MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
NATIONAL AVIATION UNIVERSITY
Faculty of Aeronavigation, Electronics and Telecommunications
Chair of aviation computer-integrated complexes

ADMIT TO DEFENSE
Head of the graduating department
__________ Viktor SINEGLAZOV
“_____”_________________2024 y.

QUALIFICATION WORK
(EXPLANATORY NOTE)

OF THE GRADUATE OF THE EDUCATIONAL DEGREE
“BACHELOR”

Specialty 151 "Automation and computer-integrated technologies"

Educational and professional program "Computer-integrated technological processes and production"

Theme: The system of remote control of the vertical movement of the aircraft

Performer: student of FAET-323stn group Danylo Mykhailenko
Supervisor: candidate of technical sciences, associate professor Mykola TUPITSYN

Norm controller: ___________ Filyashkin M.K.
(sign)

Kyiv – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет аеронавігації, електроніки та телекомунікацій
Кафедра авіаційних комп'ютерно-інтегрованих комплексів

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
______ Віктор СИНЄГЛАЗОВ
“____” _____________2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)
ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
“БАКАЛАВР”

Спеціалність 151 "Автоматизація, та комп’ютерно-інтегровані технології"
Освітньо-професійна програма "Інформаційні технології та інженерія авіаційних комп’ютерних систем"

Тема: Система дистанційного керування вертикальним рухом літального апарата

Виконавець: студент групи КП-323 стн/Ба Михайленко Данило Ігорович
Керівник: кандидат технічних наук, доцент Тупіцин Микола Федорович

Нормоконтроллер: _____________ Філяшкін М.К.
(підпис)

Київ – 2024

NATIONAL AVIATION UNIVERSITY
Faculty of Aeronautics, Electronics and Telecommunications
Chair of aviation computer-integrated complexes
Educational and qualification level bachelor
Specialty 151 "Automation and computer-integrated technologies"

I APPROVE
Head of Department
______________Viktor SINEGLAZOV
«_____»______________2024 р.

TASK
for the performance of qualification work

MYKHAILENKO Danylo Ihorovycha

1. **The topic of the qualification work** "System of remote control of the vertical movement of the aircraft".
2. **Work completion date**: from April 20, 2024. until June 15, 2024
3. **Source data for the work**: Tupitsyn M.F., Koval O.V., Matiychyk D.M. The system of remote control of the movement of an unmanned aerial vehicle.
4. **The content of the explanatory note**: 1) Overview and analysis of UAV remote control systems; 2) Description of features of remote control of UAV vertical movement; 3) Setting the task; 4) Mathematical model of remote control of UAV movement in the vertical plane; 5) Development of an algorithm and program for remote control of UAV movement; 6) An example of calculating the movement of a UAV with remote control; 7) Analysis of the obtained results.
5. **List of mandatory graphic (illustrative) material**: Presentation in Microsoft PowerPoint.

6. Calendar plan-schedule

№ пор.	Task	Term implementation	Performance note
1	Acquaintance with the formulation of the task of the qualification work.	01.04.2024-04.04.2024	Done
2	Analysis of literary sources and Internet resources.	05.04.2024-24.04.2024	Done
3	Overview and analysis of UAV remote control systems.	25.04.2024-1.05.2024	Done
4	Description of features of remote control of UAV vertical movement.	2.05.2024-10.05.2024	Done
5	Setting objectives.	11.05.2024-25.05.2024	Done
6	Mathematical model of remote control of UAV movement in the vertical plane.	26.05.2024-02.06.2024	Done
7	Development of an algorithm and program for remote control of UAV movement.	03.06.2024-09.06.2024	Done
8	Designing an explanatory note, graphic materials and presentation for the diploma project.	10.06.2024-13.06.2024	Done
9	Submission of qualifying work for defense	15.06.2024	Done

7. Issue date of the assignment: “ 20 "_ April 2024 p.

Supervisor of the thesis ________________ Mykola TUPITSYN
(sign)

The task was accepted by________________Danylo MYKHAILENKO
(sign)

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій

Кафедра авіаційних комп'ютерно-інтегрованих комплексів

Освітньо-кваліфікаційний рівень бакалавр

Спеціальність151 «Автоматизація та комп'ютерно-інтегровані технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____________ Віктор СИНЄГЛАЗОВ
«_____»________________2024 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

МИХАЙЛЕНКО Данило Ігоровича

1. Тема кваліфікаційної роботи «Система дистанційного керування вертикальним рухом літального апарата».

2.**Термін виконання роботи:** з 20 квітня 2024р. по 15 червня 2024 р.

3. **Вихідні дані до роботи**: Тупіцин М.Ф., Коваль О.В., Матійчик Д.М. Система дистанційного управління рухом безпілотного літального апарата//Патент №81370 Україна, МКИ B64C 13/00. – № u201300897; Заявл. 25.01.13; Опубл. 25.06.13;

4.**Зміст пояснювальної записки**: 1) Огляд та аналіз систем дистанційного керування рухом БПЛА; 2) Опис особливостей дистанційного керування вертикальним рухом БПЛА; 3) Постановка завдання; 4) Математична модель дистанційного керування руху БПЛА у вертикальній площині; 5) Розробка алгоритму та програми дистанційного керування руху БПЛА; 6) Приклад розрахунку руху БПЛА з дистанційним керуванням; 7) Аналіз отриманих результатів.

