Transport and Aerospace Engineering

ISSN 2255-9876 (online)
ISSN 2255-968X (print)
2018, vol. 6, pp. 37–43
doi: 10.2478/tae-2018-0005
https://content.sciendo.com

Development of a Flexible Software Solution for Controlling Unmanned Air Vehicles via the Internet

Deniss Brodņevs

Institute of Aeronautics, Faculty of Mechanical Engineering, Transport and Aeronautics, Riga Technical University, Riga, Latvia

***Abstract* – Remotely piloted operations of lightweight Unmanned Air Vehicles (UAV) are limited by transmitter power consumption and are always restricted to Line-of-Sight (LOS) distance. The use of mobile cellular network data transfer services (e.g. 3G HSPA and LTE) as well as long-range terrestrial links (e.g. LoraWAN) makes it possible to significantly extend the operation range of the remotely piloted UAV. This paper describes the development of a long-range communication solution for the UAV telemetry system. The proposed solution is based on (but not restricted to) cellular data transfer service and is implemented on Raspberry Pi under Gentoo Linux control. The goal of the project is to develop a flexible system for implementing optimized redundant network solutions for the Non-LOS remote control of the UAV.**

***Keywords* – 3G, 4G, Cellular network, Internet, LTE, Remote control, RPAS, Telemetry, Unmanned Air Vehicles (UAV).**

I. Introduction

Unmanned Air Vehicles (UAV) and Remote-Control Vehicles (RCV) require wireless communication solution to perform Remote Controlled (RC) operations. The use of airborne radio receiver in conjunction with Electronic Speed Controllers (ESCs) allows to operate UAV via a remote-control transmitter. The approach “ground transmitter→airborne receiver→ESC” is the most lightweight and cheap solution. However, such a solution does not provide fail-safe operation of the UAV (the pilot is responsible for compensating all flight conditions, e.g. wind gusts), does not provide any feedback from the UAV (e.g. remaining battery power, artificial horizon, etc.) and does not allow autonomous flight (by waypoints instead of manual remote piloting). To overcome the above-mentioned disadvantages, modern UAVs are typically equipped with flight controllers (also called “telemetry systems”).

The flight controller is responsible for controlling ECSs according to commands received from the ground transmitter. One of the most common functions of the flight controller is automatic stabilization of the UAV with respect to wind gusts or of commands from the ground pilot that have been made too fast. This enables the possibility of building more efficient flying vehicles by the cost of stability (copters are unstable by default). So, the primary function of the flight controller is UAV flight stabilization. The stabilization function requires information about roll, pitch and yaw. Such information is usually supplied from the built-in Accel/Gyro solid state unit (typically called Micro-Electro-Mechanical System, or MEMS). This means that information about roll, pitch and yaw is available in the flight controller and can be supplied to the ground pilot’s Ground Control Station (GCS) as well. This requires an additional UAV-ground radio link called “telemetry link”. Usually remaining battery capacity (from a dedicated battery sensor), barometric altitude (from the barometric pressure sensor), GPS coordinates and compass information are also supplied in addition to the artificial horizon information via the telemetry link. Finally, the GPS, barometric altitude and magnetic compass can be used to ensure autonomous flights by using waypoints. This means that typical mini-UAVs (takeoff weight is less than 30 kg) have two radio channels: one for remote-piloted operations and another one for telemetry. The telemetry channel should be bidirectional to be able not only to receive reports from, but also to specify new waypoints to the UAV. Such an approach is shown in Fig. 1.

Typically, the RC radio channel is at a frequency of 2.4 GHz, whereas telemetry utilizes 433 MHz (900 MHz can be used in the USA). The telemetry radio channel carrier frequency is lower compared to the RC radio channel frequency because lower frequencies provide slightly greater communication range, whereas the RC channel is used for hundreds of meters only when the UAV is clearly visible. All the above-mentioned frequencies propagate in Line-of-Sight (LOS) manner only. The LOS distance of the UAV flying at an altitude of 150 m is 50 km. However, its actual communication range is limited by transmitter radiated power and is less: not more than 15 km (typically even less) [1]. This means that flight by waypoints becomes fully uncontrolled and blind (without any feedback to the GCS) when the communication range maximum of the telemetry radio channel is exceeded.

