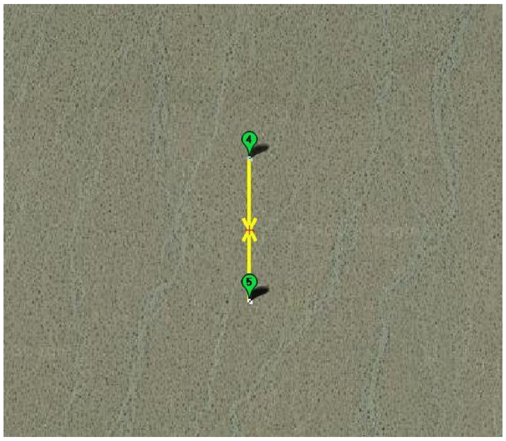

Figure 2: Flight plan for test case 1.

Once the flight plans were prepared the simulations were run using the simulator package setup. MAVProxy was used as the main GCS that created the simulated vehicle, connected to MP and Gazebo, graphed telemetry data in real time, and collected the telemetry data of the flight. MP was used as a secondary GCS that provided a simple way to upload the flight plans to the vehicle and initiate the mission. Gazebo was the 3D simulator that contained the physics engines for calculating the forces on the vehicle during flight as well as receiving and transmitting information with MAVProxy throughout the flight. Once all three programs were ready and the simulated vehicle finished its Pre-Arm checks, MP was used to upload the flight plans onto the vehicle and begin the mission. From there, Gazebo and MAVProxy were used to monitor the vehicle during flight with Gazebo providing a visual of the behavior while MAVProxy was used to graph the telemetry data in real time in order to monitor the vehicle attitude.

One telemetry log was collected for each of the test cases used in the simulator. These logs were used as the basis with which the simulator would be judged on its validity. Two tools were used when analyzing and producing the resulting graphs of the data. The first tool used was the log analyzer from MP, which allows telemetry logs and dataflash logs to be read and graphed while displaying helpful information about the data. This tool was mainly used to collect the values at the points selected in each flight and the minimum, maximum, and mean values of the overall data. The second tool used for creating graphs from telemetry data was the log analyzer that is included with MAVProxy, called MAVEXPLORER. This tool reads all types of log files and creates graphs of the data while also allowing mathematical expressions to be used and altering of the ranges to be graphed. The graphs shown in this section were created in MAVEXPLORER.

The parameters focused on in each test case were Desired Pitch, Actual Pitch, Desired Roll, Actual Roll, Actual Yaw, Error in Roll, Error in Pitch, and Throttle. Desired Roll and Pitch are the values that the vehicle has received from the pilot or autopilot while Actual Roll, Pitch, and Yaw are the values of the parameters that the vehicle has at any time. The control processes of the vehicle will attempt to make the Desired and Actual values as close to each other as possible. Error in Roll and Pitch is the difference of the Desired and Actual values showing a rough representation of how well the autopilot is controlling the vehicle. Throttle will represent the value of how much power is being output by the motors.

Results and Discussion

The collected data in the telemetry logs of the simulated runs are analyzed in the following section. The results of Test Case 1 & 2 are analyzed in depth while the analysis for Test Case 3 is included in Appendix A.

<u>Test Case 1</u>

Test case one involved a flight in which the vehicle took off, moved to a location, loitered for 60 seconds, and returned to launch. On the actual flight, the pilot took manual control at the end of the flight to return to launch and land. This portion of the flight could not be recreated in the simulator but was planned like a regular return to launch. Data towards the end of the flight will differ because of the difference in the autopilot control and the manual control. The points of interest chosen for this test case were centered on the time when the vehicle was moving to the loiter point. The area covers the time where the vehicle descends to the loiter altitude and the time where the vehicle stabilizes itself to hover over a point.

<u>Graph Patterns</u>

The generated graphs, using the telemetry data, are the first focus of analysis. The first aspect to analyze are the patterns of behavior exhibited by the vehicle during flight. The first graphs to be analyzed contain the data for Desired Pitch and are shown in Figure 3.

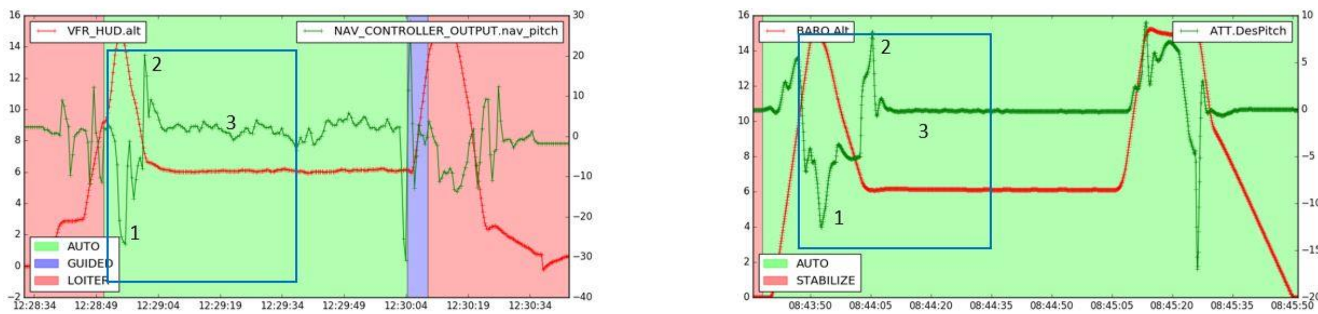

Figure 3: Graphs of Desired Pitch of actual vehicle (left) and Desired Pitch of simulated vehicle (right) in degrees.

From the graphs, the overall shape of the data follows the same trends. In the actual flight, it is evident when the vehicle takes off the pitch will fluctuate then change to a negative pitch signifying a forward movement. In the simulation, the same behavior is observed where the pitch fluctuates at the start of the flight then shifts negatively, point 1, signifying the forward movement. In both graphs, the sharp peak in the positive direction at point 2 shows the vehicle stopping at the loiter point and then attempting to loiter in place while using small pitch changes. The end of the flights differ greatly in terms of the Desired Pitch because of the different control modes used. In the actual flight, the pilot took control of the vehicle to land it while in the simulation the autopilot had control of the vehicle. Figure 4 illustrates the Pitch experienced by the vehicle.

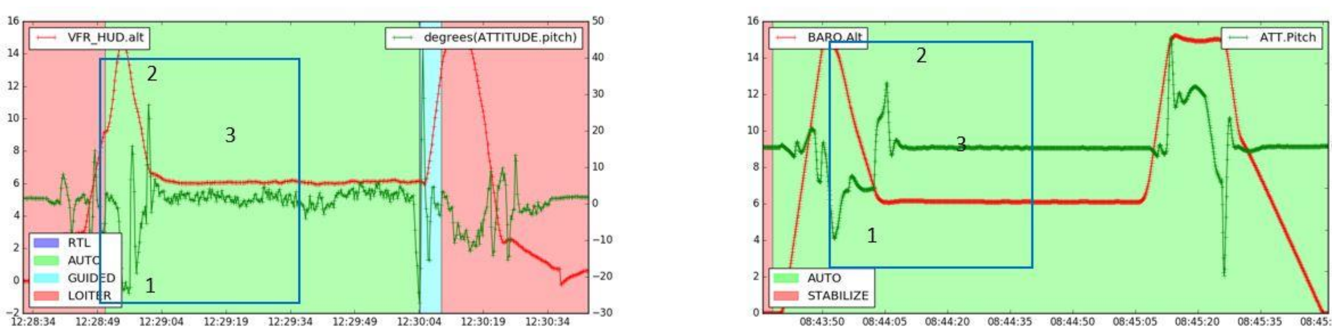

Figure 4: Graphs of Pitch of actual vehicle (left) and Pitch of simulated vehicle (right) in degrees.