5. **Перелік обов'язкового графічного (ілюстративного) матеріалу:** Презентація в MicrosoftPowerPoint.

6. **Календарний план-графік**

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Ознайомлення з постановкою задачі кваліфікаційної роботи.	01.04.2024-04.04.2024	Виконано
2	Аналіз літературних джерел та інтернет ресурсів.	05.04.2024-24.04.2024	Виконано
3	Огляд та аналіз систем дистанційного керування рухом БПЛА.	25.04.2024-01.05.2024	Виконано
4	Опис особливостей дистанційного керування вертикальним рухом БПЛА.	02.05.2024-10.05.2024	Виконано
5	Постановка завдання.	11.05.2024-25.05.2024	Виконано
6	Математична модель дистанційного керування руху БПЛА у вертикальній площині.	26.05.2024-02.06.2024	Виконано
7	Розробка алгоритму та програми дистанційного керування руху БПЛА.	03.06.2024-09.06.2024	Виконано
8	Оформлення пояснювальної записки, графічних матеріалів та презентації до дипломного проекту.	10.06.2024-13.06.2024	Виконано
9	Подання кваліфікаційної роботи до захисту	15.06.2024	Виконано

7. Дата видачі завдання: “ 20 "_квітня 2024 р.

Керівник дипломної роботи ______________ ТУПІЦИН М.Ф.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання ______________ МИХАЙЛЕНКО Д.І.
(підпис) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Система дистанційного керування вертикальним рухом літального апарата». 67 с., 26 рис., 8 джерела.

Об’єкт дослідження – система дистанційного керування вертикальним рухом літального апарата.

Предмет дослідження - багатоцільовий БПЛА(квадрокоптер).

Мета кваліфікаційної роботи - розробка програмного коду дистанційного керування вертикальним рухом БПЛА, та розрахунку .

Завдання кваліфікаційного проекту – розробка системи дистанційного керування рухом БПЛА, з врахуванням розрахунків льотно-технічних характеристик.

Теоретичні дослідження складалися з аналізу існуючих систем, розробки алгоритму системи дистанційного керування, розробки програмного коду для дистанційного керування літального апарату в вертикальній площині.

Результати кваліфікаційної роботи рекомендуються для з ознайомлення з дистанційним керуванням у вертикальній площині, програмним кодом дистанційного керування літального апарату, математичною моделлю БПЛА, кінемаматичною моделлю. Для більш точних розрахунків рекомендується додати в систему динамічну математичну модель, для розрахунків динамічного руху.

ABSTRACT

Explanatory note to the qualification work "System of remote control of the vertical movement of the aircraft". 67 pp., 26 pictures, 8 sources.

The object of research is a system of remote control of the vertical movement of an aircraft.

The subject of the study is a multi-purpose UAV (quadrocopter).

The purpose of the qualification work is to develop the software code for the remote control of the vertical movement of the UAV, and the calculation.

The task of the qualification project is the development of a remote control system for the movement of UAVs, taking into account calculations of flight characteristics.

Theoretical studies consisted of analysis of existing systems, development of a remote control system algorithm, development of software code for remote control of an aircraft in the vertical plane.

The results of the qualification work are recommended for familiarization with remote control in the vertical plane, the program code of the remote control of the aircraft, the mathematical model of the UAV, the kinematic model. For more accurate calculations, it is recommended to add a dynamic mathematical model to the system for dynamic motion calculations.

Content

Introduction

In the qualifying work, the system of remote control of unmanned aerial vehicles (UAVs), in particular quadcopters, was considered. In today's world, these technologies are becoming increasingly relevant, and their development is determined by several key factors that emphasize their importance and potential.

Safety: Remote control systems allow operators to control the UAV from a safe distance, which minimizes the risk to human life. This is especially important in dangerous or hard-to-reach places.

Flexibility and mobility: UAVs can be quickly deployed and redirected to new tasks without the need for an operator to be physically present on site. This is critical for tasks that require quick response, such as emergencies.

Cost reduction: Using UAVs instead of traditional methods such as helicopters or manned aircraft significantly reduces operating costs. This makes UAV technologies cost-effective for various industries.

Technological development: Constant advances in technology, such as improved communication, increased autonomy and the introduction of artificial intelligence, make remote control systems increasingly powerful and versatile. This opens up new opportunities for the use of UAVs in various fields, from the agricultural sector to logistics and defense.

Multi-functionality: Modern remote control systems can integrate with various sensors and cameras, allowing for real-time data collection and analysis.

Various patents have been reviewed that demonstrate different approaches to remote control of UAVs, focusing on reliability, safety and efficiency. The implementation of automatic control systems and backup flight plans ensures a high level of reliability and autonomy.

The mathematical model of the movement of the quadcopter was also examined in detail, which is an important basis for understanding the movement of this UAV and developing control algorithms. However, the model has some drawbacks, such as the simplification of many aspects and the need for numerical methods to solve differential equations.

