SAND2023-04294C

Commercial-Off-The-Shelf (COTS) Hardware Implementation Method for Real-Time Control of Mobile Robotics and UAVs

Christian Coletti, Kyle A. Williams, Julie Parish
Sandia National Laboratories
Albuquerque, New Mexico
ctcolet@sandia.gov, kwilli3@sandia.gov, jparish@sandia.gov

Research and development into mobile robotics such as unmanned aerial systems typically requires the integration of the robotic system with off-board components (*e.g.* motion capture systems or groundstations). Use of commercial-off-the-shelf (COTS) systems such as remote control unmanned aerial vehicles (UAVs) and transmitters is attractive due to the components' maturity, reliability, and cost-effectiveness but is generally not feasible to implement due to closed-source embedded electronics and signal protocols. This paper describes to interface with COTS components in real-time to send commands to mobile robotic platforms. Commands are wirelessly streamed to a microcontroller, which converts these commands to a Pulse Position Modulation (PPM) signal. This protocol is accepted in the training port of a transmitter, establishing control of the UAV with no modification to COTS systems. This method has been flight-tested and proves to be (i) quick to deploy (ii) sufficiently low-latency for control of small, agile, fixed-wing aircraft, (iii) compatible with all standard transmitters, and (iv) easily extended to multi-agent control. Integration with motion capture systems may significantly reduce cost and remove barriers of entry in the design, development, and test of new robotic systems and controllers.

I. Introduction

Mobile robotics research and development typically requires significant investment and development with the sensing, communication, and control stack. Using a motion capture system or groundstation for software-in-the-loop, hardware-in-the-loop, or any other off-board subprocesses can simplify the development of new control and teaming algorithms. In order to send commands to the robotic platform, the platform itself must be able to interface with the groundstation and motion capture system computers. Especially with UAVs such as Figure 1, it is advantageous to use commercial-off-the-shelf (COTS) platforms and components to realize potential benefits as compared to custom-built aircraft [1]. These benefits may be (i) reduction of cost, (ii) reduction of time to implementation, (iii) reduction of system technical complexity, (iv) replaceability, and (v) system reliability.

However, the required communication is typically not feasible to COTS robotic platforms and significant development is needed to design components to replace these COTS systems for development. This is because of a number of COTS systems are built with proprietary, embedded electronics that combine to control typical COTS systems. For UAVs, the aircraft radio receiver and onboard flight controller are typically designed to only interface with a specific set or brand of proprietary radio frequency transmitters using complex protocols [3]. In order to both increase the quality of their product and to prevent off-brand transmitter usage, the embedded systems in the transmitter, receiver, and flight controller are not intended to be interfaceable, and their protocols are typically not released. Thus, there exists a barrier preventing utilization of typical COTS aircraft for any form of hardware implementation of off-board sensing simulation, guidance, or control.

Fig. 1 Ultrix COTS Aircraft [2]

Nevertheless, there is an opportunity to use the reliability of COTS components with computer-based groundstations. Typical transmitters use a standardized, wired communication protocol typically used for training new pilots. In intended use, one transmitter, termed the slave, does not send

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

radio signals but instead sends commands over a cable to a secondary "instructor" transmitter, which receives and transmits the commands. This work uses a microcontroller to send commands through the training port as an entry point for computer-based control (Figure 2)