The graphs of the Pitch experienced by the vehicle follow the same patterns as the Desired Pitch. The values differ slightly than what is observed in Figure 3 because of the lag in control, environmental effects, and over corrections by the controller. Again, as observed in the Desired Pitch graphs, the data from the actual and simulated flights differ greatly at the end of the flight because of the input by the pilot. Figure 5 illustrates the Desired Roll of the vehicle.

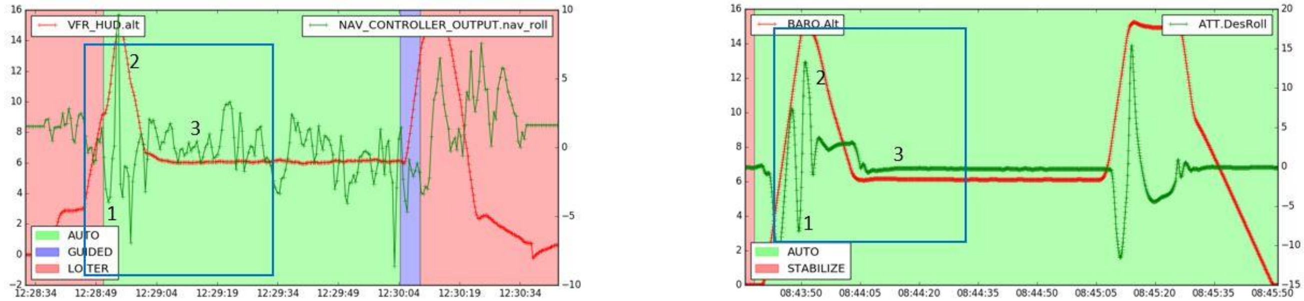

Figure 5: Graphs of Desired Roll of actual vehicle (left) and Desired Roll of simulated vehicle (right) in degree

Roll will dictate if the vehicle moves side to side. A positive roll will signify the vehicle turning or moving to the left and a negative roll will signify movement to the right. In both flights, it is evident that the vehicle is attempting to adjust its position while moving towards the target point. Once the target point is reached the vehicle attempted to maintain its position over the point while loitering by oscillating roll around 0. The oscillations are stronger in the actual flight mainly because of the environmental factors acting on it. Once the loiter is over, the graphs clearly show a large negative roll which is indicative of the turn the vehicle makes to return to launch. Both of the graphs show a very similar behavior pattern over the whole flight except after the vehicle turns to return home where in the actual flight the pilot took over. The next data graphed was the actual Roll experienced by the vehicle during flight and is shown in Figure 6.

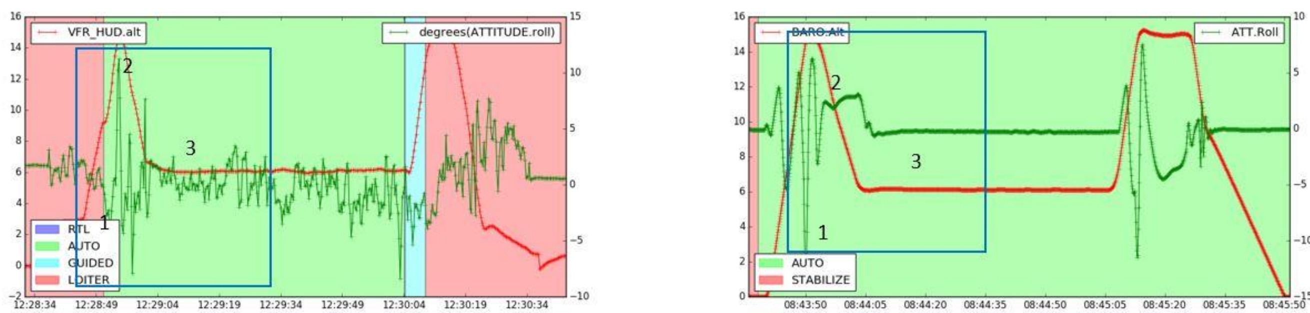

Figure 6: Graphs of Roll of actual vehicle (left) and Roll of the simulated vehicle (right) in degrees.

The patterns seen in the Roll experienced by the vehicles are very similar to the Desired roll. The largest difference between the graphs of desired Roll and Roll is the noise seen in the Roll graphs. This noise can be attributed to the environmental forces experienced during flight and the small errors in the controller propagating into the attitude of the vehicle. The next data graphed was the throttle of the vehicles shown in Figure 7.

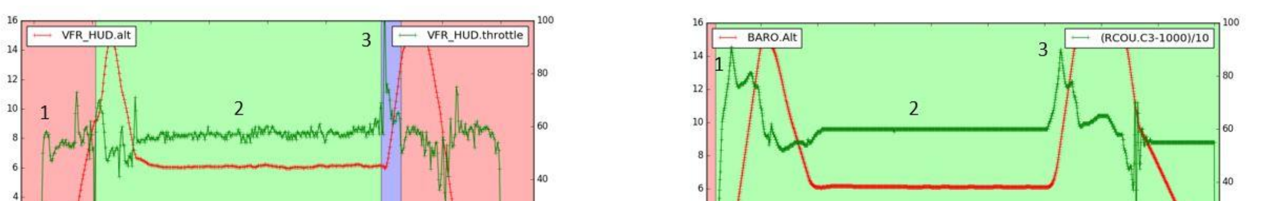

Figure 7: Graphs of Throttle of actual vehicle (left) and Throttle of the simulated vehicle (right).

The Throttle of the vehicle is a measure of how much power is being output by the motors. This is calculated based on the values of pulse-width-modulation (PWM) signals set on the vehicle to be the minimum and maximum values. Once set an input PWM of the minimum value will signal to the motor to output 0% of its power while an input of the maximum value will signal the motor to output its full power. In the graphs above the throttle was calculated already by the telemetry log in the actual flight and in the simulated fight the calculation had to be done by hand. The pattern observed in the flights involved a sharp increase for the takeoff, climb, and then a constant value was used during the loiter. At the end of the flights, the values jumped up to climb again and then dropped to return for a landing. Figure 8 shows the plotted data of the Error in Roll.

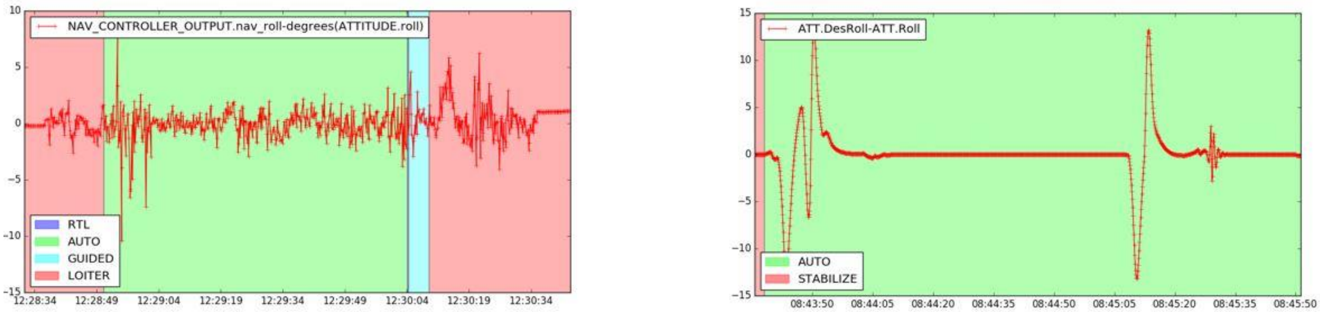

Figure 8: Graphs of Error Roll of actual vehicle (left) and Error Roll of simulated vehicle (right) in degrees.

