Control and Cybernetics

vol. **45** (2016) No. 3

HIL investigations of extended UAV on-board controller

by

Maciej Słowik, Zdzisław Gosiewski and Daniel Ołdziej

Bialystok University of Technology, Faculty of Mechanical Engineering,
ul. Wiejska 45C, 15-351 Bialystok, Poland
maciejslowik@gmail.com, z.gosiewski@pb.edu.pl,
d.oldziej@pb.edu.pl

Abstract: The paper presents the testing process dedicated to the unmanned aerial vehicle flight controller. The quality of performance of stabilization and navigation tasks has been investigated not during standard in-flight tests, but in the hardware in the loop (HIL) simulation environment. The virtual flight was controlled by the real autopilot. The artificial flight parameters were generated and autopilot uses them as in real flight conditions. Such approach saves time and is economically justified, because of the lack of hardware losses during the failed flight tests.

Keywords: unmanned aerial vehicles, hardware-in-the-loop, flight controller

1. Introduction

There is an overwhelming and rapid development in the domain of unmanned aerial vehicles (UAVs) and their applications. With the abruptly increasing popularity, entirely new areas have been constantly appearing, in which UAVs can be effectively used. At the same time, we can observe continuous miniaturization and application of new possibilities of electronic devices. This is particularly seen in case of sensors and their integration (e.g. MEMS sensors, see Sawicki et al., 2015). Simultaneously, the increase of the computational capabilities of microprocessors allows for the use of more sophisticated control algorithms, which leads to a higher level of automation and autonomy of the unmanned aircraft vehicles (see Masselli, 2013; Meier, 2011; Kang, 2009). Apart from such applications as observation and surveillance, UAVs are used in more and more complex and sophisticated missions. One of the most interesting examples is provided by the so-called sky couriers, which are faster and cheaper for carrying packages, than conventional aircraft or even car transportation. Another interesting option is flying in a swarm of many machines (see Kownacki and Oldziej, 2016; Wang, 2015). New demands for UAV applications are constantly evolving, such as, e.g., gas and pollution detection, wind turbine blade diagnosis, or thermal imaging observation.

So, a change in the construction of UAV systems, tailored for specific demands is strongly desired. The mechanical structure has nowadays a lesser impact on the exploitation properties than electronics and software subsystems. The integration process of electronic boards, sensors, controllers with dedicated software (firmware and drives) is the most time consuming and generates high additional cost. Simultaneously, the flaws and defects of the above mentioned subsystems of UAV are the hardest to find out (see Barros dos Santos, 2011; Ali, 2008). In the context of alleviating the above situation the need arises for the design of automated tests in the form of computer simulations. One of them is software in the loop (SIL) simulation. In this case, the UAV flight controller unit is emulated without its physical (electronics) equipment. The option of hardware in the loop (HIL) is applied when we use the physical driver/flight controller while we emulate the sensor and navigational signals (see Prabowo, 2015; Irawan, 2013). Such approach allows for the comprehensive analysis of the investigated flight controller, with associated electronic subsystem (embedded microcontroller and communication module, which is used for data exchange, see Crespo, 2014).

Two examples of hardware in the loop (HIL) investigations of a selected flight controller are shown in the present paper. The Pixhawk autopilot, to which the present analysis is devoted, is commonly used in different UAV applications. It is an open source and open hardware controller and it can be used in fixed-wing planes, different multirotor configurations, and ground rovers.