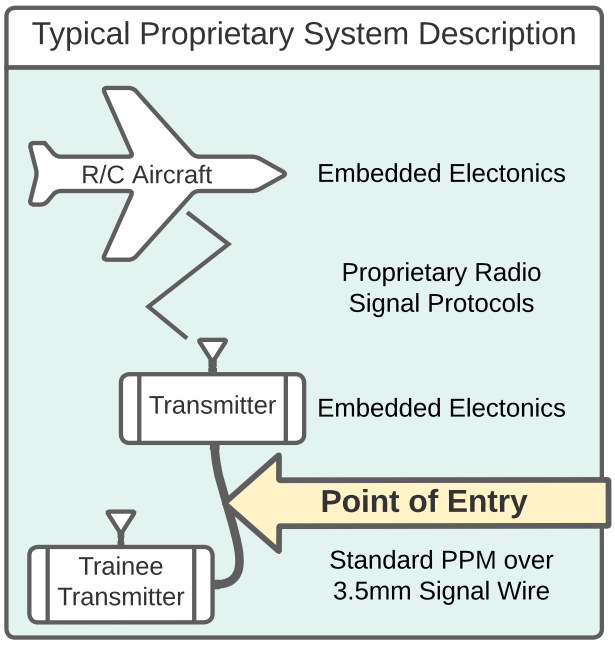


Fig. 2 Existing Systems with Point of Entry Noted

The gained benefits also include the ability for the microcontroller's commands to be switched from or overridden easily by the instructor using existing transmitter settings. This is a great benefit in controlling to a trim state before switching to computer control, as well as manually mitigating crash damage in terminal flight.

The ability to capably use COTS platforms and components in research and development may significantly impact the cost and technical burden of early-stage testing. Modern research in guidance, navigation, control (GNC) and autonomy, including aerial autonomy with neural-network-in-the-loop (NNIL), would likely benefit from the low-cost, wireless interface method described.

II. Previous Work

Use of Commercial-off-the-Shelf (COTS) components or vehicles is considered a valuable source of innovation [1] and may reduce cost and increase the effectiveness of the subsequent systems. Use of COTS components with UAVs include efforts to fly an aircraft with joystick or GPS points was explored in [4], to test avionics or electronics [5] [6], or to study swarming aircraft [7], with [4] noting that "cost reduction for processes and equipment" is "of specific concern of the aeronautical industry." Multiple studies consider COTS components for spacecraft, including development of subsystems in satellites [8] including telemetry, tracking, command [9], and propulsion [10] (albeit with potential space-related downsides [11] [12]). In fact, interest in incorporating COTS equipment ranges from use with internet of things (IOT) [13] and augmented reality (AR) testbeds [14] to livestock tracking [15] and medical devices [16].

The importance of robotic testbeds has also been endorsed in various literature, including relatively large research facilities [17]. These include various control [18] [19] and swarming testbeds [20] [21] [22] requiring "relatively small and inexpensive" [23] robotics to test with, with one noting generally that "multi-agent mobile robotic testbeds are expensive, inaccessible, and restricted characteristics" [24]. Thus, many of the robotics within those papers have similar motivations, and this effort seeks to aid both with testbed robotics, but also the design, development, test, and evaluation of new robotics and controllers.

III. Methods

A. Overview

The process described in the following sections use cover five systems connected that form a control loop. These systems are

1) a groundstation computer capable of translating state information into commands and sending those commands to the microcontroller
2) a microcontroller which receives a command and sends a corresponding Pulse Position Modulation (PPM) command over the training port
3) a COTS transmitter which receives the PPM command and transmits it to the robotic platform
4) a robot which actuates the transmitted command, and
5) a sensing and/or estimation loop which transmits the robot state to the groundstation computer.

The combination of systems in 1 and 2 form a novel technique while use of systems 3 and 4 are described in sufficient

detail but are standard COTS components. System 5 is a generalized to any sufficient sensor and/or estimator and is not detailed, but may include motion capture systems.

The specific hardware referenced for example in this study are not required; any number of similar COTS components may be used. Typical transmitters have training link capability, which is the only requirement that allows any combination of transmitter-robot systems to be controlled through this process. Similarly, any microcontroller may be used as long as it can receive commands from the groundstation computer and generate PPM signal with sufficiently low latency for the application.

In order to use the process, the system must be initialized into a state ready to begin control, after which point the loop follows the diagram in Figure 6.

B. Setup

The following components are assumed to be in place. First, a control program (using a neural-net or otherwise) is implemented based off of state estimation with sufficiently low latency for this application and can be run in real-time on a groundstation computer. Second, the groundstation computer is able to receive sufficient state information about the robotic platform when in use. Third, a transmitter-robot system is in place and the robot is bound to the transmitter such that commands sent to the robot are actuated. Lastly, there is sufficient wireless network coverage and bandwidth between the groundstation computer and the microcontroller for low-latency communication (contemporary laptops or smartphones typically have the ability to generate such a hotspot if needed). The final step may be modified to be a wired connection with the following method reasonably adapted where necessary.