Error in Roll is the difference between the Desired Roll and the Roll of the vehicles. It is a robust way of showing the efficiency of the flight controller in communicating with the hardware. The smaller the values observed on the graphs the more efficient the flight controller is at controlling the vehicle. The difference between the actual flight controller and the simulated one can be seen in the graphs shown in Figure 8. The pattern seen in the actual flight data is of small errors in the roll until the vehicle reached the first target altitude and began to descend. At the end of the flight, again some larger errors were experienced. The simulated flight graphs shows what seems to be a completely different pattern. There are very large errors when climbing and descending on the first part of the flight and then again at the end of the flight when ascending to return to launch. These differences show some error in the simulator but the source of the error could not be found with so few test cases. The last portion of data graphed was the error in Pitch shown in Figure 9.

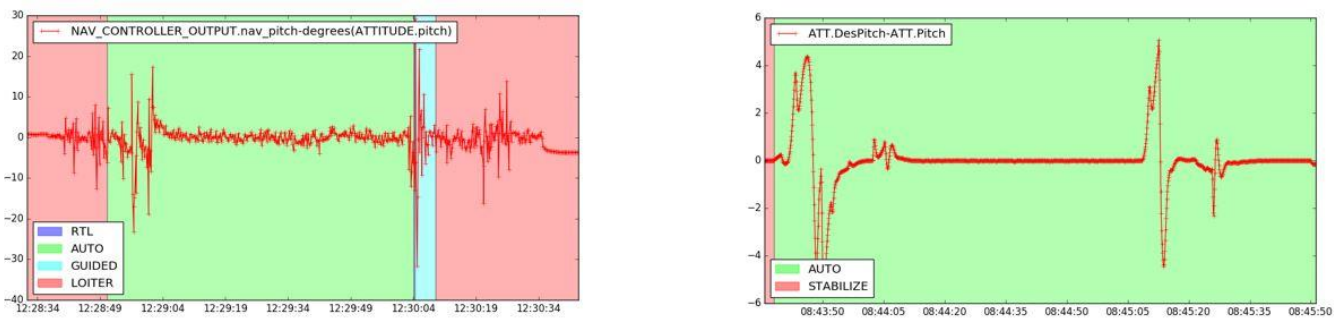

Figure 9: Graphs of Error Pitch of actual vehicle (left) and Error Pitch of simulated vehicle (right) in degrees.

Error in Pitch is exactly like Error in Roll where the smaller the error the more efficient the flight controller was during flight. Analyzing the pattern in the actual flight there were larger errors occurring when the vehicle was attempting to stop moving at the beginning of the flight. At the end of the flight there is another spike in error when the pilot took control of the vehicle to move back to launch. The simulator graph shows similar patterns but the beginning pike happens right when the vehicle takes off instead of when attempting to slow down. The spike at the end of the flight happens at the same time where the vehicle finishes to loiter on the point and move back to launch.

Overall, the patterns in the actual flights and the simulated flights were very similar with differences in only magnitude. The only graphs that displayed completely different patterns were the Error in Roll graphs, and the Error in Pitch graphs had a slight difference in the pattern.

Graph Values

The values of the numbered points in the graphs were extracted from both of the flights and compared. This comparison was done by finding a simple difference between them and a percent difference in order to illustrate how the simulator fared against the actual flights. The values for the graphs shown in Figures 3-9 are displayed in tables 1-6. In tables 1-4 the values of the three points of interest of the actual flight and simulated flight are shown as well as the min, max, and mean of the graphs while in tables 5-6 only the min, max, and mean are shown.

Table 1: Values obtained from Desired Pitch graphs.

	Actual	Simulated	Diff	% Diff
1. Peak	-26.8	-12.6	14.2	53.0
2. Peak	19.6	8.2	11.4	58.2
3. Plateau	1.5	0	1.5	100.0
Min	-27	-17	10	37.0
Max	29	9	20	69.0
Mean	0	0	0	0.0

Table 2: Values obtained from Pitch graphs

	Actual	Simulated	Diff	% Diff
1. Peak	-25.4	-11.3	14.1	55.5
2. Peak	26.2	7.5	18.7	71.4
3. Plateau	1.6	0	1.6	100.0
Min	-26	-15	11	42.3
Max	43	13	30	69.8
Mean	0	0	0	0.0

Table 3: Values obtained from Desired Roll graphs.

	Actual	Simulated	Diff	% Diff
1. Peak	-4.0	-7.9	3.9	97.5
2. Peak	9.6	12.7	3.1	32.3
3. Plateau	-0.3	0	0.3	100.0
Min	-9	-11	2	22.2
Max	10	15	5	50.0
Mean	0	0	0	0.0

Table 4: Values obtained from Roll graphs.

	Actual	Simulated	Diff	% Diff
1. Peak	-3.6	-11.8	8.2	227.8
2. Peak	10.9	6.2	4.7	43.1
3. Plateau	-1.0	0	1	100.0
Min	-8	-13	5	62.5
Max	11	8	3	27.3
Mean	0	0	0	0.0

Table 5: Values obtained from the Throttle graphs.

	Actual	Simulated	Diff	% Diff
1. Peak	57.8	91.1	33.3	57.6
2. Plateau	57.5	59.8	2.3	4.0
3. Peak	100	89.3	10.7	10.7
Min	0	0	0	0.0
Max	100	91.1	8.9	8.9
Mean	50	59.4	9.4	18.8

Table 6: Values obtained from Error Roll graphs.

	Actual	Simulated	Diff	% Diff
Min	-10.4	-13.2	2.8	26.9
Max	8.1	13.5	5.4	66.7
Mean	0	0.3	0.3	-

Table 7: Values obtained from Error Pitch graphs.

	Actual	Simulated	Diff	% Diff
Min	-32	-5.6	26.4	82.5
Max	29.2	5	24.2	82.9
Mean	0	0	0	0.0

In Tables 1-7, the main values to observe are the differences in the numbers obtained from the graphs. This difference shows how far the simulator values were from the values obtained from the real flight, which is an indicator of the quality of the simulator. When paired with the comparison of the overall patterns in the graphed data a conclusion about the effectiveness of the simulator can be reached. The difference in the values of vehicle attitude ranged from 0° to 30° and of the throttle output between 0% and 33.3%. The average error was in attitude of the vehicle was 7.6° while the throttle average was 10.8%. 9 out of the 30 attitude values compared were above 10° while throttle only had 2 values out of 6 above 10%.

Test Case 2

Case 2 was a Loiter turn around a point of interest. The vehicle takes off and moves to the target point. Once the location is reached the vehicle moves to perform a turn of 100m radius around the point three times at three different altitudes before returning to launch. The points of interest analyzed were located on the time where the vehicle performs the first turn. This choice was made because the same pattern observed during the turn is repeated in the other two.

Graph Patterns

The first graphs to be analyzed were the Desired Pitch of the vehicles; Figure 10 shows the plotted data as well as the locations of the points of interest in Test case 2. The patterns of all of the resulting graphs will be analyzed in this section.

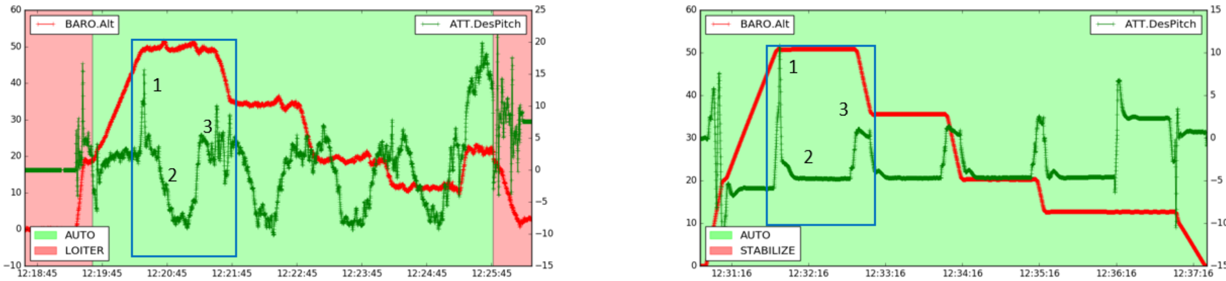

Figure 10: Graphs of Desired Pitch of actual vehicle (left) and Desired Pitch of simulated vehicle (right) in degrees.