2. The investigated flight controller

The investigated object is on-board flight controller Pixhawk (its main features being collected in Table 1). Due to the number of input and output sources and rich functionality (based on ArduPilot), the selected autopilot is widely used for control of various UAVs (see Florek, 2014; Grenestedt, 2015). It allows for using the different flight modes, from manual, through stabilized, to guided – where waypoints are considered as pairs of geographical latitude and longitude coordinates, and the altitude, on which they are located. The most sophisticated modes of operation are auto (or mission – where UAV flies between the earlier defined waypoints) and off board flight mode (online ground control or mode generated on additional microcomputer). In the last case the external microcomputer board with dedicated microprocessor (e.g. BeagleBoard Black) and more sophisticated flight mission control algorithms can be used. To such missions belong the sophisticated flight control, based on precision global navigational satellite systems (GNSS) receivers, or obstacle avoidance with vision systems. Pixhawk firmware source code and electrical scheme are available on Github Firmware (2016) with open source and open hardware licenses on Github Hardware (2016). Pixhawk system flexibility is used for rapid development of applications by the addition of new functionalities and structures, introduced by the active programmers community.

For our purposes, we add a Neo-M8N GPS with Compass GNSS receiver

Table 1. Selected parameters of Pixhawk flight controller (see Pixhawk, 2016)

Processor	32bit SMT32F427 Cortex M4, 168 MHz, 256 KB RAM, 2 MB Flash
Sensors	16 bit gyroscope, 14 bit accelerometer/magnetometer, MPU 6000 3-axis accelerometer/gyroscope, MS5611 barometer
Interfaces	5x UART, 2x CAN, PPM sum signal input, I2C, SPI, 3.3 and 6.6 V ADC inputs, 16 PWM outputs, microSD card slot
Dimensions and weight	Width: 50 mm, Thickness 15.5 mm, Length 81.5 mm, Weight: 0.038 kg

which allows for receiving both GPS and GLONASS signals at frequency 10 [Hz]. Pixhawk flight controller and the above mentioned GPS receiver are shown in Fig. 1.

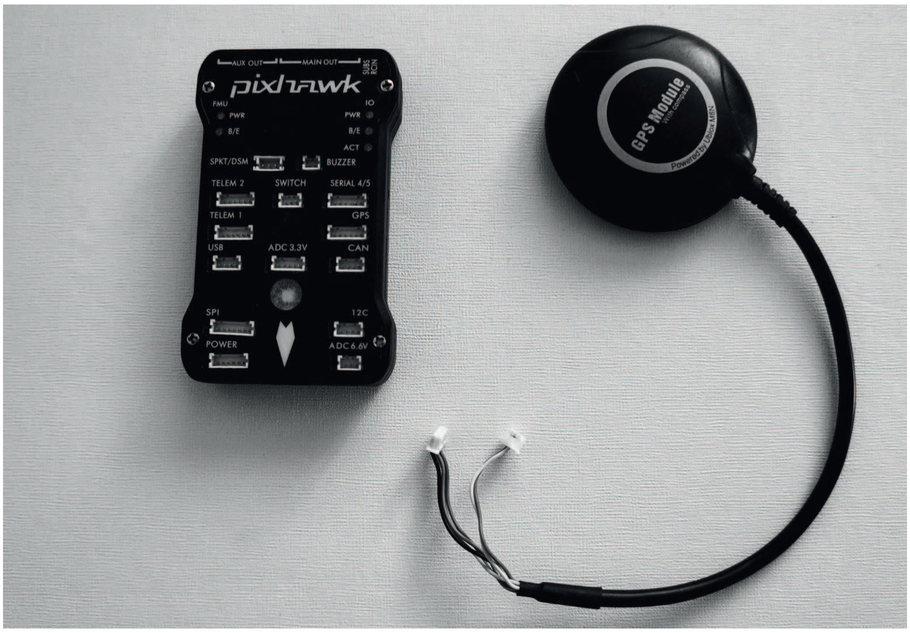


Figure 1. The Pixhawk v2 flight controller with Neo-M8N GPS with Compass GNSS receiver