The program code for controlling the movement of the quadcopter in the vertical and horizontal planes, implemented in Unity 2D, was developed. Using Unity provides a user-friendly interface and powerful tools for development and testing. However, here too there are some limitations that must be taken into account for further improvement of the model. However, it is necessary to continue to improve the existing models, taking into account additional factors, to achieve more realistic and accurate results. In general, the presented studies demonstrate the importance and relevance of remote control systems for quadcopters.

CHAPTER 1

UAV REMOTE CONTROL SYSTEM

1.1 General overview.

To consider the general provisions of the UAV remote control system, we will take already existing patents of inventions.

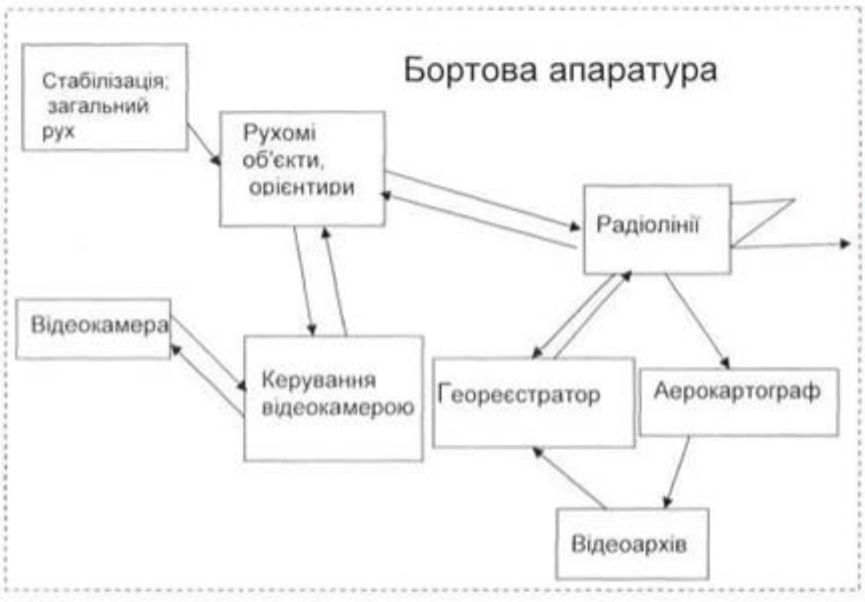
The system of remote control of the movement of unmanned aerial vehicles. [1]

The UAV (unmanned aerial vehicle) remote control system consists of: an automatic control system, a video camera, a video camera control device, a geo-recorder, an aerial cartographer, a video archive on board the aircraft, a radio line and ground equipment with a radio line, equipment for forming UAV control commands, an external pilot’s display and video archive of the external pilot. The system includes a unit for switching the control of the UAV from the automatic control system to the external pilot and vice versa, and the video camera control device consists of a servo drive that compensates for the angle of inclination of the video camera along the roll.

The model refers to aviation equipment and can be used for remote control systems of an unmanned aerial vehicle (UAV)

The aircraft control system is a complex [1] containing an automatic control system (ACS) with an on-board computer, a video camera (VC) with a monitor and a pilot's control panel on board the aircraft (LA). This complex allows the pilot, depending on the information from the VC or in the event of a malfunction of the ACS, to intervene in the operation of the automatic piloting system.

In this control system, if there is a pilot on board the aircraft, there is no remote control system for the movement of the aircraft. The payload of the aircraft is significantly reduced, the pilot and his life support systems occupy a certain volume in the aircraft, which can be used more efficiently in their absence.



[1]

Fig 1.1

Remote UAV traffic control system [2], which includes self-propelled guns located on the UAV, VC, VC control device, geo-recorder, aerial cartographer, video archive, radio line and ground equipment with radio line, equipment for forming UAV control commands, operator's display and video archive. Disadvantages of this remote control system include the following: there is no roll angle compensator in this system; the UAV control command coming from the operator can be adjusted against his will by the control commands from the autopilot, which will negatively affect the efficiency of its execution.

[2]

Fig. 1.2

The task is solved by the fact that in the remote control system of the unmanned aerial vehicle, which consists of an ACS, a BC, a VC control device, a geo-recorder, an aerial cartographer, a video archive on board the UAV, a radio line and ground equipment with a radio line, equipment for forming UAV control commands, an external pilot's display and video archive of the external pilot, according to the useful model, the unit for switching the control of the UAV from the ACS to the external pilot in the reverse direction is included, and the VC control device consists of a servo drive that compensates for the angle of inclination of the VC along the roll.

An example of the implementation of a remote control system for the movement of an unmanned aerial vehicle is explained: the drawing is a diagram of a remote control system for the movement of a UAV. The proposed UAV remote control system (drawing) consists of the UAV onboard equipment and the UAV ground control point (GNC). Radio line 1 transmits a video signal from the UAV. The on-board ACS also includes a geo-recorder, an aerial cartographer and a video archive. Radio line 2 transmits control commands from the I PIC to the UAV board. The essence of the proposed useful model is that during remote control, the UAV at the command of the external pilot, with the help of the switching device "1-2", is disconnected from the ACS and the external pilot through the switching device located in the "1" position, and the servo drives control the trajectory of the UAV, and the VC, with the help of which the external pilot controls the UAV, due to the servo drive that compensates for the angle of inclination of the VC along the roll, does not change its position along the roll angle, which significantly improves the quality of piloting the UAV. When the switching device is in the "2" position, the UAV is controlled by the ACS.