Analyzing the patterns in the Desired Pitch, it is evident that the vehicle has to pitch negatively to move forward and pitch positively to slow down or stop. During the circle, the pitch gradually pitches negatively more until the time to slow down comes. The simulated vehicle exhibits almost the same exact pattern with the exception of the gradual pitching motion during the turn. The simulated vehicle averaged the pitching motion and pitched constantly throughout the turn. Actual Pitch of the vehicle during flight was the next parameter graphed and analyzed. The graphs are displayed in Figure 11.

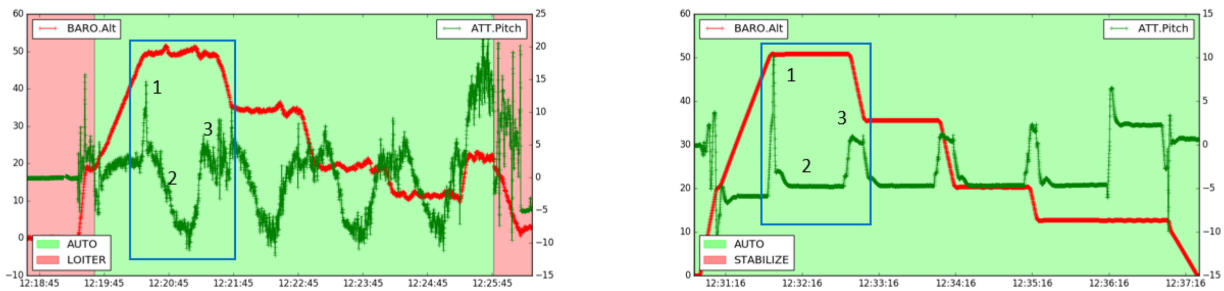

Figure 11: Graphs of Pitch of actual vehicle (left) and Pitch of simulated vehicle (right) in degrees.

The patterns seen in the actual and simulated vehicles are the same as the Desired Pitch graphs with the addition of some noise to both of them. The noise is especially prevalent in the actual flight and can be a contributing factor to any error present in the values at the points of interest. The noise seen can be attributed to the vehicle being acted on by an external source and the autopilot then trying to correct for this error. The graphs of the Desired Roll of the test case are shown in Figure 12 below.

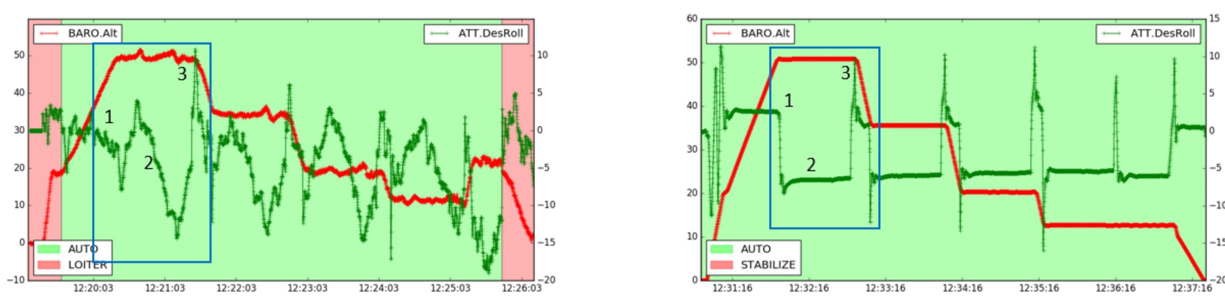

Figure 12: Graphs of Desired Roll of actual vehicle (left) and Desired Roll of simulated vehicle (right) in degrees.

In the graphs above a similar pattern to that of Pitch is observed. The vehicle will want to be at a Roll value of zero while not turning or moving side to side, roll positively for a left turn, and negatively for a right turn. The actual flight data shows the autopilot wanting to maintain zero roll during its climbs or descents while trying to correct for any perturbation and then begin to roll negatively as it performs the right turn around the point. On the other hand, the simulated vehicle almost achieves a Roll value of zero during climbs and descents and a constant Roll angle while turning. This is because of the simplified environment in which the vehicle is flying in. the overall pattern is very close to that of the actual flight. The Roll experienced by the vehicles is shown in Figure 13.

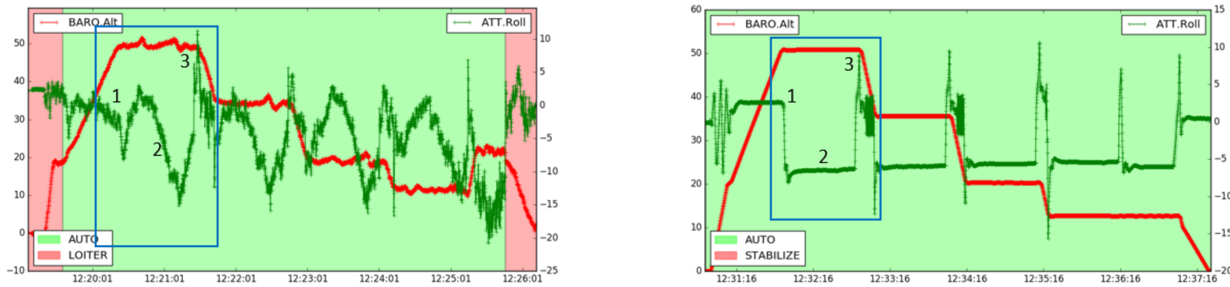

Figure 13: Graphs of Roll of actual vehicle (left) and Roll of simulated vehicle (right) in degrees.

Patterns in the Roll experienced during flight of both vehicles showed the same behavior as the Desired Roll. The Roll shows some noise in the graph again showing the effects of the environment and other external variables on the attitude of the vehicle. The simulator also averages the slopes observed in the actual flight to maintain a constant roll throughout the turns. The Yaw of the vehicle in both flights is shown in Figure 14.

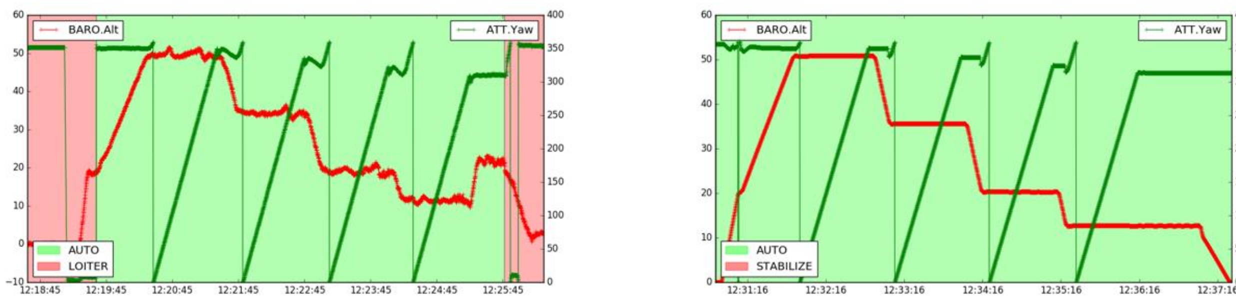

Figure 14: Graphs of Yaw of actual vehicle (left) and Yaw of simulated vehicle (right) in degrees.