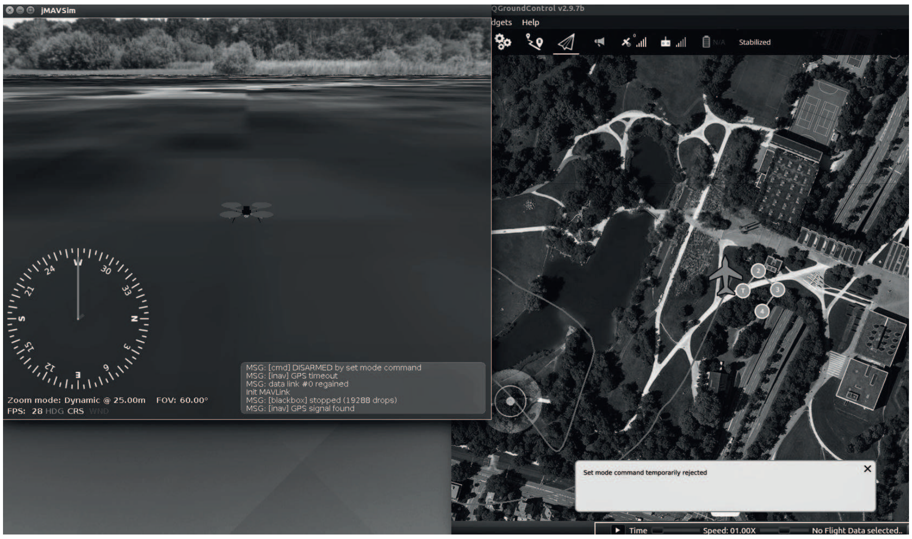
Additionally, Pixhawk is able to cooperate with many different ground control stations (including MissionPlanner, QGroundControl and APMPlanner2) and communicate by telemetry modems connected to UART and USB ports and with the help of TCP and/or UDP protocols. In the study, reported in the present paper, the authors have used Pixhawk flight controller and QGroundControl station for HIL simulations of quadrocopter control.

3. HIL experiment – manual RC flight mode

In order to carry out the HIL investigation on dedicated firmware, called Generic Quadrocopter X model was uploaded into Pixhawk flight controller, which allowed for emulating the signals from the inertial measurement units and signals provided as the navigation data. Due to the above arrangement, the flight controller may realize a simulated flight in the virtual environment, based on the data indicated by GPS location. These signals are generated and formed into specially crafted messages, included in the Mavlink protocol (see Mavlink, 2016). This makes it possible to set and read the model parameters and flight data where additional part of message can be set or waypoints generated. For security reasons, every message is terminated with the CRC calculated code, which is checked for validation of every data frame. Mavlink is the backbone of communication between ground control stations, additional inboard computer and autopilot. Mavlink was used as the basis for communication between subsystems in HIL simulation. The message code is available as a collection of C programming language header files or in the form of XML definition files. Due to open source license there are many versions of Mavlink implementations in programming languages, such as Java and Python. In the first experiment, the Mavlink protocol has been used for communication between jMAVSim simulation environment, Pixhawk board, and QGroundControl station software. As a result, this allows for the control of the emulated quadrocopter with RC radio controller (Futaba T10C) in manual and stabilized flight modes. A direct view from pilot perspective is provided in the jMAVSim windows (see Fig. 2, left part). The flight path record, obtained in QGroundControl station, is shown in the right part of Fig. 2.

4. The HIL experiment – the guided mode

In the second, more complex HIL experiment, the extended guided mode was tested. In the classic version of the above mentioned mode, quadrocopter flight is based on the setting of waypoints. Here, it is realized by list of commands, e.g. *goto(10,0)* - means 10 meters north, *change_yaw*(90) - means change of yaw angle by 90 degree rotation. In the tests, the Mavlink protocol, with addition of pyMavlink wrapper with Dronekit framework were used (see Dronekit, 2016). The framework is a collection of libraries and submodules, this collection allowing for more efficient creation of flight missions, basing on Mavlink protocol and MavProxy software. The software design is given in the text mode and it enriches the functions of the ground control station by joining different modules and plugins. It allows for changing the flight modes, viewing of UAV's position on the map, testing of the safety checks and failsafe or overwrite RC PWM values. This option of Pixhawk in HIL mode of operation can be connected



```
Zoom mode: Dynamic @ 25.00m    FOV: 60.00°
FPS:  28 HDG CRS WND
```

```
MSG: [cmd] DISARMED by set mode command
MSG: [inav] GPS timeout
MSG: data link #0 regained
Init MAVLink
MSG: [blackbox] stopped (19288 drops)
MSG: [inav] GPS signal found
```

Set mode command temporarily rejected

Figure 2. On the left: jMAVSim Quadrocopter simulator, on the right: QGroundControl ground control software with flight path

with mission script using Dronekit library by Mavlink protocol. The above mentioned software elements were connected in the form presented in Fig. 3.

