2024 International Conference "ROBOTICS & MEHATRONICS"
29 – 30 October, 2024, Sofia, Bulgaria

Design of a testing model for evaluation the levels of automation and autonomy of a helicopter autopilot

Stefan Hristozov
Unmanned Robotics Systems Laboratory
Institute of Robotics-Bulgarian Academy of Sciences
Sofia, Bulgaria
st.hristozov@ir.bas.bg

Anastas Madzharov
Unmanned Robotics Systems Laboratory
Institute of Robotics-Bulgarian Academy of Sciences
Sofia, Bulgaria
a.madzharov@ir.bas.bg

Neda Chehlarova
Unmanned Robotics Systems Laboratory
Institute of Robotics-Bulgarian Academy of Sciences
Sofia, Bulgaria
nedachahlarova@ir.bas.bg

***Abstract*—The development of Unmanned Aerial System (UAS) helicopter autopilots presents unique challenges compared to other flying platforms, particularly due to the complex dynamics and mechanics of the rotorcraft i.e. helicopter. This paper addresses key issues encountered during the design and testing phases of a UAS helicopter autopilot system and the creation of proper test cases. One of the primary challenges is the selection of a suitable testing platform. Factors such as the size, endurance, and cost of remote-controlled (RC) helicopters significantly influence platform choice.**

This work also explores the utilisation of open-source autopilot systems, such as ArduPilot, PX4, Librepilot & Rotorflight, representing multiple generations of available open-source autopilots. While open-source systems provide a valuable foundation, their integration with rotary-wing platforms, especially for advanced functions like swashplate control all the way to automatic mission execution, poses significant hurdles. Through successive generations of autopilot design evaluation, we address these challenges.

By discussing the trade-offs involved in platform selection, the intricacies of swashplate for autopilot modelling, and the role of open-source autopilots in rotorcraft control, this paper contributes insights for future UAS helicopter autopilot development, aiming to enhance performance and reliability, but also to evaluate its impact on safety and risk assessment.

***Keywords*—helicopter, autopilot, test bench, open-source**

I. INTRODUCTION

The development of Unmanned Aerial System (UAS) helicopter autopilots presents unique challenges compared to other flying platforms, particularly due to the complex dynamics and mechanics of the rotorcraft. Still, there is a significant progress that can be registered in the development of theses flying vehicles and their stabilisation and flight control systems. Their applicability reaches out to wide variety of use-cases in civil and industrial domains. These UAs incorporate a wide range of technological innovations, showing immense potential for both state (military) and civil [1]. Even if initial developments were made on fixed-wing UAs, recently a rapid growth of multi-rotor and helicopter platform is registered [2].

When it comes to modelling and designing an autopilot for helicopter flight platform it is sensibly more challenging and complex than their fixed-wing counterparts, mainly because of the highly interconnected nature of the system and the limitations of standard blade theory in deriving accurate aerodynamic forces. By further reducing the size of the platform to a small RC model additional challenges are presented, reducing the time constant even more and requiring further increase in response and precision of the control system. Furthermore, the helicopter's nonlinear and inherently unstable nature adds to the complexity of stabilisation [3] [4]

Designing a proper testing platform is crucial for aeronautical research and education. It not only provides environment to conduct hands-on data gathering, but also enable experimentation in search of the new and the better. By combining software-in-the-loop (SITL)_ and hardware-in-the-loop (HITL) such a test-bed minimises additionally the requirement to the flying platforms and furnish their users with relatively low cost and more accessible virtual environments [5]. Major use cases in education and training include conducting familiarisation exercises for the basic concepts of flight and performing simulation-based verification [6] [7].

In this paper a design of testing platform is performed in regards to preserving the helicopter flight characteristics and mechanics and while still maintaining the necessary real-estate to accommodate the autopilots in test and their periphery, which in some cases can be even more difficult. Considering also that part in such initial state of the research there is no funding for this research the cost also poses a huge obstacle. In order to bypass it a more budget platform is selected, not compromising the helicopter main mechanics.

II. TESTING PLATFORM SELECTION

When considering all the factors that influence the choice of testing platform and in this case an actual helicopter platform few things have to be taken into account. As first comes the selection of the proper size aircraft in order to accommodate the tested autopilots and their respective periphery which in many cases can be more demanding. Also wiring the frame so that there is no interference from other

ACKNOWLEDGE THE FINANCIAL SUPPORT OF THE PROJECT WITH ADMINISTRATIVE CONTRACT № KP-06-H57/8 FROM 16.11.2021. - FUNDED BY THE "COMPETITION FOR FUNDING BASIC RESEARCH - 2021." FROM THE RESEARCH SCIENCES FUND, BULGARIA.

parts and systems of the helicopter is better allowed on larger airframes [8].