The system of remote control of the movement of an unmanned aircraft, which consists of an automatic control system, a video camera, a video camera control device, a geo-recorder, an aerial cartographer, a video archive on board an aircraft, a radio line and ground equipment with a radio line, equipment for forming commands for controlling an unmanned aircraft, an external pilot’s display and video archive of the external pilot, which differs in that the system includes a unit for switching the control of the unmanned aircraft from the automatic control system to the external pilot and vice versa, and the video camera control device consists of a servo drive that compensates for the tilt angle of the video camera along the roll.

The ability to remotely control an unmanned aerial vehicle [2]

The system refers to a remote control method for controlling an unmanned aerial vehicle (also called an unmanned aerial vehicle) and an unmanned aerial vehicle for implementing a remote control method.

Unmanned aerial vehicle (UAV) missions are essentially based on pre-programmed flight paths. Planning flight paths requires accurate knowledge of the flight area and the expected air defense threat. In particular, very accurate knowledge of the target area is required for target engagement. This is often aided by end-of-approach sensors that work by comparing or matching images. Thus, these sensors require accurate images of targets, and may require images from different viewpoints under certain circumstances. Pre-programmed flights cannot react to unforeseen events.

The performance of such complex tasks can be significantly improved if the UAV is controlled remotely over a radio wave transmission line, with the "pilot" controlling the UAV from a remote control station. The control station can be located both on the ground and in flight on a piloted aircraft. Remote control is carried out using a TV or video image. The image is recorded by the camera of the unmanned UAV and transmitted to the control station, where it is presented in the appropriate form. The aforementioned traditional method has the disadvantage that it depends on the reliability of the radio wave communication line. The loss of radio communication is known to lead the UAV into an uncontrolled phase of flight, which will eventually lead to its crash.

The task of the system is to prevent the occurrence of an uncontrolled flight of a remotely controlled UAV in case of loss of radio communication.

The above-mentioned tasks are solved in the method of remote control of an unmanned aerial vehicle (UAV) from a remote control station by radio transmission of control signals from the control station to the UAV and radio transmission of the image from the UAV to the control panel. station. According to the system, the UAV first flies along a pre-programmed safe route, and then, if necessary, the UAV is guided by remote control on the flight path by appropriate radio transmission of appropriate control signals from the control station to the UAV. In case of interruption of the radio communication between the control station and the UAV, the UAV will fly an alternate route calculated on board the

UAV using the on-board equipment of the UAV, without active remote intervention by the controller. station.

It is based on the method of continuing the flight of the UAV in case of loss of radio communication using the backup flight program calculated on board. Thus, the threat of losing the UAV can be avoided: and the mission can be successfully completed despite radio interference.

The figure (Fig.1.3) shows partial views of the UAV mission scenario with the image of the remotely controlled flight path 2, and the programmed safe route 3, and shadow zones 4 and 5 (that is, areas in which control signals transmitted by radio remote control station 6 will interrupted) as a result of exemplary terrain formations such as mountains or hills 7 and the position of the control station 6 in the depicted terrain.

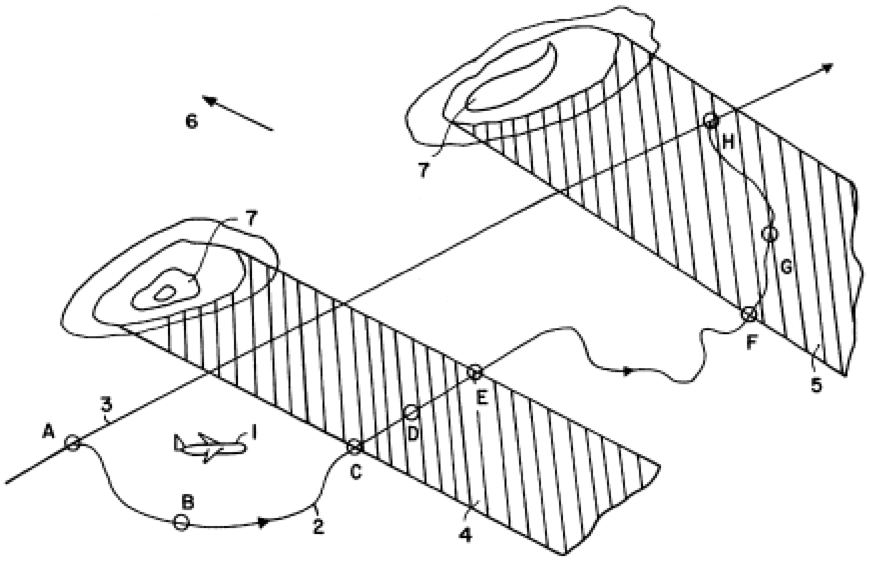


Fig 1.3

The development of the method is marked by characteristic points from A to H along the remotely controlled flight path 2. The steps of the method, assigned to individual points, are given below:

Initiation of remotely controlled flight route 2 by interfering with safe route 3.

