# Autonomous Vehicle Laboratory for Sense and Avoid Research and Hardware-in-the-Loop Simulations

Duncan Miller[1]
*University of Michigan, Ann Arbor, MI ,48109*

**As autonomous, unmanned aerial vehicles begin to operate regularly in the National Airspace System, the ability to test safely the coordination and control of multiple vehicles will be an important capability. Researchers at NASA Langley have been working to establish an autonomous vehicle testing facility that will allow complex, multi-vehicle tests to be run both indoors and outdoors. Quadrotor helicopters and ground rovers have been used to implement Proportional-Integral-Derivative (PID) control algorithms for collision avoidance and waypoint chasing. Such a testing facility will allow NASA researchers and aerospace contractors to address sense and avoid problems associated with autonomous, multi-vehicle flight control in a safe and flexible manner. This paper documents the results of preliminary system trials, specifically the hardware-in the loop (HIL) simulations.**

## I. Introduction

A closed loop feedback system to monitor autonomous air and ground vehicles gives researchers real-time data to examine and the opportunity to perfect "sense and avoid" control algorithms. An infrared tracking environment, differential GPS antennas, and hardware-in-the-loop simulations assembled at NASA Langley provide such an occasion. A number of similar laboratory research facilities already exist. The Real-time indoor Autonomous Vehicle test ENvironment (RAVEN) at MIT (Ref.1) and the Vehicle Swarm Technology Laboratory at Boeing Corporation[2] are similar testbeds that examine vehicle control and command performance. Such complex multi-vehicle scenarios are of great interest to both commercial airlines and military operations.

## II. Visualization for the Process and Product

The past decade has seen the widespread introduction of free and open source software to the public domain, which allows the freedom to copy and improve upon available source code. Such a peer-to-peer development strategy has advanced the scope of functional electronics, for the benefit of both individual and corporate growth. Over the summer, the Langley lab was used to build upon these readily accessible tools to advance the operations of the Autonomous Vehicle Laboratory.

The Arduino and Processing programming platforms are examples of two principal instruments used to facilitate autonomy. Arduino refers to both a prototyping microcontroller and the C-based software used to program it. Processing's Java-based language gave us the opportunity to resourcefully generate Graphical User Interfaces (GUI) to visually transmit and receive data packets such as the Proportional Integral Derivative (PID) commands.

## III. Hardware-in-the-Loop Simulations

Decision algorithms like PID controllers are often the limiting factor in improving flight dynamics. The hardware-in-the-loop (HIL) simulation eliminates weather and human variability for a safe, repeatable environment. FlightGear was chosen as that environment. FlightGear is a free open source flight simulator development project. It realistically models various aircraft and ground vehicles. FlightGear bridges real world tests with more complex possible scenarios, emulating sensors and state data such as GPS for our custom autopilot to parse and utilize.

### A. ArduPilot and Flight Dynamics Model

The ArduPilot is an Arduino based Inertial Measurement Unit (IMU) autopilot that can be tailored to handle stabilization and control using onboard sensors. I designed a communication platform between FlightGear's purely

[1]Undergraduate Student, University of Michigan Aerospace Engineering, 1320 Beal Ave, Ann Arbor, Mi, 48109, Student Membership.

virtual domain and the Arduino's real world autopilot (Fig.1). FlightGear serves to emulate an aircraft's thermopiles, which use the temperature gradient between earth and sky in two axes to determine pitch and roll. FlightGear outputs the pitch and roll in degrees, serially to Arduino 1, which sends it via a Pulse-Width Modulation (PWM) signal to the ArduPilot (center). Using the control switch on the ArduPilot we can swap between manual stick signals from the RC transmitter and the autopilot's onboard PID controller. The second Arduino receives the final servo surface deflections and transmits them back to FlightGear, thereby closing the loop. We can observe the plane simultaneously navigating in reality next to the virtual flight displayed on screen.