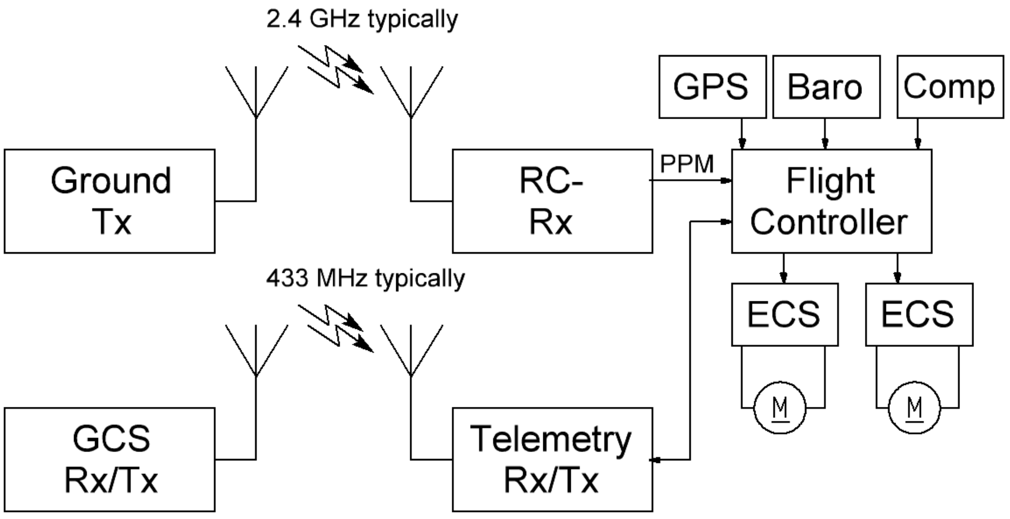

Fig.1. Typical mini-UAV wireless links

Various wireless communication solutions can be used to extend the range of controllable flight: terrestrial wireless links with wide coverage (e.g. LoraWAN or mobile cellular networks like 3G, LTE [2]) of satellite communications (SATCOM). SATCOM is more suitable for high takeoff mass UAVs intended for military use, because SATCOM equipment is expensive, and its mass, dimensions and power consumption are noticeably higher compared to terrestrial wireless equipment solutions.

The use of cellular data transfer services (e.g. 2G, 3G or LTE) to transfer telemetry signals can significantly increase the range of UAV RC operations. Available field studies state that cellular networks have sufficient signal strength – up to 300 m or even better [3]–[5], which is completely enough for the UAV flight that is restricted to 120 m by law.

The idea of using cellular data transfer services in telemetry channel implementation is not new. For example, [6] describes a solution that uses open source mavproxy software [7] to transmit Pixhawk flight controller telemetry data through the Internet via cellular data transfer service to the GCS.

II. Motivation

A. State of the Art

A typical UAV has an RC-receiver connected to the flight controller like 3DR Pixhawk, which is connected to ECSs. Both the RC-receiver and flight controller have their own radio equipment operating at different frequencies. Radio equipment is widely available for hobbyists (e.g. 3DR, XBee, etc) and can be used to develop systems with a controllable operating range of up to 10 km

[1]. The telemetry signal is then transmitted to the GCS and can be displayed for the pilot via open source software like Mission Planner, Q Ground Control Station, APM Planner, etc. The video channel for the First-Person View (FPV) is usually transmitted via a dedicated wireless channel, for example [8].

The use of cellular networks for UAV control is known as well. Minla HDW solution uses its own hardware to control UAVs from the Internet [9]. Minla HDW automatically establish connection with minla server. The pilot should connect to the Internet to be able to control his UAV via the web interface. A similar project with its own hardware is called Virt2Real [10]. Navio2 [11] provides a hardware extension for Raspberry PI to get the same functionality as described above. All projects allow to control and get feedback from the UAV and do not provide customization of software.