1. Microcontroller PPM

As seen in Figure 2, the intended control signal enters the control loop by a specific connection on a COTS transmitter. This connection is between a microcontroller and the training port of a COTS transmitter, which was discussed in Section I. This connection requires an analog signal where commands are communicated by a series of high and low voltage states with specific pauses to communicate the command. All commands must happen within a prespecified amount of time to complete the signal, or a *frame*. This is the basis of Pulse-Position Modulation (PPM) shown in Figure 3.

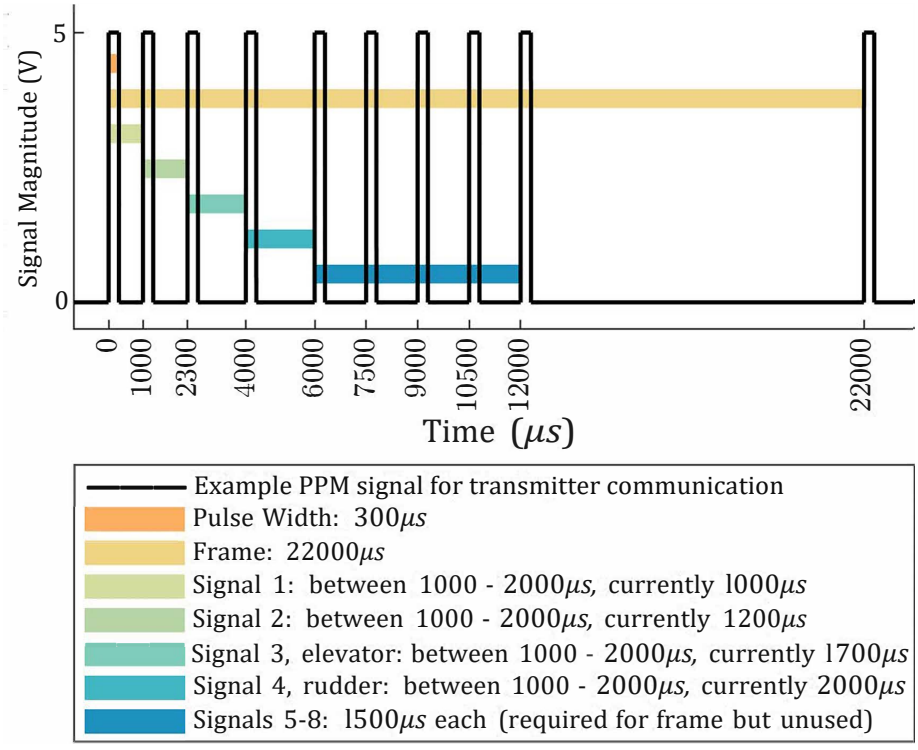
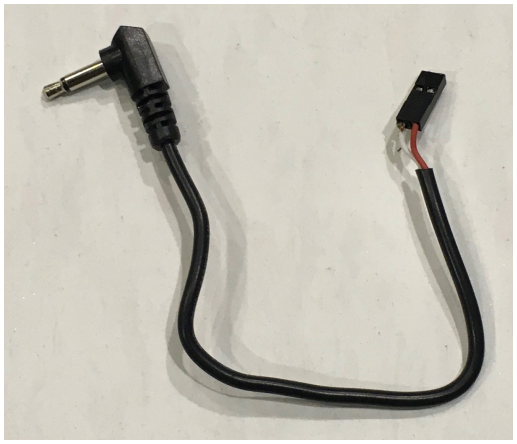