The second factor to consider when selecting an RC helicopter endurance. It can vary significantly among these helicopters due to factors such as size, power source, rotor configuration, and overall design. For example, small electric RC helicopters typically have endurance of 5 to 15 minutes due to battery limitations, while larger, fuel-powered RC helicopters may reach 30 to 60 minutes or more. Small electric models often sacrifice endurance for agility, while larger or fuel-powered helicopters trade manoeuvrability for extended flight times.

Also balancing endurance with cost is crucial, as longer-endurance platforms are generally more expensive, impacting the feasibility of large-scale testing, which leads to last consideration in this article.

Three Align platforms are considered and brand is just for reference and not binding to this research. It could be any other helicopter with the same setup of the mechanics and swashplate:

- Trex 250: This smaller model typically costs between \$200 and \$300 for a basic kit, not including additional components like batteries, controllers, or specific upgrades for improved performance. It’s suitable for shorter tests and confined spaces but has limited endurance and payload capacity.
- Trex 450: Priced around \$400 to \$600, the Trex 450 provides a good balance between size and capability. It has moderate endurance and is more stable than the Trex 250, making it popular for testing with some flexibility in outdoor conditions. However, it’s still limited in carrying additional payload or sensors.
- Trex 600: This larger, more powerful model can cost from \$700 to over \$1,000, depending on the setup and optional features. It offers superior endurance and stability, making it more suitable for carrying additional testing equipment or running longer experiments but comes with a substantial increase in both initial cost and operating expenses.

TABLE I. COMPARISON BETWEEN THE THREE MODELS

Characteristics	**Model Helicopter**		
	Trex 250	***Trex 450***	***Trex 600***
Length	431mm	660mm	1160mm
Height	150mm	230mm	319mm
Main Blade Length	205mm	325mm	600mm
Main Rotor Diameter	460mm	700mm	1347mm
Tail Rotor Diameter	108mm	150mm	260mm
Motor Ponion Gear	15T	12T/13T	14T
Main Drive Gear	120T	150T	118T
Tail Drive Gear	28T	25T	34T
Drive Gear Ratio:	1:8:4.28	1:12.5:4.24/ 1:11.5:4.24	1:8.42:3.85
Weight (with Motor):	260g	450g	2900g

Fig. 1. *Trex 450 class helicopter*

The Trex 450 is selected since it provides a good balance between size and capability. It has moderate endurance and is more stable than the Trex 250, making it popular for testing with some flexibility in outdoor conditions. Despite its limited carrying capability it comes at a reasonable price and operational expenses and not like the other counterpart Trex 600. The Trex 450 design preserves the classic swashplate and rotor mechanics, which are critical for gaining a realistic understanding of flight dynamics in a smaller, more affordable platform, making it ideal candidate for controlled experiments and development phases in UAS autopilot systems.

III. SWASHPLATE MOVEMENT MODELING

In aeronautics the swashplate is a critical component in helicopter control, enabling changes in blade pitch that allow the helicopter to perform complex manoeuvres. It is a mechanical device that translates input via the helicopter flight controls into motion of the main rotor blades. In the words of an RC helicopter pilot the swashplate is designed to take the servo inputs that are based on the pilot’s pitch, roll, and collective commands and translate them to individual blade pitch inputs. Because the main rotor blades are spinning, the swashplate is used to transmit three of the pilot's commands from the non-rotating fuselage to the rotating rotor hub and main blades [9].