The solution described in [6] uses Raspberry PI embedded system to run open source mavproxy software in order to stream 3DR telemetry information into the Internet via 3G/LTE. Note that there is also a free available project providing a ready to deploy image for the embedded computers, which runs mavproxy software by default and streams 3DR Pixhawk flight controller telemetry information [12].

B. "To-Do" List

All of the above-mentioned solutions utilize cellular data transfer services by using 3G/LTE dongles like Huawei 3372. However, available field studies [4] as well as our preliminary testing [13] show a dramatic decrease in performance and service quality of 3G and LTE cellular systems in the skies. To combat the problem, a redundant communication solution [14] or even a more flexible yet unpublished multipath solution [15] can be used. These solutions are able to combine cellular 3G HSPA+ and LTE, as well terrestrial long range LoreWAN like systems. Such solutions require a flexible, open source platform for development.

Unfortunately, all the above-mentioned solutions do not provide the required flexibility to implement the above-mentioned techniques. The proposed project must be based on embedded hardware under flexible operating system control. The development steps are listed below:

- Hardware and operating system selection: Raspberry PI 3B under Gentoo Linux control;
- UAV hardware selection: Pixhawk Pixracer flight controller;
- Telemetry control channel link to the WAN: telemetry control and monitoring can be done by using mavproxy. Only cellular connections like 3G/LTE have been implemented by now;
- Development of the flexible redundant network solution: is still being debugged, optimized and tested;
- Video signal streaming with balancing: is on "to-do" list and has not been implemented yet;
- Development of a board for backup RC control: not started yet.

Raspberry PI 3B was selected as embedded hardware because its cost is reasonably low, it uses 5V power supply and has well-developed hardware. Starting from 64-bit Linux kernel version 4.10, Raspberry PI 3B support was introduced. The Gentoo Linux was selected as one of the most flexible and still user-friendly environments. It allows to easily build its kernels with specific kernel implementations; the use of "use flags" helps to optimize the system and resolve software conflicts as well; small yet well-educated community allows to solve complicated problems more quickly.

The Pixhawk Pixracer (XRacer board family) was chosen as a smaller modern flight controller that can be controlled via a USB port in real time. This enables fast and flexible links and omits the necessity to use UART or Wi-Fi interfaces (the use of 2.4 GHz built-in Wi-Fi interface should be avoided at all during the flight because it can cause interference with the RC-channel).

The mavproxy software is used in the project to "build a bridge" between the WAN (Internet) and flight controller. The GCS can run all the above-mentioned (already developed) software like Mission Planner, so the development of GCS software is not a goal of this project.

III. IMPLEMENTATION

Raspberry PI 3 model B (further called rpi3) is a 64-bit board. Thanks to the Gentoo community, 64-bit Gentoo Linux is now also available for the rpi3 as an image [16]. This image provides a ready to use desktop system with all necessary applications like Internet Browser, Office, Video Player, etc. However, this image provides Gentoo kernel without sources. Thanks to the flexible Gentoo Linux structure, a custom kernel can be manually built from available sources (sources can be easily downloaded from github).

This project implementation uses a prepared image [16] with custom 64-bit kernel version 4.14.44 with the flexible multipath implementation of the network [15]; however, this implementation is not used because it has not been fully tested yet. Unnecessary functions like graphics (xserver) are disabled.

In order to get access to the WAN from the rpi3, a Huawei 3372h dongle is used. Huawei E3372h is a Cat 24 device able to operate in 3G dual cell DC-HSPA+ mode; and LTE Cat 4 device (3G LTE) which, in addition, supports 2 CA in downlink. Its default operational mode is HiLink (CdcEthernet). In this mode, the device operates as a NAT server and emulates a virtual network card (NDIS) on the local computer; all its configuration is done by accessing its internal web server page (default IP is 192.168.8.1). Please note that initially the dongle will not initialize network functions until its software is installed. The Huawei software installation can be omitted because Linux kernel supports such types of dongles if USB driver for GSM and CDMA modems is enabled in the kernel. Now the dongle network functions can be switched on by running free usb-mode switch software.