Vehicle Yaw describes the orientation that the vehicle is facing and effects only the signs of the Roll and Pitch values when performing a maneuver. The vehicle was set to face the next waypoint at all times in the flight in both the simulation and the actual flight. Because of this parameter, the graphs of both the actual and simulated flights look almost identical. As the vehicle is performing the turn around the point, the controller will turn the craft to face towards the direction of motion resulting in the constant slope change in yaw. The throttle output by the vehicle is graphed in Figure 15.

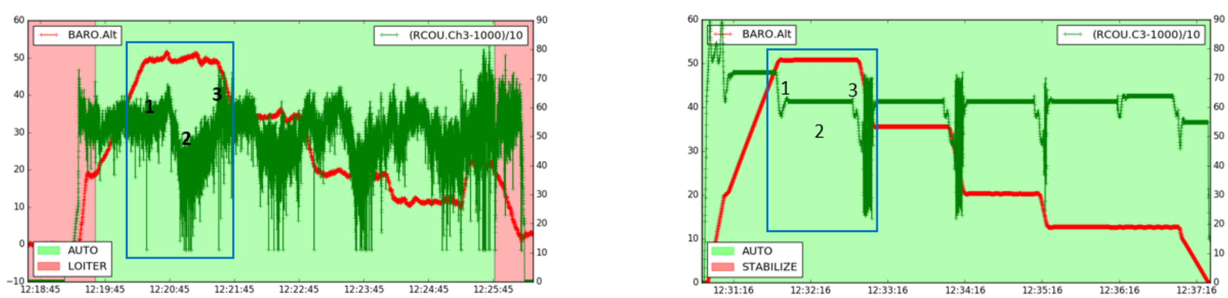

Figure 15: Graphs of Throttle of actual vehicle (left) and Throttle of simulated vehicle (right).

Throttle output by the vehicle followed a similar pattern in both of the flights. The major difference in the two flights can be seen in the throttle output during the turn. In the actual flight, the throttle stays constant when climbing or descending and fluctuates during the turn to maintain the desired altitude. In the simulator, the throttle will be maintained constant during the turn since there are no external factors affecting its flight. The constant value held by the simulator is close to the average value of throttle in the fluctuation. Assuming the simulator idealizes the throttle output during flight, then the vehicle maintains the same pattern throughout the flight. Error in Pitch was plotted and the graphs are displayed in Figure 16.

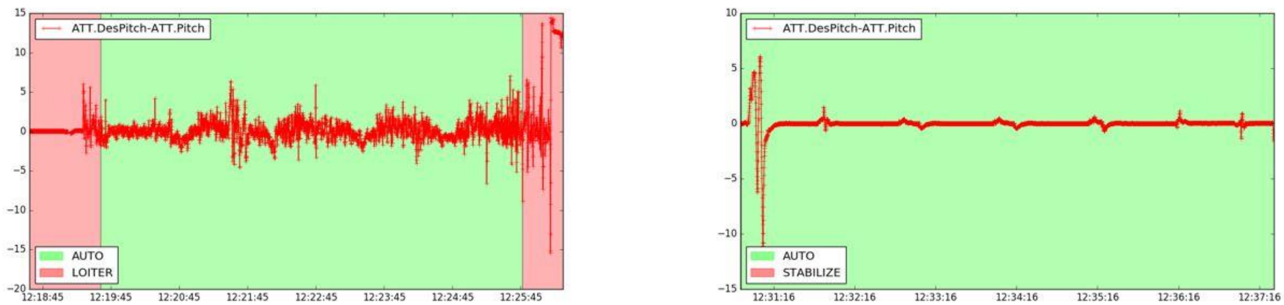

Figure 16: Graphs of Error Pitch of actual vehicle (left0 and of Error Pitch of simulated vehicle (right) in degrees.

The difference from the Desired Pitch and the Pitch of the vehicle measured by the sensors stays centered around zero with some peaks at various times in the flight. The first peak can be observed at the beginning of the flight where the vehicle takes off in both graphs at around the same magnitude. In the actual flight, the Error in Pitch has scattered jumps periodically around the time where the vehicle climbs or descends. Small fluctuations can be seen in the simulated data at the same times but not to the same magnitude. In the actual flight, there are large peaks at the end of the flight possibly caused by the manual flight. The last data to be graphed is the Error in Roll shown in Figure 17.

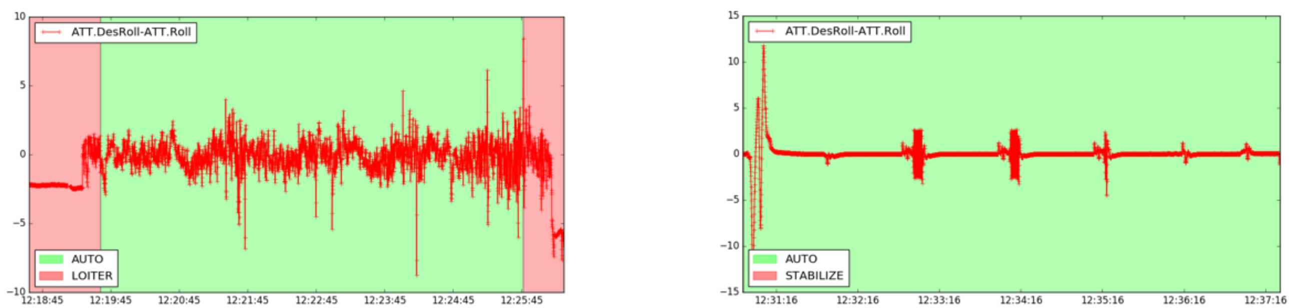

Figure 17: Graphs of Error Roll of actual vehicle (left) and of Error Roll of simulated vehicle (right) in degree.

The Error in Roll of the vehicles follows a similar pattern to that observed in the Error in Pitch graphs. The big exception to this is the initial 5° error in the actual flight. This error can be attributed to the telemetry log including three flights as well as transportation between them. Had the graphs started right as the vehicle completed the alignment the graph would have started at zero.

Graph Values

The same process that was done in test case 1 was applied to the graphs in test case 2 in order to obtain the desired values and calculation of the differences in the values was conducted. The resulting data is displayed in tables 8-15 showing the calculations.

Table 8: Values obtained from Desired Pitch graphs.

	Actual	Simulated	Diff	% Diff
First Peak	15.3	10.7	4.6	30.1
Slope Avg.	-3.1	-4.8	1.7	54.8
Plateau	2.2	0.5	1.7	77.3
Min	-10	-13	3	30.0
Max	22	11	11	50.0
Mean	1	-3	4	400.0

Table 9: Values obtained from Pitch graphs.

	Actual	Simulated	Diff	% Diff
First Peak	14.4	10	4.4	30.6
Slope Avg.	-3.2	-4.8	1.6	50.0
Plateau	1.7	0.3	1.4	82.4
Min	-12	-11	1	8.3
Max	22	10	12	54.5
Mean	1	-3	4	400.0

14

Table 10: Values obtained from Desired Roll graphs.

	Actual	Simulated	Diff	% Diff
Slope Avg.	-1	2.3	3.3	330.0
Slope Avg.	-5.2	-6.5	1.3	25.0
Peak	10.7	9.5	1.2	11.2
Min	-19	-16	3	15.8
Max	11	11	0	0.0
Mean	4	-3	7	175.0

Table 11: Values obtained from Roll graphs.

	Actual	Simulated	Diff	% Diff
Slope Avg.	-1	2.6	3.6	360.0
Slope Avg.	-6.4	-6.4	0	0.0
Peak	11.2	8.7	2.5	22.3
Min	-21	-16	5	23.8
Max	11	11	0	0.0
Mean	-3	-3	0	0.0

Table 12: Values obtained from Yaw graphs.

	Actual	Simulated	Diff	% Diff
Min	0	0	0	0.0
Max	360	360	0	0.0
Mean	234	249	15	6.4

Table 13: Values obtained from Throttle graphs.