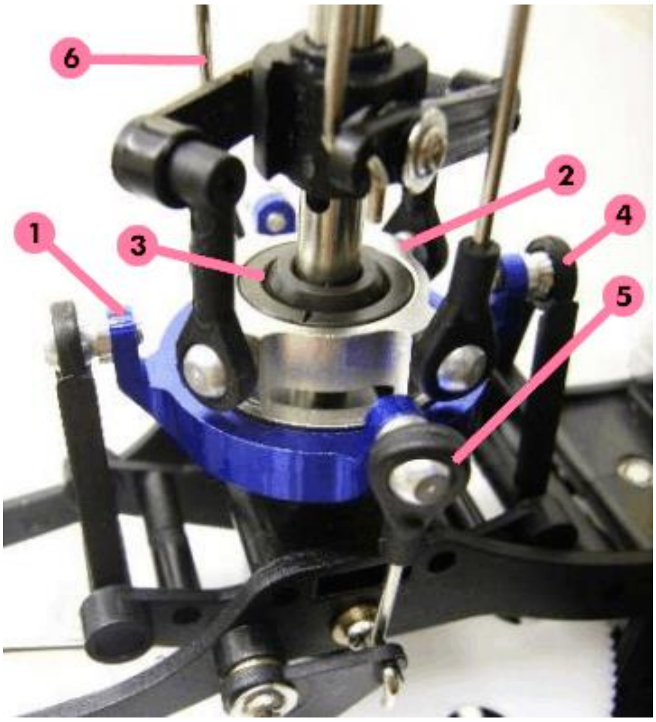

Fig. 2. *RC Helicopter swashplate*

1. Non-rotating outer ring (blue) or baseplate (stationary). 2. Turning inner ring (silver) (rotating). 3. Ball joint or sphere (rotating). 4. Pushrod Control (pitch) preventing turning of outer ring. 5. Pushrod Control (roll). 6. Swashplate-Blade Linkages (silver) to the rotor blade (3 in number, rotating). 7. Rotor Hub (rotating) (not shown) [10].

The swashplate consists of two main parts: a stationary swashplate and a rotating swashplate. The stationary (outer) swashplate is mounted on the main rotor mast and is connected to the cyclic and collective controls by a series of pushrods. It is able to tilt in all directions and move vertically. The rotating (inner) swashplate is mounted to the stationary swashplate by means of a bearing and is allowed to rotate with the main rotor mast. An anti-rotation link prevents the inner swash from rotating independently of the blades, which would apply torque to the actuators. The outer swashplate typically has an anti-rotation slider as well to prevent it from rotating. Both swashplates tilt up and down as one unit. The rotating swashplate is connected to the pitch horns by the pitch links. Alternative mechanics to the stationary (outer) swashplate are the hexapod and the universal joint [11]. Swashplates for helicopters having two rotors mounted on the same shaft are much more complex than the single rotor helicopters and out of the scope of the current research.

A. *Cyclic blade control*

Cyclic controls are used to change a helicopter's roll and pitch. Push rods or hydraulic actuators tilt the outer swashplate in response to the pilot's commands. The swashplate moves in the intuitively expected direction, tilting forwards to respond to a forward input, for instance. However, "pitch links" on the blades transmit the pitch information way ahead of the blade's actual position, giving the blades time to "fly up" or "fly down" to reach the desired position. That is, to tilt the helicopter forward, the difference of lift around the blades should be maximum along the left-right plane, creating a torque that, due to the gyroscopic effect, will tilt the rotor disc forward and not sideways [9].

B. *Collective blade control*

To control the collective pitch of the main rotor blades, the entire swashplate must be moved up or down along its axis without changing the orientation of the cyclic controls. Conventionally, each control mechanism, (roll, pitch, and collective) had an individual actuator responsible for the movement. In the case of pitch, the entire swashplate is moved along the main shaft by a one actuator. However, some newer model helicopters remove this mechanically complex separation of functionalities by using three interdependent actuators that can each move the entire swashplate. This is called cyclic/collective pitch mixing (CCPM). The benefit of CCPM is that smaller actuators can work together to move the swashplate across its full range of control, meaning the actuators can be smaller and lighter [9].

C. *Swashplate Leveling*

The swashplate can be levelled using either a tool specially designed to keep the swashplate perpendicular to the shaft or, a less expensive way, using a magnet and nail (shown below). Using the magnet and nail won't require you to remove your rotor head to level your swashplate. Rotate the shaft so the nail passes over the swashplate arm. Adjusts swashplate using one of methods below so the nail touches the top of each swashplate arm [12].