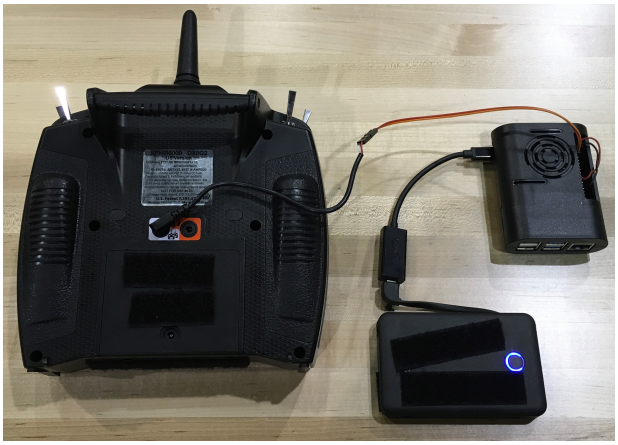
Fig. 3 Example PPM Signal

In a single frame, eight channels are expected by the receiving transmitter, specified by delays in rising signals between 1000 microseconds (μs) (the minimum signal) and 2000 microseconds (the maximum). Typically, transmitters use Aileron-Throttle-Elevator-Rudder (ATER) ordering for channels 1-4, but some (such as Spektrum) use Throttle-Aileron-Elevator-Rudder (TAER) ordering. Channels 5-8 may be used for other switches, such as gear or flight mode, but are unused in this paper. Despite this, channels 5-8 are specified to match the number of channels expected by the receiving transmitter. For throttle, the minimum command corresponds to 0 throttle, but for aerodynamic surface actuation the minimum signal corresponds to the maximum command deflection in the negative direction and 1500 microseconds corresponds to no deflection. The maximum accepted amount of between rising voltage signals is 2000 μs and corresponds to the maximum command in the positive direction for all channels. The command with zero actuation is then $[ATER] = [1500, 1000, 1500, 1500]\ \mu s$ or $[TAER] = [1000, 1500, 1500, 1500]\ \mu s$.

This waveform is constructed on a microcontroller (*e.g.* a Raspberry Pi 4) using the required utility (*e.g.* pigpio [25]). For pygpio, the waveform specification is sent to pigpio's daemon process that generates the wave on the GPIO. The pigpio daemon continually sends the wave to the GPIO until a new wave is commanded (or the daemon is stopped), which is advantageous as it allows the constant connection expected by the transmitter. This wave is carried by the training cable, a standard two-wire 3.5mm male-to-any cable (note that the more common three-wire 3.5mm cable may not work as intended). The side opposite a male end is cut and connected to the GPIO, while the male end is inserted in the training port of the transmitter. For reference, the modified training cable can be seen in Figure 4a. This cable connects the GPIO board pins to the training port, transmitting the PPM analog signal.



(a) The training cable that connects the systems



(b) Connection of the transmitter-side components

Fig. 4 Transmitter hardware components

2. UDP Connection Setup

In order to receive the commanded signals, the microcontroller must be set up to receive information from the groundstation. Many methods are viable, but a UDP socket is especially attractive due to the low-latency connection. User Datagram Protocol (UDP) is a type of one-way, low-latency, loss-tolerating connection uses the Internet Protocol (IP) to send data [26]. A network socket is an endpoint with a specified port and address for sending or receiving data. In order to establish this connection, the groundstation and microcontroller must be able to connect via IP (or, if not using UDP, otherwise be able to receive data from the groundstation computer).

On the groundstation side, a socket is established targeting the microcontroller's IP address and continually sends these PPM timing signals to the microcontroller (e.g. $[1000, 1200, 1700, 2000]$) in JavaScript Object Notation (JSON) format [27]. Similarly, on the microcontroller side, a socket is established to receive this data, and decode it from JSON format before generating a wave object. The receiving socket buffer is set to a value between the maximum size of a single UDP packet and a low multiple of that size. Since more recent data is added to the end of a receive buffer, it is best to make the receive buffer only large enough to store one or two commands. This setup ensures a low-latency connection between the groundstation and the microcontroller.

It may be advantageous to start a different connection that allows for establishing and debugging the microcontroller and data connection without access to the microcontroller directly, and a Flask server was established for these reasons

[28]. This this setup, the groundstation automatically requests a web address hosted by the microcontroller that starts/restarts the UDP port and command generation processes of the microcontroller. By running Flask in development mode, error output and arbitrary code execution can be done from the groundstation for debugging.