The solution, outlined above, allows for using the high abstraction layer algorithms with Dronekit API (Application Programing Interface), and then for its transition to the middle layer. Subsequently, messages are sent by the low level layer using the serial (USB) protocol to the flight controller. Each of the above submodules has the ability to communicate in both directions (sending/receiving). Such approach increases the security level for real flight operations. Every command is ended with a proper message – having the ACK header (acknowledge message). Depending on the configuration, in case of error occurrence, a mission can be stopped and the error handling procedure is executed, e.g., during execution of the guided mode, the quadrocopter can change the mode for position hold or RTL (RTL – Return To Lunch, what means return to the start point). This solution increases safety and allows for handling unexpected situations. A Dronekit script flight mission, in which the quadrotor is flying along a square path, is presented in Fig. 4. Flight path is assembled by four lines in the form of square with 40 meter side. The flight was conducted in the modified guided mode at 5 meters of altitude.

Goto Dronekit function, which was used during the experiment, works according to the following principle. First, a current global reference location parameter value is fixed. Then, movement of the UAV is continued until UAV achieves the desired incremental position (according to axis of geographical frame which follows UAV). It allows to set flight paths in different way, not

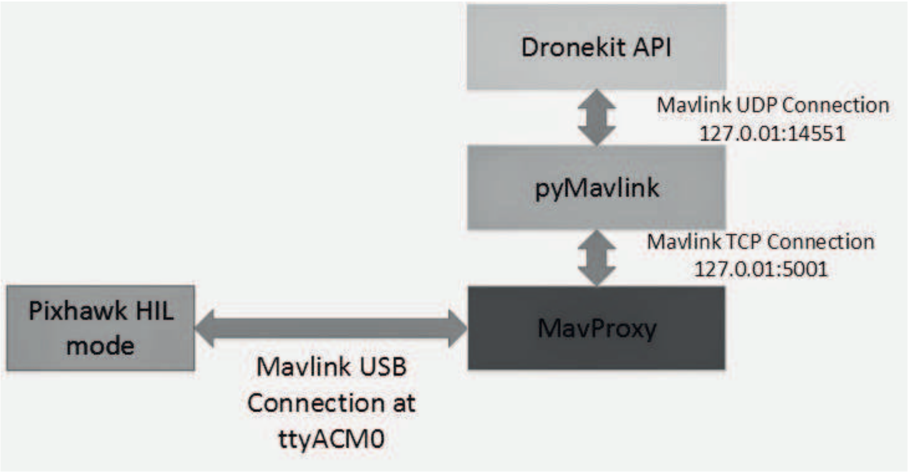


Figure 3. Communication scheme of HIL mode

as set of waypoints (in fixed geographic frame), but rather as points generated by the user function. The guided mode was used in NED (North, East, Down) local reference coordinates system. In this case D, the altitude value, was constant. This allows for generation of new waypoints as function of control algorithms results.