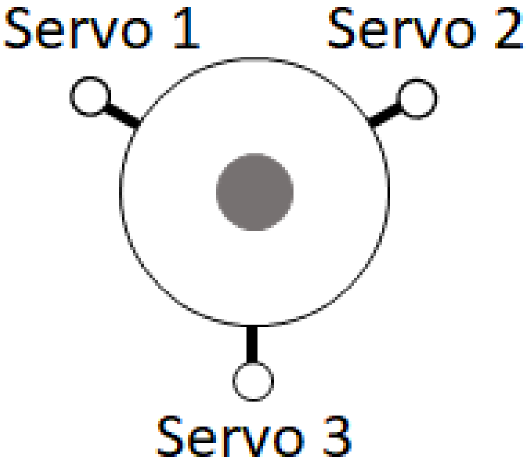

Fig. 3. *Software representation of a swashplate* [12]

The swashplate has undergone slight innovative adaptations aimed at reducing its complexity, maintenance, and associated costs. One such approach is the virtual swashplate concept, which utilizes rotor torque modulation combined with modified lead-lag hinges to adjust the angle of attack for each blade synchronously. By adopting this method, the traditional mechanical complexity is reduced, minimizing the need for multiple actuators and extensive mechanical linkages. This not only simplifies the control mechanisms but also reduces the wear and tear on the helicopter's moving parts [13].

IV. INTEGRATION OF OPEN-SOURCE AUTOPILOT SYSTEMS

Disclaimer that has to be made: Most of the following information presented in this section is to a large extent based on public available information from the web pages of the corresponding open-source projects developing the software and hardware. This approach however implies that missing information about the presence of a specific feature in a specific artifact is a threat to this research validity. Manually experimenting with all of the documented projects was however deemed unattractable. [14]

A. *Considered Open-Source Hardware Platforms (OSH)*

In the following section considered Open-Source Hardware platforms are reviewed and most notable features are presented with respect for helicopter control. Their documentation containing blueprints, mechanical and electronical drawings and schematics, bill of materials etc. are published under free license available online at [15] their respective repository.

a) *Pixhawk Series:*

The Pixhawk flight controller originated at the Computer Vision and Geometry Lab of ETH Zurich and has since grown into an independent OSH platform project supported by the Linux Foundation's DroneCode initiative. Built on the PX4-Flight Management Unit (FMU), Pixhawk has evolved through multiple versions to cater to a wide range of applications, offering modular and scalable options for developers and researchers.

The original Pixhawk model combines the PX4 FMUv2 with the PX4 IOv2 board, featuring an STM32F427 processor and an STM32F103 as a failsafe co-processor. Equipped with 256 KB of RAM. By late 2017, the PX4 FMU had advanced to FMUv5, incorporating the STM32F7 processor with a double-precision floating-point unit (FPU), enhancing computational precision, and relying on the Bosch BMI055 IMU for improved attitude accuracy. As of 2018, the PX4 FMUv6 incorporates the STM32H7 processor, allowing for even higher processing power and stability in autopilot functions.

b) CC3D:

The CC3D (CopterControl 3D) flight controller is developed by Librepilot (formerly OpenPilot), these controllers run the Librepilot firmware, reviewed later in this paper. The controller supports various airframe types and is released under the GPLv3 license, promoting open collaboration and modification [14].

c) H743-Wing:

H743-Wing is an OSH platform from Mateksys. It is based on the STM32H743VIT6 Microcontroller Unit (MCU) and published under the CC BY-NC-SA 4.0 license. The H743's high processing capacity and versatile connectivity make it suitable for integration with open-source platforms.

B. Considered Open-Source Software Platforms (OSS)

In the following section some of the considered Open-Source Software (OSS) flight controllers are reviewed with respect to helicopter control. Some of them are discontinued from developed, but that will not reflect future exploration of their architecture and functionality in terms of flight automation and autonomy levels. Their documentation containing code and manuals are published under free license available online at their respective repository.

a) MultiWii Series – Betaflight and Rotorflight development

The MultiWii originated from low-cost UAV control needs. On its basis Baseflight was developed. Cleanflight, originally forked from Baseflight, has since expanded its functionality. Betaflight emerged from Cleanflight, differentiating itself with a focus on high-performance features and experimental development. Known for supporting a wide range of hardware, Betaflight offers advanced tuning for UAV racing and acrobatic applications. INAV is another Cleanflight fork, emphasizing navigation functionalities like the "follow me" mode, making it ideal for GPS-based applications. With a mission planner that supports Windows, Linux, iOS, and Android, INAV has broadened the scope for autonomous flight [14].

Rotorflight is a recent fork of Betaflight designed specifically to optimize the control of UAVs with collective-pitch rotor systems, such as helicopters and custom multi-rotor configurations. Unlike Betaflight, which has focused on high-performance fixed-pitch quadcopters for racing and acrobatics, Rotorflight integrates additional functionalities tailored to rotorcraft needs, including collective-pitch control, which enables finer control over thrust and lift. This specialization allows for more advanced stabilization and manoeuvring in complex flight dynamics, where collective-pitch inputs are critical.