The Pixracer v1.0 flight computer communicates with the rpi3 via its USB port. The use of USB connection allows to easily build a connection without the necessity to use UART or Wi-Fi interfaces (please note that the use of 2.4 GHz Wi-Fi interface can cause interference with the RC-channel and should be disabled).

The rpi3 runs software called mavproxy written in python language. The mavproxy software receives telemetry data transmitted from the flight computer via USB (usbTTY0 port) and redirects it to the desired IP address. It is possible to use local or WAN IP addresses. Since this project operates via the Internet, the WAN real IP is used.

There are two types of Internet Protocol (IP) traffic: TCP and UDP. The TCP protocol is typically used when data should be delivered without corrupted packets. The TCP retransmits all lost/corrupted packets, which leads to increased jitter. Assuming that some number of retransmissions will take place, the receiving side must provide relatively high buffer size to be able to reassemble messages from successfully delivered and retransmitted packets. This means that the receiving side must be tolerant to jitter and delays. The use of UDP network protocol helps to avoid the retransmission of undelivered packets, hence such a protocol is more suitable for delay-sensitive data transmission, but by the cost of corrupted data loss. Typically, the UDP protocol is preferred in telemetry data transmission, because artificial horizon, altitude, etc. data should be transferred with minimum possible delays. In this setup UDP is used. However, it is planned to use TCP when Multipath solution will be fully implemented, as it can combat the jitter problem and ensures secure delivery of data [15].

Gentoo Linux have no mavproxy in its package repository. To be able to run mavproxy, some software should be installed first. Dependencies requirements and installed software are listed below:

- Python programming language, version 2.7 (2.7.14-r2 was installed);
- Collection of algorithms “Opencv”, latest version with python support enabled (3.3.0-r5);
- Cross-platform C++ GUI toolkit “wxGTK”, version 2.x (2.8.12.1-r2 was installed),
- Interactive library for plotting and doing basic data analysis in python “matplotlib” (2.2.2-r1);
- Fast array and numerical python library “numpy”, latest version (1.13.3);
- Python imaging library “pillow”, latest available (4.3.0);
- Python serial port extension “pyserial”, version 3.x (3.4).

Then “pymavlink” can be installed. It will operate correctly only if it has been assembled by python version 2.7. Please note that there are no prepared ebuilds for Gentoo Linux. You must create it, or python package installer “pip” can be used (however, the use of pip is not recommended by the Gentoo community). Finally, “mavproxy” can be installed. Again, there are no prepared ebuilds and the software should be installed by python 2.7 over the custom ebuild or via the pip installer.

IV. Experimental Results

The proposed communication solution must be lightweight and reliable. The traditional telemetry wireless link was also tested in the following benchmark to be able to compare both solutions.

The first experiment was made to determine the reliability of existing and proposed communication channel solutions. During this experiment, the UAV was immovable. The duration of each benchmark was 5 minutes, and the distance between the GCS and UAV was 20 m. The experimental results were reported by Mission Planner software (v.1.3.56) and are shown in Table 1.