B remotely controlled flight path 2 is controlled by continuous extrapolation of the instantaneous flight path.

entry of the UAV 1 into the shadow zone 4, thereby causing the start of an approaching uncontrolled flight, with further recognition and analysis of remote control interruption.

D continuation of the flight of the UAV along the backup flight path in accordance with the backup flight program formed on board the UAV 1.

E transition back to remote control of UAV 1 when exiting the shadow zone

F termination of remote control of UAV 1 when entering another shadow zone

G guiding the UAV 1 back to the safe route 3 according to the backup flight program formed on board the UAV 1.

H continued flight safe route 3.

For example, UAV 1 first automatically follows a pre-programmed flight course to a safe route 3. The "pilot" at the control station 6 intervenes, if necessary, in this pre-programmed course. As a result, the UAV deviates from the set course and continues to fly along the remotely controlled flight path 2 according to the control signals transmitted or commands from the remote control station 6. At this stage, the flight depends on the critical reliability of radio communication. If the radio communication between the control station 6 and the UAV 1 fails due to shadowing or interference, the invention provides that the flight in danger of becoming uncontrollable continues with a backup flight program generated on board after the interruption has been recognized and the type is scattered.

The backup flight program takes the UAV through the danger zone (that is, the shadowing or obstruction zone) until it enters the zone where there is radio communication again and it becomes possible to guide the aircraft by remote control.

If such radio communication recovery zone cannot be reached, an automatic return flight to the specified runway is performed after recalculation on board the corresponding flight program.

In a further variant, it is provided that the UAV returns to the pre-planned course and approaches the alternatively planned target, performs the necessary maneuvers of the pilot through remote intervention in the software control, and then returns to the runway after the pre-planned course.

The method and operation of the UAV described above require certain equipment and/or functional processes, which are described below.

UAV 1 is equipped with a traditional programmable flight guidance system, which allows the UAV to autonomously approach one or more target areas in flight and return to a predetermined runway. According to the invention, the automatic flight guidance system is equipped with an interface for control using radio control. When pointing commands are received, this interface effectively disables software control and enables remote control. As a result, the UAV performs movements at the command of the control station 6.

The automatic flight guidance system actually has a conditional database of terrain data, with the help of which the UAV orients itself during automatic flight over the terrain. During remote control flight, the accompanying route planner uses data from this databank to continuously or periodically monitor the aircraft's guidance commands. The route planner does this by extrapolating the instantaneous flight path based on the guidance commands and comparing it to data in the terrain databank.

Detected dangerous situations are immediately reported to the control station and displayed there, or a collision prevention program is calculated on board and corresponding corrective maneuvers are automatically performed.

In case of loss of radio communication, recognition is provided, distinguishing between "loss due to tracking" and "loss due to interference". The recognition sensor is the radio itself, which distinguishes between "no input signal" and "input signal, but unread modulation", which are attributed to shadowing and to interference, respectively.

In the event of loss of radio communication due to shadowing of the control station, the shadowing altitude zones are calculated using on-board equipment based on the accompanying terrain profile and the x/y/z coordinate values of the control station 6. Based on the data calculated from the shadow zone, the optimal flight path on the terrain is further calculated for flight through the shadow zone and is supplemented by the calculation of the transition point (for example, point E) to return to the control panel at the planned exit point from the shadow zone (for example, shadow zone 4).

In case of loss of radio communication due to "obstacles" of the receiver, the source of the interference is located on the terrain (x/y coordinate values) with the help of on-board equipment. The value of the Z coordinate is determined by the known terrain profile. Obstacle zone near depends on height, calculated taking into account location data. In the case of a known obstacle zone, the optimal flight path through the obstacle zone is calculated, supplemented by the transition point of returning to the control panel at the end of the obstacle zone.

In the event that the transition point for returning to remote control is not found or the continuation of the remotely controlled flight is no longer of interest, the on-board equipment will calculate the return route of UAV 1 to its original position. If no further target area is planned, the mission is aborted and a route is planned to the pre-planned return flight course. Alternatively, you can plan a route using a new return course if it is more profitable than the originally planned course.

Alternate route planning may also be performed during the unobstructed remote control phase as a preventive measure, either continuously or at intervals to have a calculated alternate flight program readily available in the event of an emergency.

Management of flight and technical control of an unmanned aerial vehicle. [3]

There are two unmanned aerial vehicle (UAV) architectures and a mission plan execution method. One UAV architecture includes a Flight and Mission Management (FCME) component that makes strategic decisions, a Flight Technical Control Manager (FTCM) that makes tactical decisions, and a Vehicle Management System (VMS) that provides navigational support. FCME and FTCM run on the same processor, while VMS runs on a separate processor. The second architecture includes redundant processors to run FCME and FTCM, as well as redundant processors to run VMS. The UAV executes a mission plan, which may include flight plan(s), communications plan(s), weapons plan(s), sensor plan(s), and/or contingency flight plan(s). The UAV can also control various optical sensors, training sensors and lighting.