Rotorflight retains Betaflight's intuitive configuration tool as a Google Chrome extension, but it incorporates custom parameters for rotor-specific tuning, such as throttle curves and tail rotor mixing.

b) OpenPilot Series

The OpenPilot series includes a collection of flight controller software that has evolved through multiple projects and contributors. OpenPilot was the original software in this series; however, development ceased in 2015. Parts of its documentation remain accessible on its wiki, and the source code is still available. This software supports various flight controllers, including the CC3D. The Librepilot project emerged in July 2015, built on the OpenPilot foundation, Librepilot supports CC3D. Comprehensive online documentation provides setup guidance for supported boards, UAV configurations, sensor integration, and utilizing the Ground Control Station (GCS). Development seized in 2019.

c) Ardupilot

Ardupilot is a versatile open-source flight controller software capable of managing various vehicle types, including fixed-wing aircraft, multirotors, helicopters, boats, and even submarines. Originally developed for 8-bit ARM microcontrollers on its dedicated Ardupilot board, it transitioned to the Ardupilot Mega (APM) and has since evolved to optimise performance on 32-bit ARM microcontrollers.

Beyond embedded systems, Ardupilot is also compatible with Linux, allowing it to operate on a broad range of devices, from single-board computers to full PC systems. It features a desktop Ground Control Station (GCS) for mission planning, calibration, and vehicle configuration across Windows, Linux, and macOS.

d) PX4

The PX4 flight stack and autopilot is an open-source platform that is part of the DroneCode project, a collaborative initiative supporting Ground Control Stations (GCS), hardware platforms, and simulation tools. PX4 is highly adaptable, supporting a wide variety of airframes, including multirotors, fixed-wing aircraft, gliders, helicopters, and VTOL systems. The flight stack is compatible with the QGroundControl GCS, which facilitates parameter configuration, sensor monitoring, and autonomous flight management. PX4's performance and capabilities have been documented. Licensed under BSD, PX4 is attractive for commercial applications due to its permissive licensing and robust feature set [14].

V. EXPERIMENTAL SETUP AND DISCUSSION

By equipping four identical Align Trex 450 RC helicopters with the discussed autopilot systems (two CUAV v5 Nanos, one CC3D and one Mateksys H743) each running its respective firmware (Ardupilot, PX4, Librepilot and Rotorflight), a structured evaluation of the functionalities within different software architectures and generations of autopilot hardware (e.g., CC3D and H743) can be achieved. This setup enables a comparative analysis of each system's unique performance capabilities, sensor handling, control precision, and response characteristics.

The next step is to define test cases to measure these parameters effectively. Test cases will focus on key performance metrics, such as stabilisation accuracy, waypoint-following precision, responsiveness under wind disturbance, altitude hold effectiveness, overall system reliability and last but not least the level of automation and autonomy and how that does reflect other aspects in the overall systems of piloting.

Also to consider is the possibility of cross-pollination between different architectures and the possibility to 'migrate' certain modules [16].

Worth mentioning is additional simplification in the helicopter mechanics by introduction of virtual swashplate [13].

Evaluating these differences in real flight scenarios will offer insights into how each generation of autopilot handles real-world conditions and varying environmental inputs, providing data on suitability for different UAS applications and informing further development. But also having the test setup in place HITL and SITL still can be performed in order to spare time, reduce costs from crashes and collect data directly.

VI. CONCLUSION

This paper details the design and construction of a UAS platform intended to support the development and testing of an autopilots system, its functionalities and reactions to disruptors. The core aim was to create a robust, cost-effective hardware framework that allows safe and effective testing of autopilot algorithms.

Equipping four identical RC helicopters with the different autopilots previously discussed allowed for a detailed comparison of software and hardware functionalities across multiple generations (e.g., CC3D and H743). This approach facilitated an analysis of how each autopilot system's architecture and firmware version impacted flight performance and stability, providing valuable insights into their control accuracy, sensor responsiveness, and reliability. This testing environment can be later used to test the automation and autonomy levels and how that impacts other stakeholders (e.g. pilot, pilot reactions etc.).

For future research a series of test cases to quantitatively measure differences in flight performance need to be defined and conducted. The planned evaluations will focus on critical metrics such as altitude hold stability, waypoint-following precision, responsiveness under simulated environmental disturbances, and overall system robustness. This comparative data will guide future development and selection criteria for suitable autopilot platforms in diverse UAS applications, ultimately informing best practices for autopilot integration into various UAS configurations..