The set of Dronekit script for flight commands in our experiment was:
goto(40, 0) – first line – south–north direction
goto(0, 40) – second line – west–east direction
goto(-40, 0) – third line – north–south direction
goto(0, -40) – fourth line – east–west direction

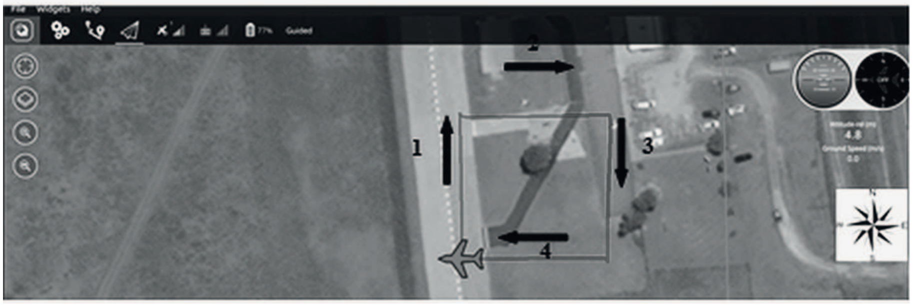


Figure 4. Visualization of Dronekit *goto* commands – square flight path with subsequent (1-4) steps of the mission

The same flight record is shown in the form of emulated navigational signals (in the form of on-board GNSS receiver standard) – geographical latitude and longitude – as it is presented in Fig.5.

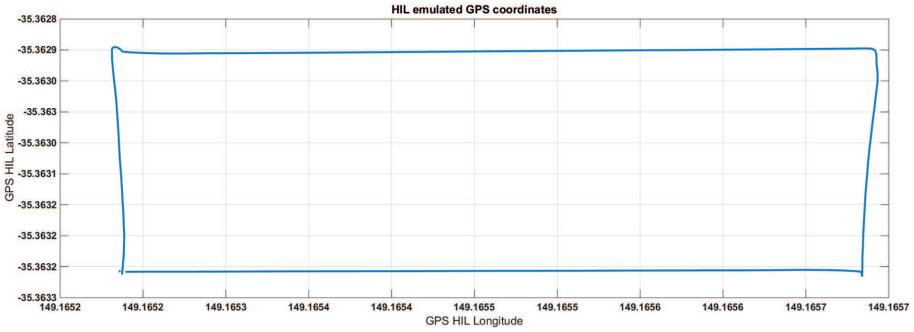


Figure 5. Plot of emulated geographical latitude and longitude

Additionally, the possibility for control of quadrocopter rotation – which is especially desired when surveillance head does not have yaw control ability – can be realized by *commanded_yaw(yaw_angle)* command. Changes of yaw angle during the second HIL experiment are presented in Fig. 6.

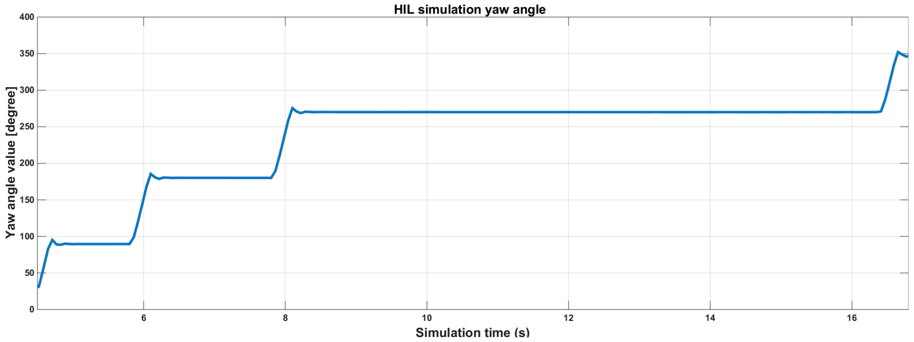


Figure 6. Yaw angle changes in HIL simulation versus time. The longest change of the yaw angle is caused by the landing procedure start at the time of reception of last command

5. Conclusions

The complexity level of UAV on-board equipment constantly rises along with the capacity to realize new applications. The dimensions and the weight of respective electronic systems (e.g. MEMS sensors) decreases, while computing abilities of microprocessors increase, so that one can realize increasingly complicated inboard control and diagnostics functions. Consequently, the need arises, in particular, to extend the computer simulation tests of inboard equipment.

