UDC 004.042

A. Akhaladze, I. Akhaladze

IMPROVING THE EFFICIENCY OF SOFTWARE DEVELOPMENT FOR UNMANNED SYSTEMS USING A SIMULATION ENVIRONMENT

Abstract: This article discusses tools and approaches to solving the problem of software development efficiency in terms of development cycle time. It is important to take into account the factors of preserving the physical integrity of unmanned systems by using a simulation environment for testing and debugging control algorithms.

The proposed tools and methods of using a simulation environment allow you to reduce the time spent developing and testing control algorithms for unmanned systems and reduce the risk of physical loss of the UAV by identifying and eliminating defects at the pre-flight testing stage.

Keywords: simulation, UAV, sensor system, firmware.

Introduction

The process of developing and debugging software for controlling unmanned systems has several major drawbacks:

1. Long development cycle time due to the need to conduct flight tests in each cycle of making changes to the control system code base
2. There is a risk of physical damage during testing due to defects in the control system.
3. Difficulties in maintaining the stability of the control system due to, for example, difficult to predict weather conditions
4. Undefined behavior when creating new systems and different flight modes

The traditional approach to developing software for UAV control modules requires flight testing on each cycle of changes to the code base, which not only becomes a large time investment for the project but also requires the availability of space for such testing and permission to use frequencies for transmitting the control signal, which is strictly limited by the military services in the languages of martial law and constant threats to the use of UAVs in the territories where the tests are conducted. Figure 1 shows a diagram of the development process by time distribution, taking into account the need to verify the operability of the created or modified UAV control algorithm, which demonstrates the strict dependence of the total development time of the UAV control system on time for testing on each cycle of changes to the code base.

© A. Akhaladze, I. Akhaladze

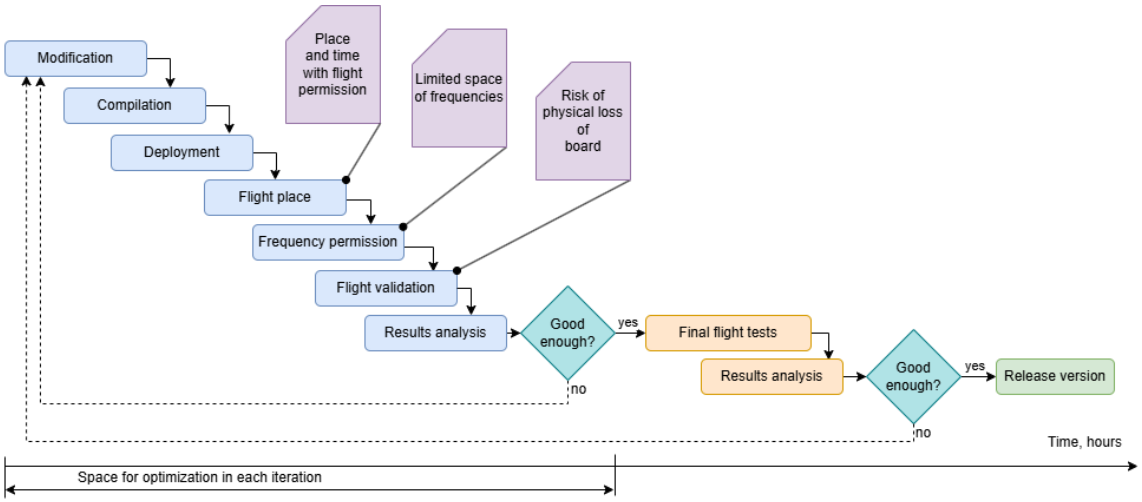

Figure 1. Diagram of the traditional UAV control system development process

Formulation of the problem

To propose tools and methods for reducing the overall development time of a control system for unmanned systems and reducing the risk of loss of sides by solving the following problem:

1. Using a simulation environment to test new changes to the control system with the ability to create scenarios and landscapes of any complexity and add complex weather conditions

2. Using software tools to simulate the flight behavior of aircraft, using new approaches to aerodynamics and new flight modes

Solving the tasks

Simulation as a flexible environment for flight testing

The decomposition of the UAV control system[1] allows us to identify components for replacement with simulation equivalents, which allows us to replace flight tests in the real world with the launch of a simulation scenario process with a flexible configuration of the landscape, scenario, and weather conditions. Figure 2 clearly shows the need to simulate a world where the drone performs a planned mission and receives information from the camera and position/acceleration sensors.