ACKNOWLEDGMENT

THE AUTHORS ACKNOWLEDGE THE FINANCIAL SUPPORT OF THE PROJECT WITH ADMINISTRATIVE CONTRACT № KP-06-H57/8 FROM 16.11.2021. "METHODOLOGY FOR DETERMINING THE FUNCTIONAL PARAMETERS OF A MOBILE COLLABORATIVE SERVICE ROBOT ASSISTANT IN HEALTHCARE", FUNDED BY THE "COMPETITION FOR FUNDING BASIC RESEARCH - 2021." FROM THE RESEARCH SCIENCES FUND, BULGARIA.

REFERENCES

[1] H. Stanev и S. Hristozov, „Applicability of JARUS SORA to State UAS Operations in Disaster Relief," *ENVIRONMENT. TECHNOLOGIES. RESOURCES. Proceedings of the International Scientific and Practical Conference,* pp. 244-250, 2024.

[2] T. Frigioescu, M. Dombrovschi, G. Badea, G. Cican, M. Condruz и D.-E. Crunțeanu, „Development of an UAV Platform for Autopilot Implementation and Validation," *Turbo,* том X, № 2, pp. 6-16, 2023.

[3] A. E. Barabanov и D. V. Romaev, „Helicopter Modeling and Autopilot Design," в *18th IFAC World Congress*, Milano, Italy, 2011.

[4] A. E. Barabanov и D. V. Romaev, „Design of Helicopter Autopilot," 2010. [Онлайн]. Available: https://www.researchgate.net/publication/277245708_Design_of_Helicopter_Autopilot. [Отваряно на 29 10 2024].

[5] X. Dai, C. Ke, Q. Quan и K.-Y. Cai,, „RFlySim: Automatic test platform for UAV autopilot systems with FPGA-based hardware-in-the-loop simulations," *Aerospace Science and Technology,* № 114, 2021.

[6] H. Wang, S. Chen, U. Durak и S. Hartmann, „Simulation Infrastructure for Aeronautical Informatics Education," в *50th Computer Simulation Conference,* Bordeaux, France, 2018.

[7] Z. Cheng, R. West и C. Einstein, „End-to-end Analysis and Design of a Drone Flight Controller," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* 2018.

[8] A. ab Wahab, R. Mamat и S. Shamsudin, „The Development of Autopilot System for an Autonomous UAV Helicopter Model," 2013. [Онлайн]. Available: https://www.researchgate.net/publication/41058033_The_Development_of_Autopilot_System_for_an_Autonomous_UAV_Helicopter_Model. [Отваряно на 29 10 2024].

[9] „Swashplate (aeronautics) - Wikipedia," [Онлайн]. Available: https://en.wikipedia.org/wiki/Swashplate_(aeronautics). [Отваряно на 29 10 2024].

[10] COMSOL, „Helicopter Swashplate Mechanism," [Онлайн]. Available: https://doc.comsol.com/6.0/doc/com.comsol.help.models.mbd.helicopter_swashplate/helicopter_swashplate.html. [Отваряно на 29 10 2024].

[11] J. F. Below и D. A. Miloslavich, „Dynamics of the Swash Plate Mechanism," в *International Compressor Engineering Conference*, 1984.

[12] „Traditional Helicopter – Swashplate Setup — Copter documentation," [Онлайн]. Available: https://ardupilot.org/copter/docs/traditional-helicopter-swashplate-setup.html. [Отваряно на 29 10 2024].

[13] J. Flores, S. Salazar, I. Gonzalez-Hernandez, Y. Rosales, R. Lozano, E. Salazar и B. Nicolas, „Control of Helicopter Using Virtual Swashplate," *Drones,* том VIII, № 7, p. 327, 2024.

[14] E. Ebeid, M. Skriver, K. H. Terkild, K. Jensen и U. P. Schultz,, „A survey of Open-Source UAV flight controllers and flight simulators,," *Microprocessors and Microsystems: Embedded Hardware Design,* № 61, pp. 11-20, 2018.

[15] E. Ebeid, S. Martin и J. Jie, „A Survey on Open-Source Flight Control Platforms of Unmanned Aerial Vehicle," в *Euromicro Conference on Digital System Design (DSD),* Vienna, Austria, 2017.

[16] T. Jargalsaikhan, K. Lee, Y.-K. Jun и S. Lee, „Architectural Process for Flight Control Software of Unmanned," *Aerospace,* том IX, № 2, p. 62, 2022.