TABLE I

Performance of the Telemetry Link

Setup	Options	Experimental Results	Subjective Opinion
Native 433 MHz Sik Radio	Radio interface speed: 64 kbps ECC: enabled Max. send window: 131 ms	Speed: 11.3 kbps Packets Lost: 0.11 % Max. time between packets: 383 ms	Artificial horizon display changes unevenly
Native 433 MHz Sik Radio	Radio interface speed: 64 kbps ECC: disabled Max. send window: 131 ms	Speed: 13.9 kbps Packets Lost: 0 % Max. time between packets: 451 ms	No noticeable difference with previous result
Native 433 MHz Sik Radio	Radio interface speed: 64 kbps ECC: enabled Max. send window: 33 ms	Speed: 14.0 kbps Packets Lost: 0 % Max. time between packets: 242 ms	No noticeable difference with previous result
Native 433 MHz Sik Radio	Radio interface speed: 64 kbps ECC: disabled Max. send window: 33 ms	Speed: 10.5 kbps Packets Lost: 0.015 % Max. time between packets: 269 ms	No noticeable difference with previous result
Over IP network	100Mbps Ethernet + 2.4 GHz Wi-Fi between rpi3 and portable computer	Speed: 22.1 kbps Packets Lost: 0.041 % Max. time between packets: 250 ms	Screen update jitter becomes stable (steps are still visible)
Over IP network	100Mbps Ethernet between rpi3 and portable computer	Speed: 21.9 kbps Packets Lost: 0 % Max. time between packets: 251 ms	No noticeable difference with previous result
Over cellular data transfer service	Mobile cellular 3G LTE service (lte cat. 4)	Speed: 21.2 kbps Packets Lost: improper reporting, similar to [17] Max.time between packets: 258 ms	No noticeable difference with previous result

The second experiment was made to determine additional weight (about 150 grams) and power consumption impact (note that Huawei 3372h dongle power consumption is more (up to 300 mA) in LTE compared to 3G, whereas rpi3 kernels operate with reduced 600 MHz frequency because they are not heavily loaded). During the experiment, a quadcopter with 500 mm frame was flared 10 minutes about 1 m over the ground level at a +26 °C temperature without wind at all. It was consuming 2.93 Ah with standard 433 MHz Sik radio and 3.51 Ah when equipped with rpi3 and Huawei dongle operating in LTE mode. The consumption was measured during the battery charging procedure after each flight.

V. Results

This paper introduces a new flexible open source solution that can be used to build a long-range telemetry link via (but not limited to) cellular data transfer services. The main goal is to achieve a flexible system which allows: to easily build its kernels with specific kernel implementations; to use custom ebuilds for specific software installations; to use ”use flags” to optimize the system and resolve software conflicts as well.

The use of cellular data transfer service (LTE Cat. 4) allows to get a smoother display of the artificial horizon (screen update jitter becomes stable) with almost the same or even less delay compared to the native Sik Radio microwave link. The use of Wi-Fi access of the GCS can introduce packet loss, but has no noticeable impact onto delays. Such a solution provides noticeably greater data rates, which allow to transmit additional data streams like FPV. The RC range becomes limited by cellular coverage only (compared to LOS in case of the Sik Radio microwave link). However, additional weight of the cellular dongle and embedded computer rpi3, as well their power consumption, reduce remaining battery time (in this experiment the remaining battery time was reduced by 15 % approximately).

As a future work, flexible redundant communication solution will be implemented first (refer to State of the Art chapter for the motivation and more details). In parallel, remote video streaming will be implemented too. Beside this, we hope that the use of such implementations will help us investigate the sources of and to develop solutions to overcome cellular network interference problems, which are very common at high altitudes or in open spaces (refer to Introduction for more details).

Acknowledgements

Would like to thank Olafs Slabkovskis for bit, but still useful comments supporting this project’s software part.