One of the most popular tools for simulating the flight environment is FlightGear. This is an open-source simulation tool that does not impose restrictions on the use for the development of proprietary UAV control systems and allows you to create flexible scenarios and landscapes for conducting flight tests of control algorithms. As a basic control and communication system, we will consider the open-source PX4 autopilot and QGroundControl (QGC), as a proxy controller and a tool for flashing and changing flight

 ISSN 1560-8956

controller parameters. According to the 2022 scientific conference held by the PX4 developer community, the direction of development of FlightGear and Unreal Engine integration was manifested, which allows you to create a simulation world of any complexity using variable weather conditions including wind, rain, fog, and snow. Figure 3 shows a diagram of a simulation environment for conducting flight tests of UAV control systems with the following component stack:

1. FlightGear + Unreal Engine: for building, and visualizing the external environment for flight[7]
2. QGC: as a proxy for firmware and changing flight controller parameters, as well as configuration of the type of UAV under test[8]
3. PX4: as a basic open-source flight controller, compatible with a wide range of control boards and UAV types[9]
4. Matlab: as a calculation engine and a tool for running a simulation scenario[10]

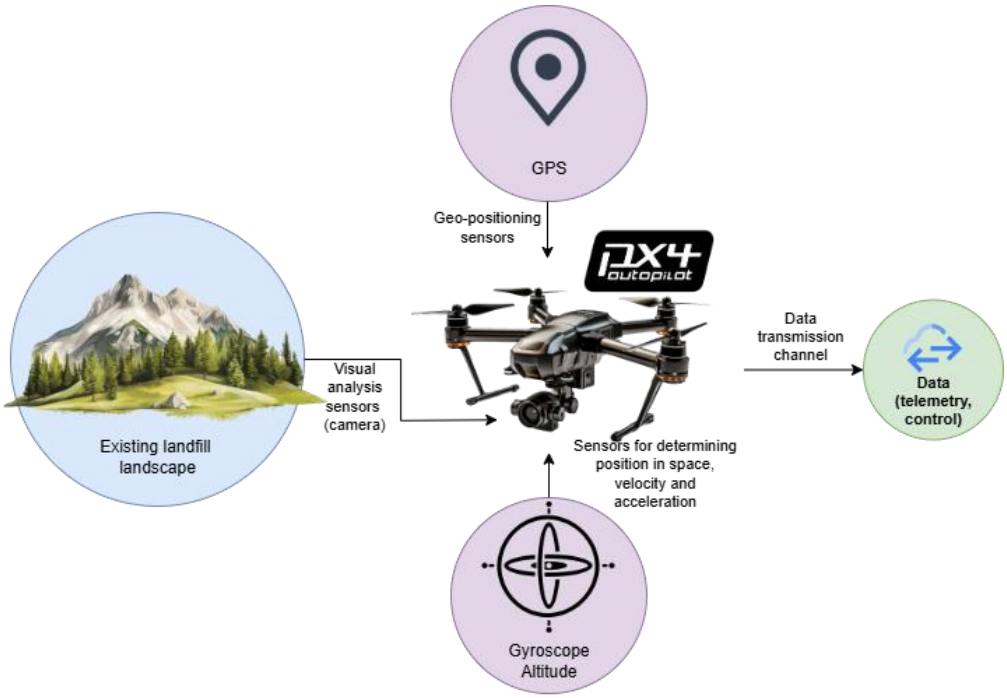

Figure 2. Decomposition of UAV sensor control systems

The study identified the following advantages and disadvantages of the above-mentioned tooling.