This system belongs to the field of software architecture. More specifically, this system refers to a modular software architecture organized by layers and segments that can be applied to systems that contain multiple domains or functions, such as unmanned aerial vehicles (UAVs).

The first option: The aircraft includes a processing unit, data storage and machine language instructions. Machine language instructions are stored in data storage and can be executed by the processor to perform functions. Functions include: (a) obtaining a mission plan, which includes a flight plan; (b) verification of the mission plan; (d) determining the current location of the aircraft is outside the flight plan, and (e) prompt execution of the conditional flight plan.

Second option: The aircraft includes a first standby processing unit, a first data store and first machine language instructions. The first standby processor includes a first processor and a second processor. The first machine language instructions are stored in the first data store and may be executed by at least one processor of the first standby processor to perform the first functions. The first functions include: (a) obtaining a mission plan that includes a flight plan and a contingency plan, (b) verifying the mission plan, (c) executing the mission plan based on the flight plan, and (d) synchronizing the first processor and the second processor.

Third option: The mission plan is accepted on the aircraft. A mission plan includes a flight plan, a sensor plan, a communications plan, a contingency flight plan, and a weapons plan. The mission plan is recognized as valid. In response, a message is sent that the mission plan is valid. The mission plan is downloaded to one or more processors of the aircraft. At least part of the mission plan has been completed.

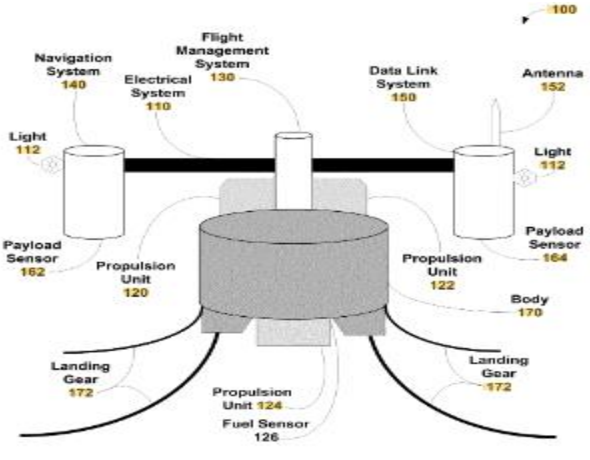


Fig.1.4. An example of a UAV is shown, according to the options.

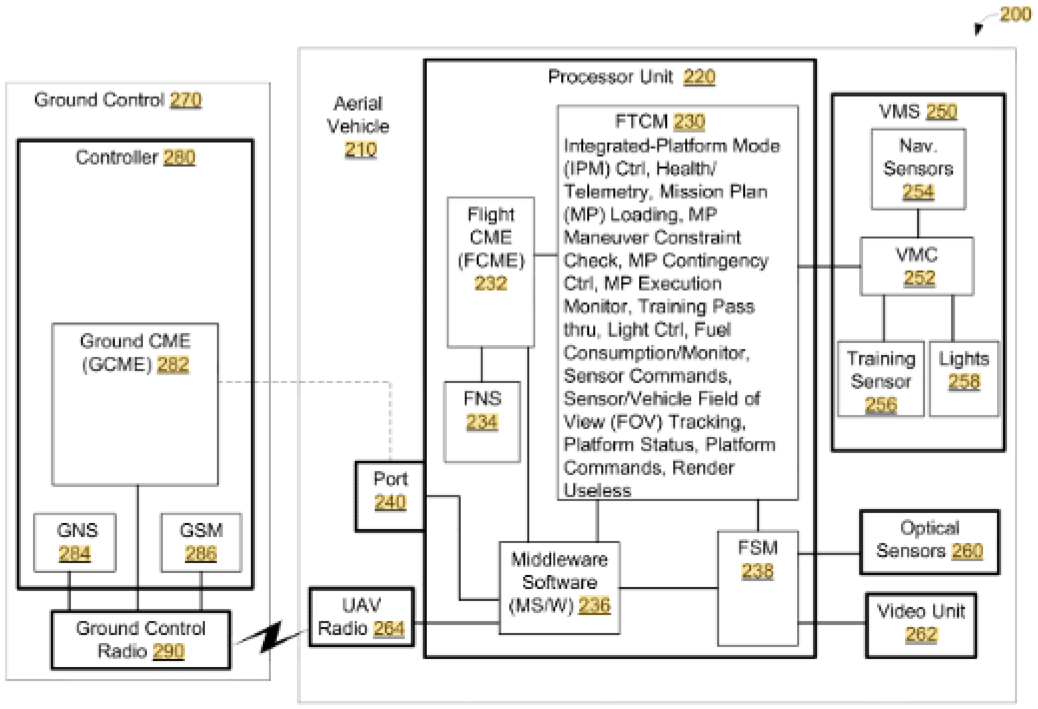


Fig.1.5. An example of an aircraft control system is shown.