	Actual	Simulated	Diff	% Diff
Plateau	57.1	71.3	14.2	24.9
Plateau	52.5	62.5	10	19.0
Peak	72.4	71.8	0.6	0.8
Min	0	0	0	0.0
Max	81.4	90.7	9.3	11.4
Mean	46.9	67.00	20.1	42.9

Table 14: Values obtained from Error Roll graphs.

	Actual	Simulated	Diff	% Diff
Min	-8.7	-13.1	4.4	50.6
Max	8.4	11.6	3.2	38.1
Mean	-0.1	0	0.1	100.0

Table 15: Values obtained from Error Pitch graphs.

	Actual	Simulated	Diff	% Diff
Min	-15.5	-11.1	4.4	28.4
Max	14.4	6	8.4	58.3
Mean	0.4	0	0.4	100.0

The tables display the calculated values for difference between the actual and simulated flights as well as the percent difference between them. The numbers to be focused on are the raw differences between the two values. In Test case 2 the range of vehicle attitude values went between 0° and 15° with 3 values out of 33 being over 10°. The values compared for Throttle ranged between 0% and 20.1% with 2 out of 6 values being over 10%. The average difference in vehicle attitude values was 3.4°, and 9.0% for throttle.

Test Case 3

A final test case was studied that involved a square pattern flight as seen in Appendix A. During this flight, the vehicle takes off and hovers at a point before moving to perform a full square pattern before returning to the point again. This case allows clear turn patterns to be observed as well as the throttle changes for climbing and descending during movement. The results of this case will be summarized because of its similarity to the previous two Test Cases.

Graph Patterns

Like the Patterns and behaviors observed in the previous two test cases, the data collected from the test case 3 simulation followed similar patterns to that of the actual flight. Desired Pitch and Pitch showed again the forward and reverse movement with the only notable difference being in the sign of the pitch on the second leg of the box. In this part of the actual flight, the pitch took a positive pitch to move forward when negative had been the convention before. This changed could be attributed to the front of the vehicle facing away from the waypoint for this portion causing the sign to be flipped. This flip was not seen in the simulated flight where the pitch was maintained negative throughout movement. In Desired Roll and Roll, the values oscillate around zero for most of the flight except in the corners and turns of the plan. In these areas the vehicle performs right turns with a combination of roll and pitch to face the next waypoint. This behavior is outlined by the peaks at these points. A sign change occurs in this test case where positive roll signifies a right turn and a negative pitch signifies a left turn. Patterns in the Throttle output of the vehicles were very similar with the exception of the magnitudes and the peak at the beginning of the flight. The simulated vehicle showed a significantly higher initial throttle output at takeoff while the actual flight showed a smaller throttle output. Like test cases 1 and 2, the Error graphs show deviations in patterns as the error in the actual flights was larger at all times than the simulated flights except during takeoff.

Graph Values

The values from the graphs created from test case 3 data were analyzed in the same manner the previous two cases data was. Three points of interest were chosen and values at these points were collected and compared between the actual and simulated data. The values of vehicle attitude in test case 3 ranged from 0° to 14.2°. Out of the 30 values compared only 3 were over 10°. The Throttle values compared ranged between 0% and 11.1% with only 1 value over 10%. The average difference was 4.4° for attitude and 4.5% for throttle.

Extra Simulations

Some additional simulations were run to showcase the various conditions that could be simulated. The four cases tested were: GPS fail, Remote Control (RC) fail, Battery fail, and a strike during flight. In the GPS fail and RC fail the SITL parameters for testing these was manually flagged during flight to observe if the vehicle would initiate the failsafe programmed into the flight controller. The Battery fail test was conducted in a similar fashion except the voltage reading of the battery was manually altered to simulate a battery failure. Finally the strike during flight was conducted using the 'Apply Force' tool in Gazebo. The Apply Force tool allows the user to apply a force to a model at any time and the physics engines will calculated the changes on the model. Three strikes during flight were conducted at different locations on the aircraft. The first one was through the center of mass, the second at the edge of the motor arm, and the last was at the edge of the vehicle leg. The locations of the applied forces are shown on the modeled vehicle in Figure 18.

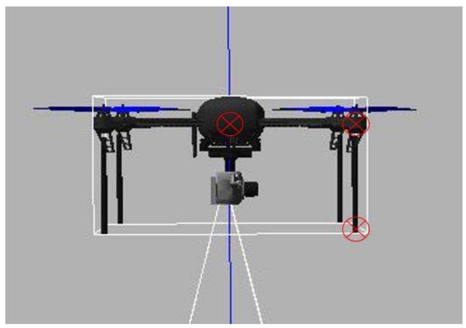

Figure 18: Strike locations on the vehicle model.

Conclusion

The results of the simulations conducted showed that the behavior of the simulated vehicle resembled the behavior of an actual vehicle. The majority of the graphed data followed the same trends and patterns when flying the same flight paths with only a few graphs showing large differences. The graphs that showed the largest differences were the Error Roll and Error Pitch graphs in all three test cases. These graphs show the error in the vehicle attitude, which is influenced largely by the effects of external forces, imbalances in the aircraft, or overcompensations made by the flight controller.

When analyzing the values at the chosen points of interest in the test cases, the main value focused on was the difference from the actual flight. This value is a key indicator to how accurate the simulator was to producing a similar flight to that of an actual flight. The vehicle should behave in the same way as a real flight to be an accurate representation of what would occur. The differences in the vehicle attitude values ranged between 0° and 30°. The 30° value appeared in the Pitch graph of test case 1 when comparing the maximum values. The maximum value in the actual flight was 43° when the pilot took control of the vehicle while in the simulator the vehicle only reached a maximum value of 13°. The range of the throttle comparisons done was between 0% and 33.3%. The 33.3% value was observed in the throttle graph of test case 1 at the initial climb. The simulator showed a high value of 91.1% throttle while in the actual flight the vehicle climbed at only 57.8%. The average difference between the simulated and actual values across the 93 comparisons in vehicle attitude was 5.6° with only 15 values being above 10°. The average difference between the 18 compared throttle values was 8.1% with 5 values being above 10%.

These results show that the open source simulator being studied produces flight data that is accurate to an average of 5.6° in vehicle attitude and 8.1% in throttle. From ranges given by the UAS pilots when beginning the study, these values are acceptable for the simulator to be used as a tool to test UAS platforms before flights. The accuracy of the simulator can be improved by continuing development in modeling environmental factors, better vehicle models, and other parameters.

Future Work

Throughout the work on the project, many new questions and issues were brought to light. The first issue needed to be addressed is to increase the accuracy of the simulator by developing better vehicle models closer to the system specifications of the comparison vehicle and working to use the physics simulator on Gazebo to its full potential. One of the key elements of Gazebo and the SITL that was not debugged was simulating wind in the environment. This feature can be used by working to integrate the Gazebo wind calculations with the SITL wind calculations in the code. The next work to be done is to model an environment in Gazebo using Digital Elevation Model (DEM) data in order to test terrain following in the UAS. If a world can be created using DEM data and the information added to the packets being exchanged between the different programs then the terrain following feature of the vehicle can be tested and debugged in this simulator.

Acknowledgments

The author of this paper would like to thank the Remote Sensing Laboratory for hosting this research and provided support in every aspect of the study. The insights and feedback from the UAS program team was invaluable to the study conducted. The Department of Homeland Security for providing the Internship and funding for this research, and the McNair Scholars Program for its continued assistance and guidance on developing professionalism. To Dr. Sathya Gangadharan for the guidance and support he provide as a faculty mentor in all areas of research and professional development.