3. Transmitter Setup

Configuring a transmitter for training mode (which is required for this method) is transmitter-specific and outlined in the manual, but shares some generalities. The training cable connection point is typically in the center of the rear of the device, and marked with two controllers. The settings for the trainer link detail two settings, either a trainer mode or a slave mode; this process requires the transmitter be set to the trainer mode. Some additional settings may optionally be set for various uses in development and testing, such as disabling or overriding throttle control. This can be handled in most transmitter trainer options menu without modifications, enabling more granular and safe control over potentially dangerous components such as throttle or even allowing a pilot to manually recover a vehicle. Additionally, control of the vehicle is typically set with a switch on the transmitter that toggles between pilot control and microcontroller control, and can likely be configured in transmitter settings if desired.

4. Holding the microcontroller and battery to the transmitter

The battery and microcontroller were initially held to the transmitter with typical velcro, which worked fairly well but did not stand up to detachment cycling and eventually became unreliable. An additional component was designed and printed to hold the microcontroller and battery, seen in Figure 5. This system is designed to keep the battery, microcontroller, and transmitter connected and secured so that there are no stresses on the connections during use.

(a) Front view **(b) Rear view**

Fig. 5 3D printed holder for microcontroller and battery. The microcontroller pictured in Figure 4b is placed on the bottom slot, while the battery is in the middle and the transmitter is attached on top.

C. Nominal Operation

1. Groundstation Computer

Information received from the sensing and/or estimation loop (in this case a motion capture system) is processed into a state compatible with the controller. For a neural network control of a UAV, the state mapping to control action is truncated into acceptable ranges associated with the limits of actuation on the aircraft, *e.g.* ± 15 deg deflection. These values are then mapped to their corresponding PPM signal timings in microseconds with special care taken to ensure the correct direction of the deflection, e.g. 3 deg aileron in the positive direction is $1600\mu s$. These values are formatted into JSON and sent over a wireless internet connection using the UDP socket. This process loops each time state information is received if the process is not already processing state information.

2. Microcontroller

The microcontroller used in this case is a Raspberry Pi 4B and is connected to the same wireless internet connection as the groundstation computer. Once a UDP packet is received, the waveform is generated using pigpio and sent to the daemon. These commands are also logged (with a timestamp) as not all commands sent by the groundstation are received by the microcontroller, due to the lossy protocol and small receive buffer. Using the gpio board, the pigpio daemon sends the commands over the training cable to the transmitter. Similar to the groundstation computer, this process loops each time state information is received if the process is not already working (to prevent race errors).

3. Transmitter

The transmitter and robot used were a Spektrum DX8 Generation 2 as seen in Figure 4b and a FliteTest UMX Ultrix aircraft (Figure 1). Once the transmitter receives commands over the training cable, the transmitter transmits the commands (if the required switch is set to allow this passthrough). Additionally, control was set to be completely by moving the transmitter gimbals, which allows the throttle to be cut and/or a crash to potentially be averted.

4. Robotic System

The robot then receives these commands and actuates accordingly. Since the Ultrix uses elevons, the onboard flight controller maps incoming aileron and elevator input commands to elevons, and thus the traditional 4-channel control can still be established; similarly, rudder is mapped to differential thrust. This process works with aircraft with separate aerodynamic control surfaces as well.

IV. Results

The system runs sufficiently quickly to establish control of a small UAV with a wingspan of 342 mm (Figure 1), receiving and generating new PPM signals at 50Hz. This is sufficient for many robotics platforms, including most fixed wing systems, quadrotors, wheeled robotics, or most other robotic systems. Implementation of this method to test a new neural-network-in-the-loop (NNIL) controller [29] with the aircraft (which was unflown at the time) was reduced to about 2 weeks of work for an single engineer (with an existing motion capture system in place), as compared to several months for a quadcopter with PixHawk [30]. Data logging was significantly simplified by logging commands onboard the microcontroller and enabled better system identification in postprocessing.