References

[1] F. Dimc and T. Magister, “Mini UAV communication link systems,” *Promet Glob. Zb. Ref. Conf. Proc.*, p. 9, 2006.
[2] D. Carrillo and J. Seki, “Rural area deployment of internet of things connectivity: LTE and LoRaWAN case study,” *Proc. 2017 IEEE 24th Int. Congr. Electron. Electr. Eng. Comput. INTERCON 2017*, 2017. https://doi.org/10.1109/INTERCON.2017.8079711
[3] N. Goddemeier, K. Daniel, and C. Wietfeld, “Coverage evaluation of wireless networks for unmanned aerial systems,” in *2010 IEEE Globecom Workshops, GC’10*, 2010, pp. 1760–1765.
[4] B. Van Der Bergh, A. Chiumento, and S. Pollin, “LTE in the sky: Trading off propagation benefits with interference costs for aerial nodes,” *IEEE Commun. Mag.*, 2016. https://doi.org/10.1109/MCOM.2016.7470934
[5] L. Afonso, N. Souto, P. Sebastião, M. Ribeiro, T. Tavares, and R. Marinheiro, “Cellular for the skies: Exploiting mobile network infrastructure for low altitude air-to-ground communications,” *IEEE Aerosp. Electron. Syst. Mag.*, vol. 31, no. 8, pp. 4–11, 2016. https://doi.org/10.1109/MAES.2016.150170
[6] N. Santos, A. Raimundo, D. Peres, P. Sebastião, and N. Souto, “Development of a software platform to control squads of unmanned vehicles in real-time,” in *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, 2017, pp. 1–5. https://doi.org/10.1109/ICUAS.2017.7991528
[7] H. Choi, M. Geeves, B. Alsalam, and F. Gonzalez, “Open source computer-vision based guidance system for UAVs on-board decision making,” in *IEEE Aerospace Conference Proceedings*, 2016. https://doi.org/10.1109/AERO.2016.7500600
[8] G. Crespo, G. Glez-de-Rivera, J. Garrido, and R. Ponticelli, “Setup of a communication and control systems of a quadrotor type Unmanned Aerial Vehicle,” *Proc. 2014 29th Conf. Des. Circuits Integr. Syst. DCIS 2014*, pp. 0–5, 2014. https://doi.org/10.1109/DCIS.2014.7035590

[9] Minla-RC, "Minla HDW." [Online]. Available: http://minlarc.com/ [Accessed: May 20, 2018].

[10] Virt2real-ltd, "Virt2Real." [Online]. Available: http://virt2real.com/ [Accessed: June 4, 2018].

[11] Emlid, "Navio2." [Online]. Available: https://emlid.com/navio/ [Accessed: May 26, 2018].

[12] Ardupilot, "APSync." [Online]. Available: http://firmware.ap.ardupilot.org/Companion/apsync/ [Accessed: May 23, 2018].

[13] D. Brodnevs and A. Kutins, "Deterioration Causes Evaluation of Third Generation Cellular LTE Services for Moving Unmanned Terrestrial and Aerial Systems," *unpublished*.

[14] D. Brodnevs and A. Bezdel, "High — Reliability low — Latency cellular network communication solution for static or moving ground equipment control," in *2017 IEEE 58th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, 2017, pp. 1–6. https://doi.org/10.1109/RTUCON.2017.8124756

[15] D. Brodnevs and A. Kutins, "Reliable data communication link implementation via cellular LTE services for static or moving ground equipment control," *unpublished*.

[16] Sakaki, "Bootable 64-bit Gentoo image for the Raspberry Pi 3B/B+." [Online]. Available: https://github.com/sakaki-/gentoo-on-rpi3-64bit#kernelbuild [Accessed: June 3, 2018].

[17] "Issue $1440: GCS Link Quality indicator incorrect with MavProxy redundant connection," 2016. [Online]. Available: https://github.com/ArduPilot/MissionPlanner/issues/1440 [Accessed: May 23, 2018].

Deniss Brodņevs graduated from Riga Technical University, Aviation Institute, as an avionic engineer of aircraft technical maintenance in 2013.
From 2006 to 2007 – Maintenance and upgrade of oil/gas tanker vessel computer equipment, server and network engineer in SQS Ltd. From 2011 to 2013 – Chief laboratory technician at Riga Technical University, Aviation Institute. From 2014 to 2015 – Aviation component fatigue testing automation electronic engineer. From 2013 to 2016 – Lecturer at Riga Technical University, Institute of Aeronautics.
Study courses: Aircraft Communication Systems, Aircraft Electrical Power Supply Systems, Aircraft Engine Control Systems, Modern fiber optics systems in Aviation.
Address: Institute of Aeronautics, Faculty of Mechanical Engineering, Transport and Aeronautics, Riga Technical University, Lomonosova 1A, k-1, Riga, LV-1019, Latvia.
Phone: (+371) 67089990
E-mail: Deniss.Brodnevs@rtu.lv