Works Cited

1. National Security Technologies. Remote Sensing Laboratory. Retrieved from: http://www.nnsa.gov/pages/facilities/RSL.html
2. ArduPilot Dev Team (2016). Setting up SITL on Linux. Retrieved from: http://ardupilot.ord/dev/docs/setting-up-SITL-on-Linux.html.
3. Open Source Robotics Foundation Github (2017). From Source (Ubuntu and Mac). Retrieved from: http://gazebosim.org/tutorials?tut=inistall_from_source&cat=install
4. Open Source Robotics Foundation (2017). Gazebo History. Retrieved from: http://gazebosim.org

Appendix A

Figures 19 & 20: The extra graphs collected from all of the simulations not presented in the results section

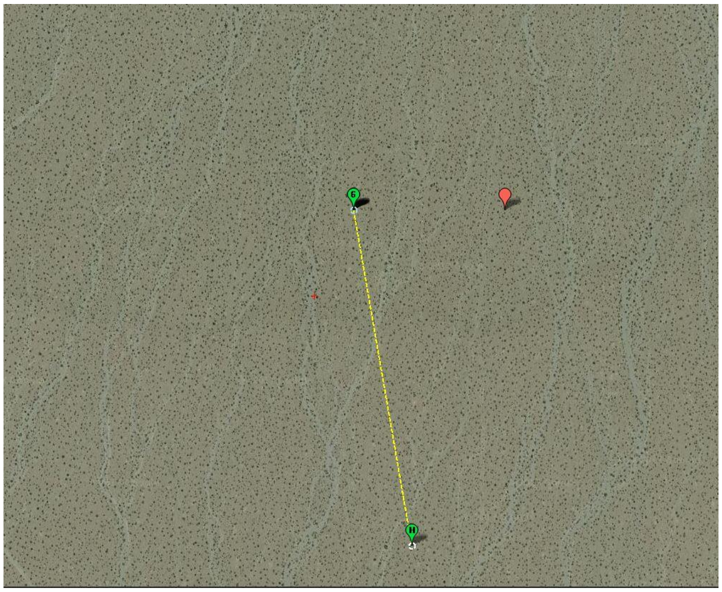

Figure 19: Flight Plan for test case 1.

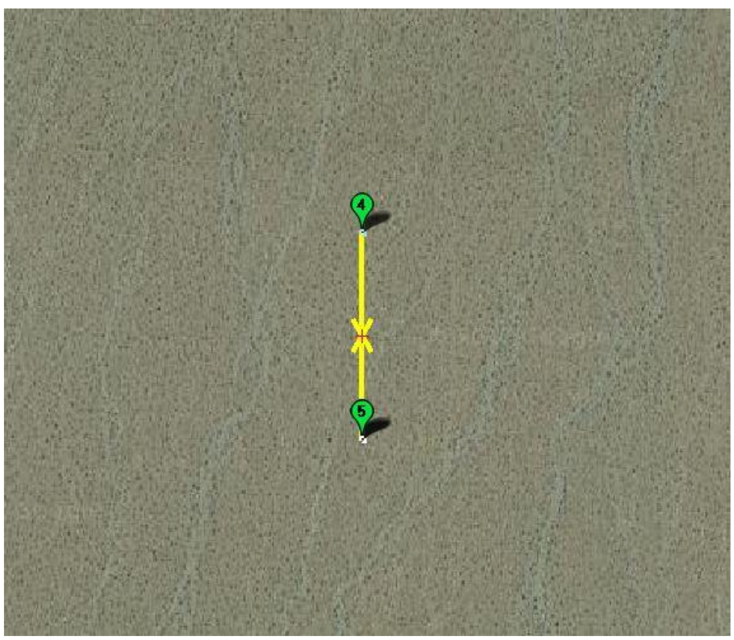

Figure 20: Flight Plan for test case 2.

Figures 21- 27: Graphs from test case 3

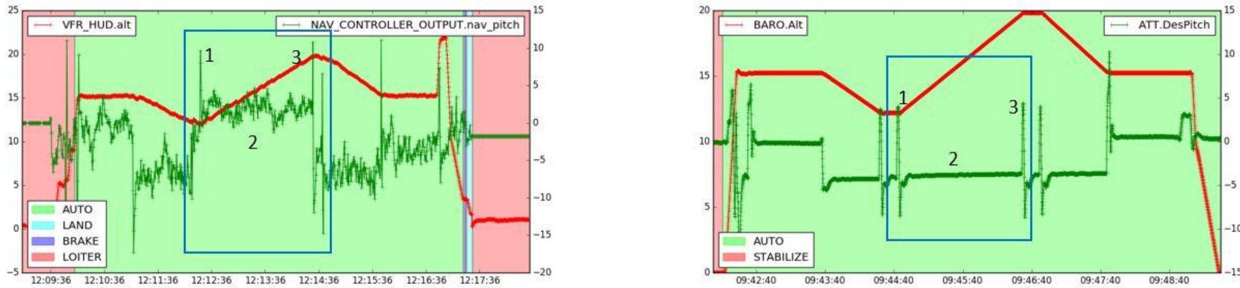

Figure 21: Graphs of Desired Pitch of actual vehicle (left) and Desired Pitch of simulated vehicle (right) in degrees.

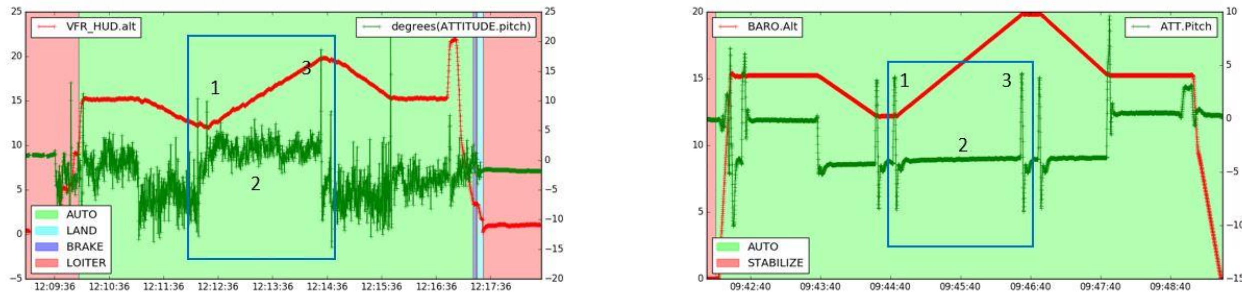

Figure 22: Graphs of Pitch of actual vehicle (left) and Pitch of simulated vehicle (right) in degrees.

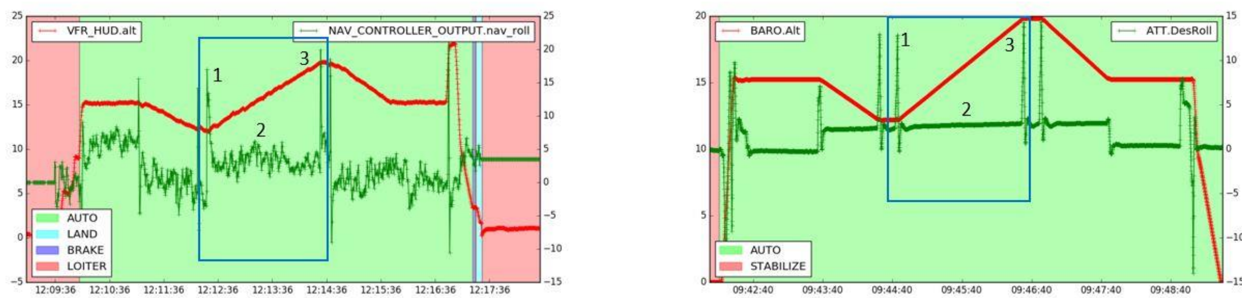

Figure 23: Graphs of Desired Roll of actual vehicle (left) and Desired Roll of simulated vehicle (right) in degrees.