Multiple simultaneous flights were sustained with a single groundstation computer sending commands to two or more microcontroller-transmitter-UAV systems without a noticeable increase in latency.* At the moment, no limitations are found if the system can be controlled when a command rate of 50Hz or less, excepting limitations in the motion capture system or hardware device.

While a NNIL controller could not have been feasibly implemented on the aerial vehicle itself due to both the computational burden and

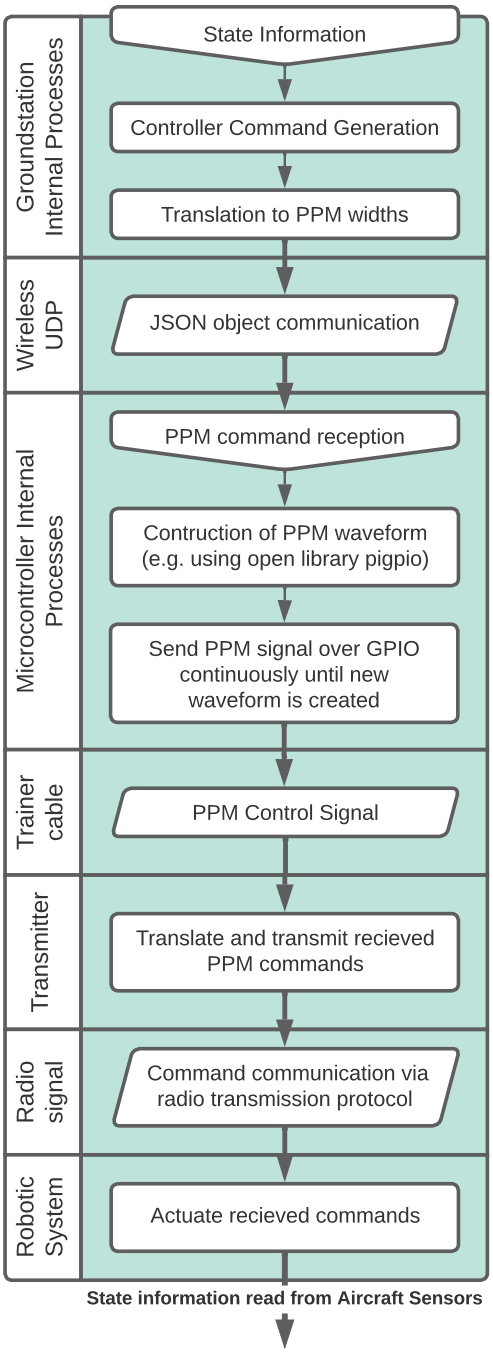


Fig. 6 Process Overview

*Results can be seen here `www.youtube.com/watch?v=SFrgWiNF3Ic`

the difficulty of interfacing with embedded electronics, this method allowed a groundstation to send the commands through COTS hardware to control the aerial vehicle. This avoids use of Robot Operating System (ROS) [31] and either Pixhawk [30] or PX4 [32], simplifying the implementation and allowing near-direct control of actuation. In this way, advanced robotic systems and/or developmental controllers can be designed, tested, and evaluated more directly, easily, and affordably.

V. Conclusion

This study primarily outlines a method for scientists and researchers to interface with commercial-off-the-shelf (COTS) hardware, including transmitters, antennas, receivers, and robotic systems. By generating an analog signal through the training port of COTS systems, existing COTS infrastructure, which was previously inaccessible due to proprietary protocols, can be used with custom control algorithms or motion capture systems interfaceable through a groundstation computer. The work is implemented on representative electronics and tested on a characteristically difficult system to control due to the nonlinear dynamics and fast dynamics that require low-latency control. However, this method was able to control multiple small UAVs at 50 Hz after only two weeks of development, significantly improving on implementation with ROS and Pixhawk which had taken several months for a quadcopter.

This work opens the door to more attainable research studying the neural network reality gap problem (also called sim2real) or any other advanced or untested guidance and/or control scheme. Additionally, the logging and direct actuation at the point of command enable more accurate system identification due to a better measure of the actuated control with the system. This method has benefit at various levels of development, being useful whenever an external groundstation is required or advantageous in the research or testing of a robotic platform. Proven long-range transmitters and low-cost COTS systems can be used with developmental algorithms or methods at a large scale or in-environment to control things such as mutliagent coordination or other high-level commands.