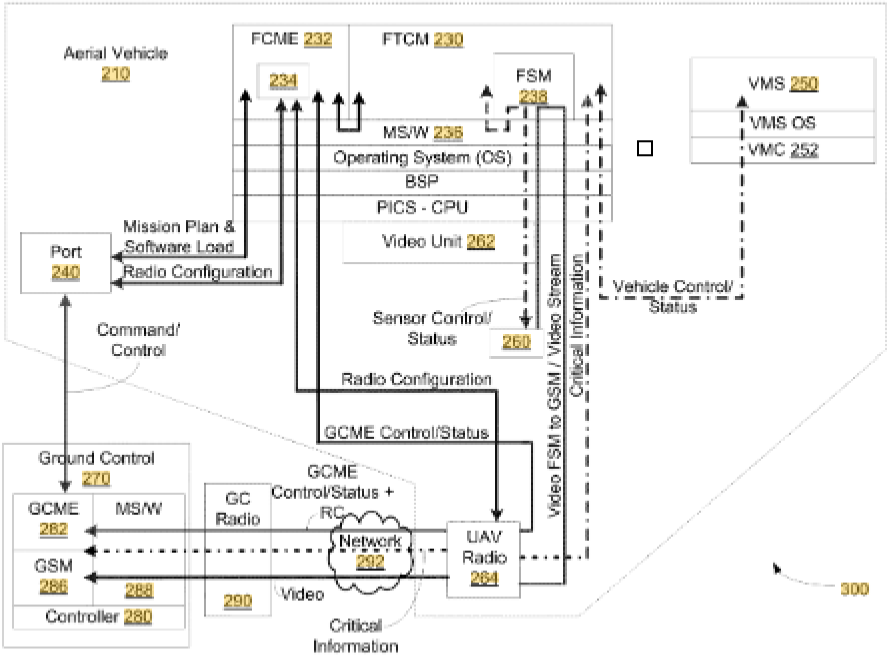


Fig.1.6. Shows an exemplary architecture of an exemplary aircraft control system.

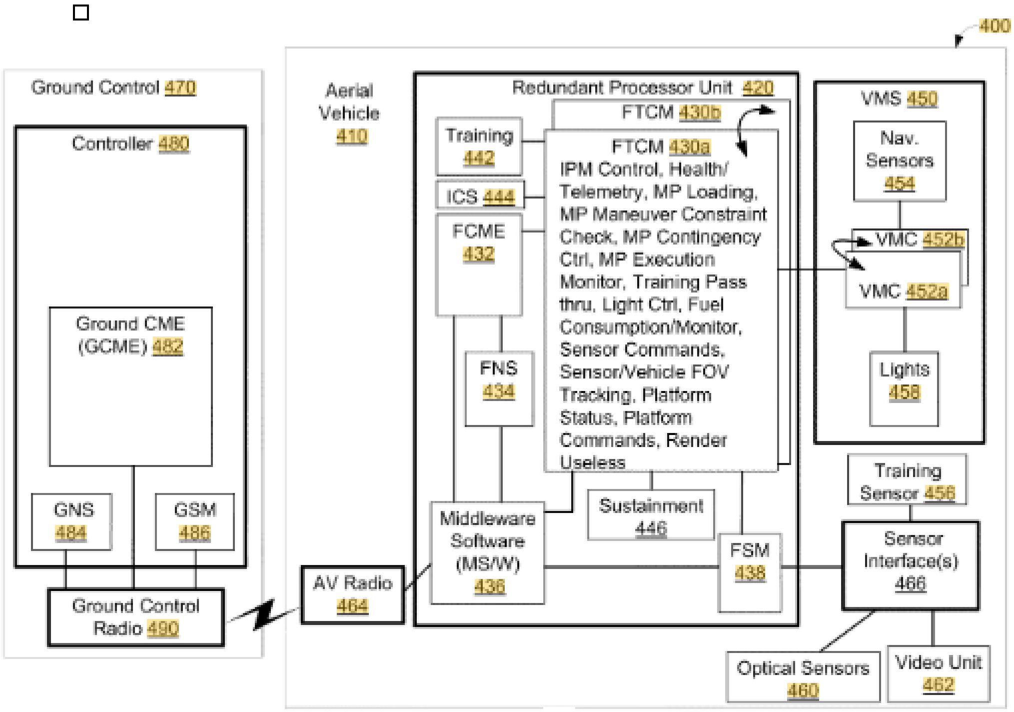


Fig.1.7. An example of an aircraft control system with spare components is shown.