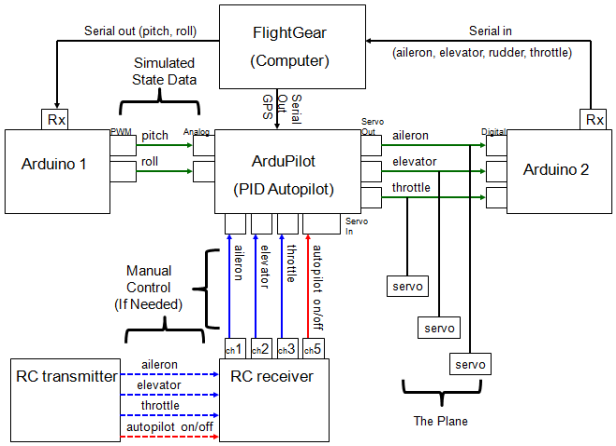


**Figure 1. Block diagram of aircraft HIL simulation.**

A concrete understanding of Proportional Integral Derivative controllers was critical for the eventual hardware in the loop incorporation of the ArduPilot. UAV Playground is another open source Java application that we adapted for better outer loop control of UAVs. Out of the box, stabilization and waypoint navigation already existed. My augmentations included a guided circle, figure eight and concentric circles of increasing radii. Tuning UAV Playground's outer loop controller varies the amount of oscillatory motion seen by the aircraft and was instrumental in guiding my design for the HIL simulation.

### B. Trials with a Ground Vehicle

Subsequently replacing the aircraft with an automobile such as a rover offers an alternative testing agent to model the autopilot controller. A matching block diagram similar to Figure 1 exhibits a similar communication structure. A servo mux was substituted as the control switch and aileron deflections serve to steer the vehicle. By observing the RC rover next to its virtual likeness within FlightGear, we can monitor the HIL simulation in action. Since all of the vehicle's sensors collect state data as if outside, the car believes that it is travelling to waypoints. In reality, it is perched next to the computer, turning its wheels ineffectively alongside a simulated course as shown in Fig 2.

Manual and auto modes are switched on the truck's transmitter, while throttle is shorted straight to the motors, giving the pilot full throttle management during both manual and autopilot control. With the controller perfected indoors in FlightGear's repeatable environment, I then ran the exact same circuit outside and obtained analogous paths on the first trial. This result is encouraging considering we used two distinct mediums to confirm the autopilot's functionality.

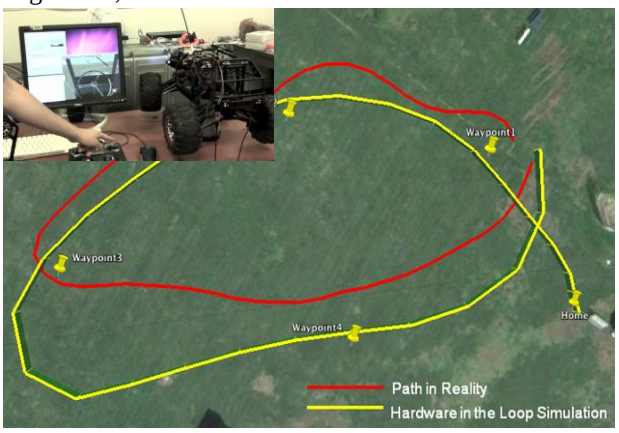


**Figure 2. Simulated path nearly matches reality.**

## IV. Conclusion

An established Autonomous Vehicle Laboratory at NASA Langley provides researchers with an intermediary step between software protocols and full scale reality. Such technology will accelerate the contributions to the emergent field of Unmanned Arial Vehicles.

## Acknowledgments

I would like to thank Garry Qualls for his NASA mentorship and all the students who have built on his research.

## References

[1] How, J.P., Bethke, B., and Frank, A., "Real-time indoor vehicle test environment," *Control Systems Magazine IEEE*, Vol. 28, Issue 2, April 2008, pp. 51-64.

[2]Saad, E., and Vian,J., "Vehicle Swarm Rapid Prototyping Testbed," *AIAA Infotech*, Seattle, Washington, 9 April 2009.