Acknowledgments

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. Approved for Unclassified Unlimited Release (UUR) by SAND 2022-12494 C

References

[1] Gansler, J. S., and Lucyshyn, W., "Commercial-off-the-shelf (cots): Doing it right," Tech. rep., Maryland Univ College Park Center for Public Policy and Private Enterprise, 2008.

[2] E-flite, "UMX Ultrix," `https://www.e-fliterc.com/product/umx-ultrix-bnf-basic-with-as3x-and-safe-select-342mm/EFLU6450.html`, 2020.

[3] Liang, O., "FPV Protocols Explained," `https://oscarliang.com/rc-protocols/`, Apr 2021.

[4] Rangel, R. K., Kienitz, K. H., and Brandao, M. P., "Development of a complete UAV system using COTS equipment," *2009 IEEE Aerospace conference*, 2009, pp. 1–11. https://doi.org/10.1109/AERO.2009.4839603.

[5] Tso, P., and Galaviz, P., "Improved aircraft readiness through COTS," *1999 IEEE AUTOTESTCON Proceedings (Cat. No.99CH36323)*, 1999, pp. 451–456. https://doi.org/10.1109/AUTEST.1999.800414.

[6] Pham, K. L., Leuchter, J., Pham, N. N., and Pham, V. T., "Design of Commercial-Off-The-Shelf System for Monitoring UAV's Accumulator," *2021 International Conference on Military Technologies (ICMT)*, 2021, pp. 1–7. https://doi.org/10.1109/ICMT52455.2021.9502781.

[7] Srivastava, D., Pakkar, R., Langrehr, A., and Yamane, C., "Adaptable UAV Swarm Autonomy and Formation Platform," *2019 IEEE Aerospace Conference*, 2019, pp. 1–6. https://doi.org/10.1109/AERO.2019.8741683.

[8] Karvinen, K., Tikka, T., and Praks, J., "Using hobby prototyping boards and commercial-off-the-shelf (COTS) components for developing low-cost, fast-delivery satellite subsystems," *Journal of Small Satellites*, Vol. 4, No. 1, 2015, pp. 301–314.

[9] Lovascio, A., D'Orazio, A., and Centonze, V., "Design of a Telemetry, Tracking, and Command Radio-Frequency Receiver for Small Satellites Based on Commercial Off-The-Shelf Components," *International Journal of Aerospace and Mechanical Engineering*, Vol. 13, No. 9, 2019, pp. 564–573.

[10] Gibbon, D., Cowie, L., and Paul, M., "Cots (commercial off the shelf) propulsion equipment for low cost small spacecraft," *38th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, 2002, p. 3994.

[11] Shirvani, P. P., Oh, N., Mccluskey, E. J., Wood, D., Lovellette, M. N., and Wood, K., "Software-implemented hardware fault tolerance experiments: COTS in space," *International Conference on Dependable Systems and Networks (FTCS-30 and DCCA-8), New York (NY)*, 2000.

[12] Underwood, C., and Oldfield, M., "Observed radiation-induced degradation of commercial-off-the-shelf (COTS) devices operating in low-earth orbit," *IEEE Transactions on Nuclear Science*, Vol. 45, No. 6, 1998, pp. 2737–2744.

[13] Ghose, N., Lazos, L., and Li, M., "In-Band Secret-Free Pairing for COTS Wireless Devices," *IEEE Transactions on Mobile Computing*, Vol. 21, No. 2, 2022, pp. 612–628. https://doi.org/10.1109/TMC.2020.3015010.

[14] Behringer, R., Tam, C., McGee, J., Sundareswaran, S., and Vassiliou, M., "A wearable augmented reality testbed for navigation and control, built solely with commercial-off-the-shelf (COTS) hardware," *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, 2000, pp. 12–19. https://doi.org/10.1109/ISAR.2000.880918.