Advantages:

1. A flexible system for building a simulation world using the 3D engine Unreal Engine
2. Flexibility to change the logic of the PX4 flight controller by making changes to the code base

Disadvantages:

1. High cost of Matlab license
2. The need to make changes to the basic functionality of PX4 with a lack of modularity
3. The need to build models of fundamentally new aircraft designs

An alternative toolkit has been studied to avoid some of the shortcomings of the previously considered toolkit. An alternative to using Unreal Engine and FlightGear is to use the open-source Gazebo simulation environment, which effectively integrates with the QGC software ground control station and the PX4 autopilot. In addition, the proposed stack allows for changes to the flight controller control logic based on the modularity principle. That is, integration with the ROS operating system allows for changing or expanding the functionality of the flight controller by building separate modules in C++ or Python that can run on separate hardware platforms connected by a data bus that works according to the message broker scheme. One of the biggest advantages of the proposed stack is the price since the tools used are open source, including the Linux operating system.

Figure 3 shows a diagram of the use of simulation tools, which allows you to partially replace the data received by the drone flight controller while making it possible to check and debug the control logic taking into account the characteristics of flight scenarios, UAV types, and weather conditions. The choice of a simulation tool stack depends on the availability of licenses in the case of using proprietary ones and the knowledge of using the selected stack.

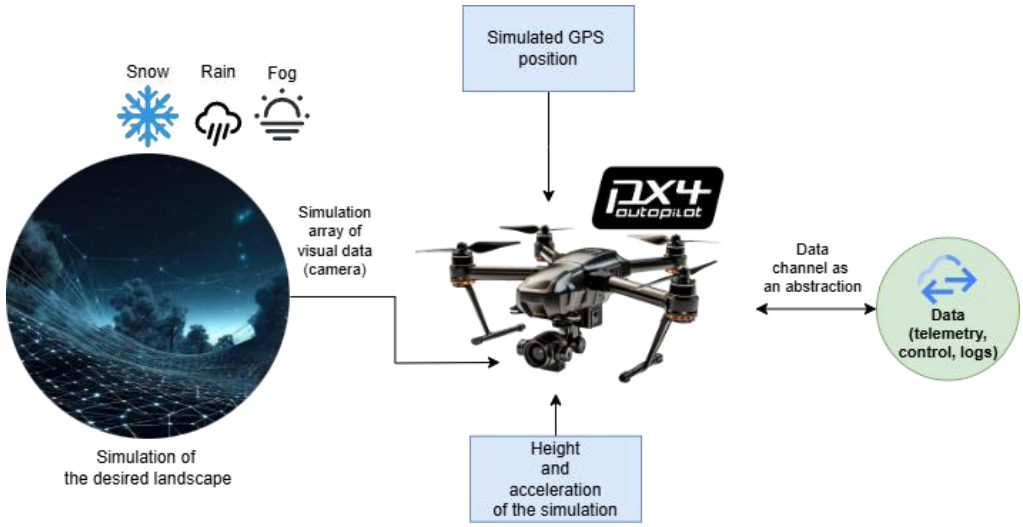

Figure 3. Decomposition of sensor systems using a simulation environment

Figure 4 shows a time-consuming diagram for developing UAV control module software using simulation tools. The diagram demonstrates the reduction in development

 ISSN 1560-8956

time at each iteration of control module software changes up to the final flight test stage in real-world conditions.