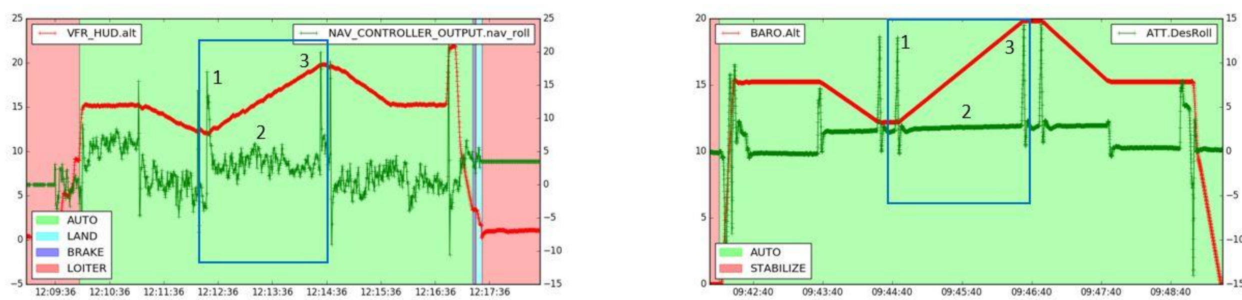

Figure 24: Graphs of Roll of actual vehicle (left) and Roll of simulated vehicle (right) in degrees.

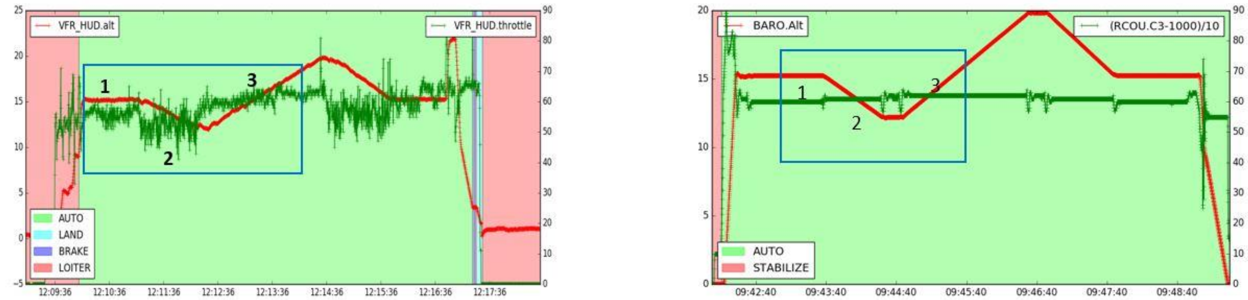

Figure 25: Graphs of Throttle of actual vehicle (left) and Throttle of simulated vehicle (right) in degrees.

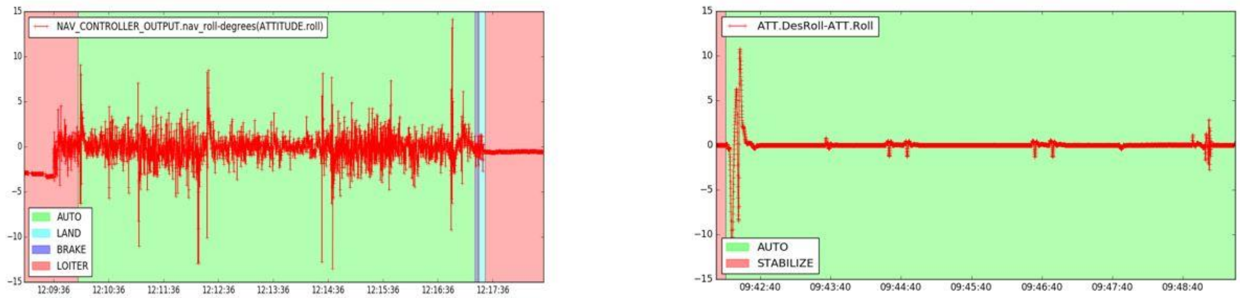

Figure 26: Graphs of Error Roll of actual vehicle (left) and Error Roll of simulated vehicle (right) in degrees.

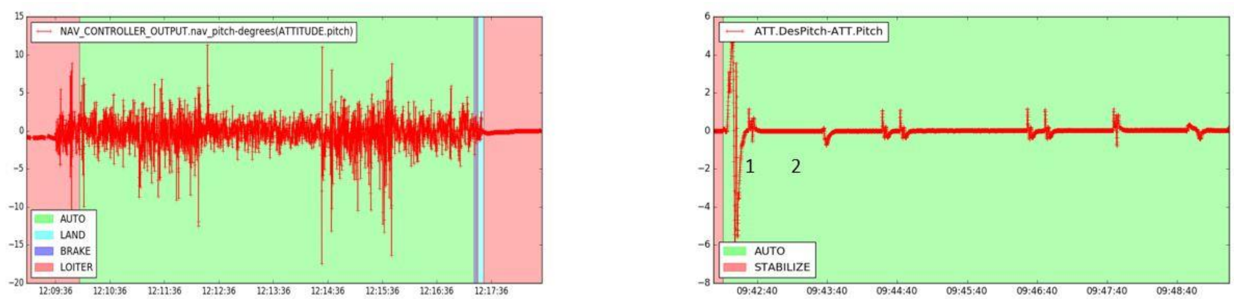

Figure 27: Graphs of Error Pitch of actual vehicle (left) and Error Pitch of simulated vehicle (right) in degrees.

Tables 16 – 22: tables of test case 3 containing values of minimax, mean and points of interest.

Table 16: Values obtained from Desired pitch graphs.

	Actual	Simulated	Diff	%Diff
1. Peak	9.3	3.9	5.4	58.1
2. Plateau	1.9	-3.9	5.8	305.3
3. Peak	10.6	4.1	6.5	61.3
Min	-17	-12	5	29.4
Max	11	10	1	9.1
Mean	-2	-2	0	0.0

Table 17: Values obtained from Pitch graphs.

	Actual	Simulated	Diff	%Diff
1. Peak	9.6	3.6	6	62.5
2. Plateau	2.3	-3.9	6.2	269.6
3. Peak	18.3	4.1	14.2	77.6
Min	-18	-10	8	44.4
Max	21	10	11	52.4
Mean	-2	-2	0	0.0

Table 18: Values obtained from Desired Roll graphs.

	Actual	Simulated	Diff	%Diff
1. Peak	16.7	12.7	4	24.0
2. Plateau	3.1	2.7	0.4	12.9
3. Peak	19.4	14.0	5.4	27.8
Min	-16	-13	3	18.8
Max	23	14	9	39.1
Mean	3	2	1	33.3

Table 19: Values obtained from Roll graphs.

	Actual	Simulated	Diff	%Diff
1. Peak	15.7	12.3	3.4	21.7
2. Plateau	3.4	2.6	0.8	23.5
3. Peak	19	14.3	4.7	24.7
Min	-16	-13	3	18.8
Max	23	14	9	39.1
Mean	3	2	1	33.3

Table 20: Values obtained from Throttle graphs.

	Actual	Simulated	Diff	%Diff
1. Plateau	55.5	59.2	3.7	6.7
2. Plateau	52.2	60	7.8	14.9
3. Plateau	61.8	61.6	0.2	0.3
Min	0	0	0	-
Max	87.0	90.9	3.9	4.5
Mean	49.0	60.1	11.1	22.7

Table 21: Values obtained from Error Roll graphs.

	Actual	Simulated	Diff	%Diff
Min	-13.5	-13	0.5	3.7
Max	14.1	10.7	3.4	24.1
Mean	0	0	0	0.0

Table 22: Values obtained from Error Pitch graphs.

	Actual	Simulated	Diff	%Diff
Min	-17.6	-6.6	11	62.5
Max	11.2	4.7	6.5	58.0
Mean	-0.2	0	0.2	100.0