[15] Karl, J. W., and Sprinkle, J. E., "Low-Cost Livestock Global Positioning System Collar from Commercial Off-the-Shelf Parts," *Rangeland Ecology and Management*, Vol. 72, No. 6, 2019, pp. 954–958. https://doi.org/https://doi.org/10.1016/j.rama.2019.08.003, URL https://www.sciencedirect.com/science/article/pii/S1550742419300582.

[16] Khan, S. R., Mugisha, A. J., Tsiamis, A., and Mitra, S., "Commercial Off-the-Shelf Components (COTS) in Realizing Miniature Implantable Wireless Medical Devices: A Review," *Sensors*, Vol. 22, No. 10, 2022. https://doi.org/10.3390/s22103635.

[17] Pickem, D., Glotfelter, P., Wang, L., Mote, M., Ames, A., Feron, E., and Egerstedt, M., "The Robotarium: A remotely accessible swarm robotics research testbed," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1699–1706. https://doi.org/10.1109/ICRA.2017.7989200.

[18] Goyal, S., Wang, W., and Brambley, M. R., "An agent-based test bed for building controls," *2016 American Control Conference (ACC)*, 2016, pp. 1464–1471. https://doi.org/10.1109/ACC.2016.7525123.

[19] Reiser, M. B., and Dickinson, M. H., "A test bed for insect-inspired robotic control," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, Vol. 361, No. 1811, 2003, pp. 2267–2285.

[20] Humann, J., Pollard, K. A., Hill, S. G., Ayorinde, O. A., Foots, A. N., and You, S., "Physical Robot Swarm Testbed at ARL: Specifications and Experimental Design Possibilities," Tech. rep., CCDC Army Research Laboratory, 2019.

[21] Jones, S., Milner, E., Sooriyabandara, M., and Hauert, S., "DOTS: An Open Testbed for Industrial Swarm Robotic Solutions," , 2022.

[22] Michael, N., Fink, J., and Kumar, V., "Experimental testbed for large multirobot teams," *IEEE robotics & automation magazine*, Vol. 15, No. 1, 2008, pp. 53–61.

[23] Motter, M., Logan, M., French, M., and Guerreiro, N., "Simulation to flight test for a UAV controls testbed," *25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference*, 2006, p. 3305.

[24] Ospina, N. I., Mojica-Nava, E., Jaimes, L. G., and Calderón, J. M., "ARGroHBotS: An Affordable and Replicable Ground Homogeneous Robot Swarm Testbed," *IFAC-PapersOnLine*, Vol. 54, No. 13, 2021, pp. 256–261. https://doi.org/https://doi.org/10.1016/j.ifacol.2021.10.455, URL https://www.sciencedirect.com/science/article/pii/S2405896321018929, 20th IFAC Conference on Technology, Culture, and International Stability TECIS 2021.

[25] Joan, "The pigpio library," https://abyz.me.uk/rpi/pigpio/index.html, Mar 2021.

[26] Postel, J., "RFC0768: User Datagram Protocol," , 1980.

[27] Crockford, D., "The application/json media type for javascript object notation (json)," Tech. rep., 2006.

[28] Ronacher, A., Brandl, G., Zapletal, A., Afshar, A., Edgemon, C., Grindstaff, C., et al., "Flask (a Python microframework)," *Dosegljivo*, 2010.

[29] Coletti, C. T., Williams, K. A., Lehman, H. C., Kakish, Z. M., Whitten, D., and Parish, J. J., "Effectiveness of Warm-Start PPO for Guidance with Highly Constrained Nonlinear Fixed-Wing Dynamics," *American Control Conference (ACC)*, 2023.

[30] Thiercelin, A., Galloway, I., Schwarzmiller, C., and Klimaj, A., "Pixhawk | The hardware standard for open-source autopilots," https://pixhawk.org/, 2008.

[31] Wyrobek, K., and Berger, E., "ROS: Home," https://www.ros.org/, 2007.

[32] Meier, L., Honegger, D., and Pollefeys, M., "PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms," *2015 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2015, pp. 6235–6240.