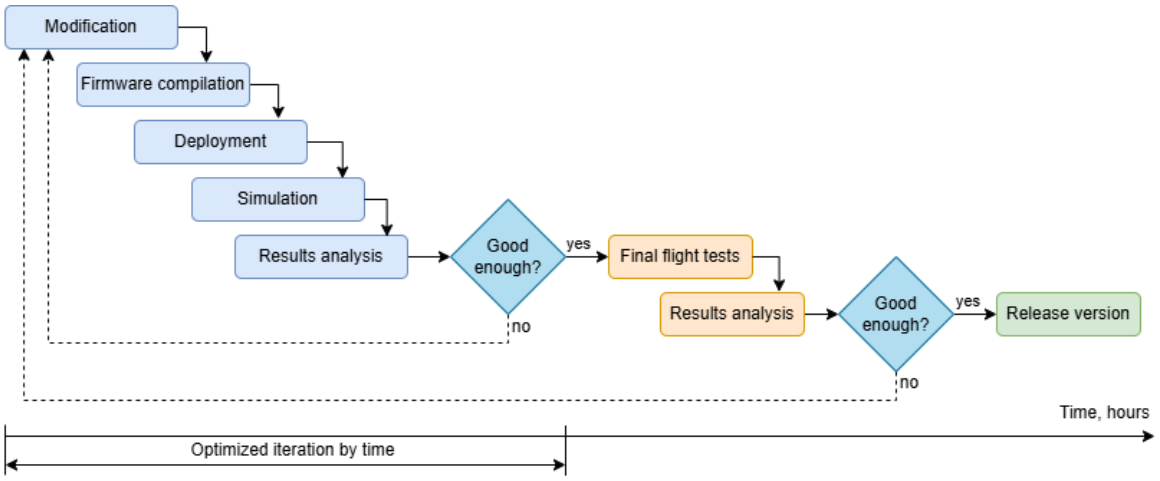

Figure 4. Time schedule for developing control algorithms

Analysis of results

The development and modification of UAV control system software are two of the main vectors of civil and military capabilities, and the creation of fundamentally new approaches in the design and use of drone aerodynamics requires changes in the logic of existing autopilot systems. However, the time required to develop and test the operability of autopilot algorithms is a critical indicator and negatively affects the timing of development and implementation of projects.

The proposed approach to increasing the efficiency of software development for unmanned systems using a simulation environment allows to avoid loss of time for obtaining flight frequency permits and time for searching for a test site for flight tests. Also, the use of a simulation environment allows to reduce the time for development and modification of the logic of existing autopilots at each iteration of making changes. An additional advantage of using this approach is the ability to conduct flight tests in different weather conditions, on the desired simulation landscape, and validate behavior under boundary conditions and different flight modes.

An important advantage is the ability to test new systems and flight modes with verification of the feasibility of their use and confirmation of the ability to perform the applied task without the likelihood of physical loss of the sides.

REFERENCES

1. Akhaladze A.E. Using IoT to synchronize flight trajectories of drones // Adaptive automatic control systems. - 2021. - Vol. 1, iss. 39. - P.20-26 - Available from: https://doi.org/10.20535/1560-8956.39.2021.247381

2. Akhaladze A.E. Synchronization of flight trajectories based on the "Internet of Things" architecture when implementing swarm control Adaptive automatic control systems. - 2022. - Vol. 1, iss. 40. - Available from: https://doi.org/10.20535/1560-8956.40.2022.261536

3. Liu, S., Zou, C., & Song, Y. (2019). Multi-objective optimization of UAV path planning based on genetic algorithm. Complexity. - 2019. P.1-15 Available from: https://www.researchgate.net/publication/322920720_Multi-objective_genetic_algorithm_for_civil_UAV_path_planning_using_3G_communication_networks

4. Han, Z., Li, Q., Feng, X., Liu, H., & Yu, H. (2020). A hybrid optimization method for drone path planning based on genetic algorithm and neural network. Complexity, 2020. - P.1-18 - Available from: https://www.researchgate.net/publication/305685753_A_Hybrid_Multi-Population_Genetic_Algorithm_for_UAV_Path_Planning

5. Akhaladze I.E. Increasing the efficiency of streaming video processing using serverless technologies // Adaptive automatic control systems. - 2021. - Vol. 1, iss. 39. P.32-40 - Available from: https://doi.org/10.20535/1560-8956.39.2021.247393

6. Akhaladze I.E. The use of serverless functions in the algorithm for calculating the target point of the trajectory under dynamic loading. - 2022. - Vol.1, iss. 40. P.34-38 - Available from: https://doi.org/10.20535/1560-8956.40.2022.261531

7. Using Unreal Engine Visualization for Airplane Flight - Available from: https://www.mathworks.com/help/aeroblks/using-simulation-3d-visualization-with-aerospace-blockset.html

8. QGrounfControl: Fly View. - Available from: https://docs.qgroundcontrol.com/master/en/qgc-user-guide/fly_view/fly_view.html

9. Open Source Autopilot for Drone - PX4 Autopilot - Available from: https://docs.px4.io/main/en/

10. PX4 Autopilot Support from UAV Toolbox - Hardware Support - Available from: https://www.mathworks.com/hardware-support/px4-autopilots.html