This kind of approach was used to investigate the Pixhawk flight controller and the results of testing are presented in the paper. Here, two hardware-in-the-loop simulations are described. The first one was focused on testing of manual and stabilized flight mode in the 3D simulation environment, where control was carried out by the operator through radio controller, connected by the RC receiver to the real flight controller. In the second case, automated guided flight mode was investigated. The guided mode was used in local reference system (NED). Both experiments were successful. Stabilized flight mode using radio controller and guided flight mode were both properly interpreted by flight controller and ground control station software. Based on these results we foresee that the future investigations will be focused on development of new flight modes. The landing in the net in high wind conditions or drive subsystem failure detection and emergency landing can serve as examples.

Acknowledgement

The work has been accomplished under the research project No. S/WM/1/2016 and was financed by the Polish Ministry of Science and Higher Education

References

Ali, A. (2008) Validation of an autopilot through hardware in the loop simulation. In: *Proceedings of the IEEE International Multitopic Conference*, 2008, INMIC 2008, December 23-24, Karachi, 571-574.

Barros dos Santos, S. R. (2011) Longitudinal autopilot controllers test platform hardware in the loop. In: *Proceedings of the IEEE International Systems Conference, SysCon*, April 4-7, Montreal, 379-386.

Crespo, G. (2014) Setup of a communication and control systems of a quadrotor type Unmanned Aerial Vehicle. In: *Proceedings of Design of Circuits and Integrated Circuits, DCIS Conference*, November 26-28, Madrid, 1-6.

Dronekit: http://dronekit.io, access date: June 2, 2016.

Florek, M. (2014) Comparing approaches to quadrocopter control. In: *Proceedings of the 23rd International Conference on Robotics in Alpe-Adria-Danube Region, RAAD*, September 3-5, Smolenice, 1-6.

Grenestedt, J. (2015) A high performance USV with applications to surveillance and monitoring. In: *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR*, October 18-20, West Lafayette, 1-6.

Github Firmware, https://github.com/PX4/Firmware, access date: June 2, 2016.

Github Hardware, https://github.com/PX4/Hardware, access date: June 2, 2016.

Irawan, A. (2013) Three-loop autopilot for attitude control system on hardware in loop simulation. In: *Proceedings of the 3rd International Conference on Instrumentation, Communications, Information Technology, and*

Biomedical Engineering ICICI-BME, November 7-8, Bandung, 286-291.

Kang, Y. (2009) Linear tracking for a fixed-wing UAV using nonlinear model predictive control. *IEEE Transactions on Control Systems Technology* **17**(5), 1063-6536.

Kownacki, C. and Oldziej, D. (2016) Fixed-wing UAVs flock control through cohesion and repulsion behaviours combined with a leadership. *International Journal of Advanced Robotic Systems* **13**, 1729-8806.

Masselli, A. (2013) A cross-platform comparison of visual marker based approaches for autonomous flight of quadrocopters. In: *Proceedings of the International Conference on Unmanned Aircraft Systems ICUAS*, May 28-31, Atlanta, 685-693.

Mavlink: http://qgroundcontrol.org/mavlink/start, access date: June 2, 2016.

Meier, L. (2011) Pixhawk: a system for autonomous flight using onboard computer vision. In: *Proceedings of the IEEE International Conference on Robotics and Automation ICRA*, May 9-13, Shanghai, 2992-2997.

Pixhawk: https://pixhawk.org/modules/pixhawk, access date: June 2, 2016.

Prabowo, Y. A. (2015) Hardware in-the-loop simulation for visual servoing of fixed wing UAV. In: *Proceedings of the IEEE International Conference on Electrical Engineering and Informatics ICEEI*, August 10-11, Bali, 247-252.

Sawicki, A., Walendziuk, W. and Idzkowski, A. (2015) Estimation of the object orientation and location with the use of MEMS sensors. *Proceedings of SPIE* 9662, 1-8.

Wang, X. (2015) UAV formation: from numerical simulation to actual flight. In: *Proceedings of the IEEE International Conference on Information and Automation*, August 8-10, Lijiang, 475-480.