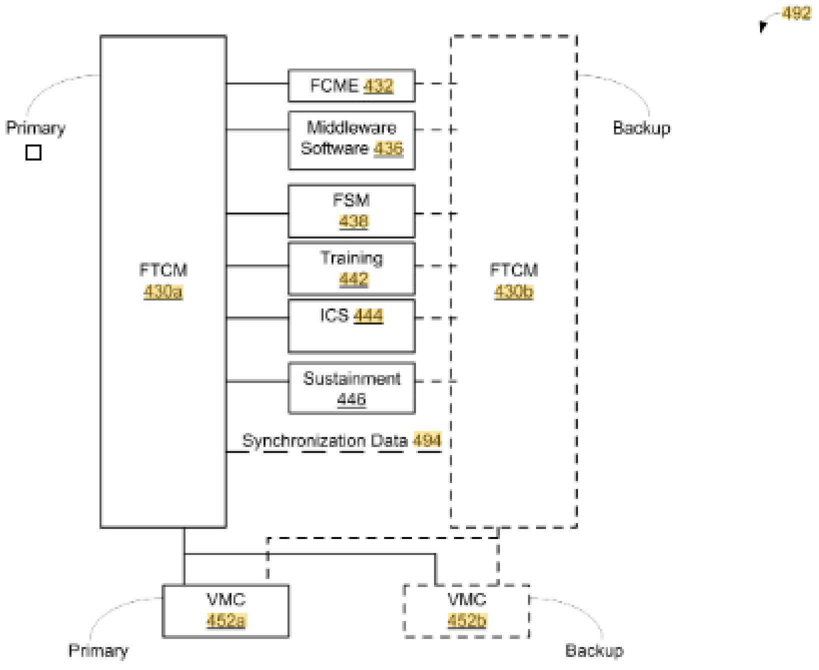


Fig.1.8. Shows a sample block diagram of redundant FTCM and VMC

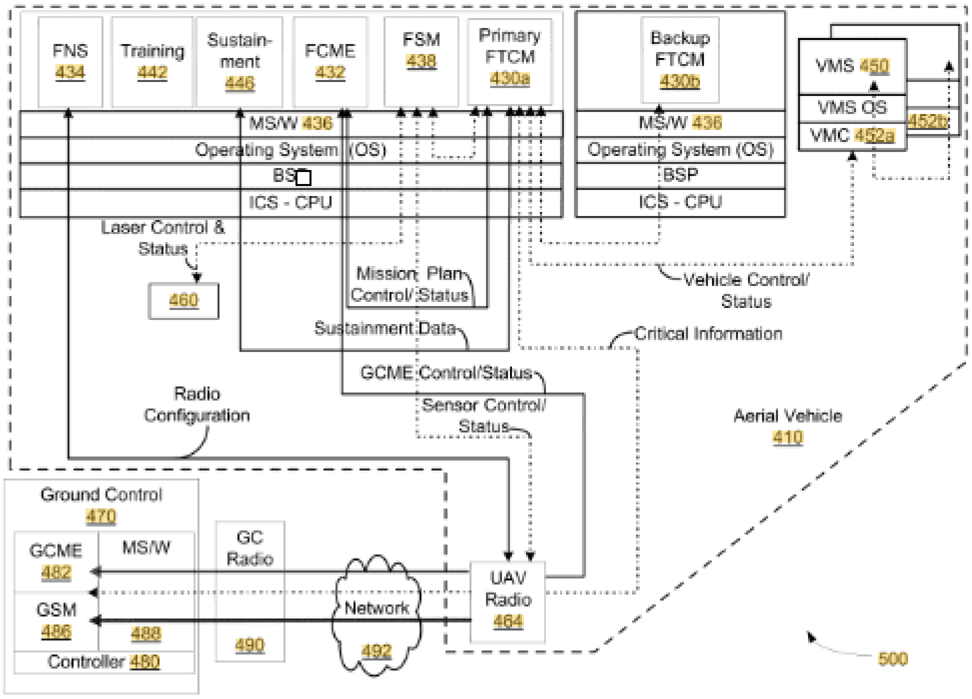


Fig.1.9. Shows an example of an aircraft control system architecture with redundant components.

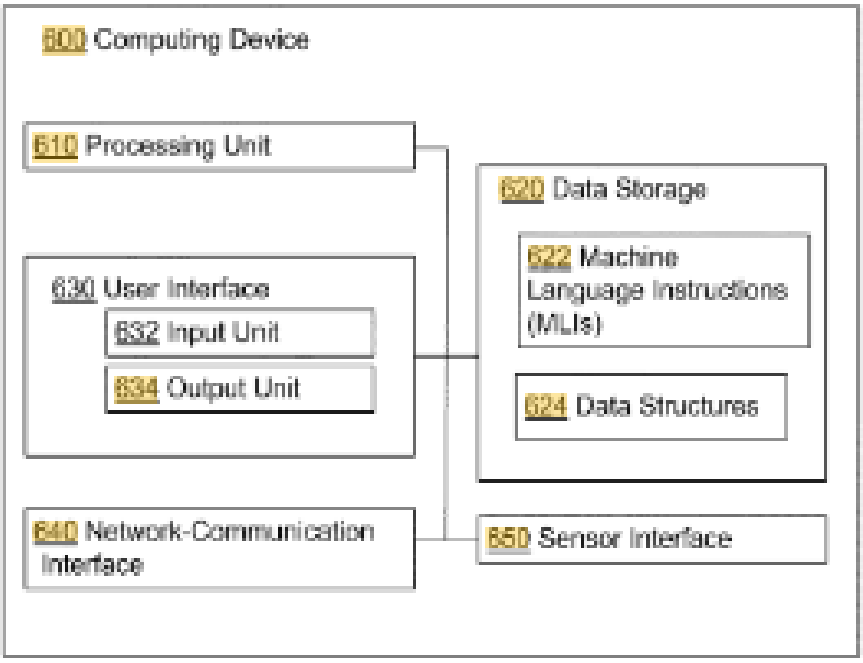


Fig.1.10. It is a block diagram of an exemplary computing device.

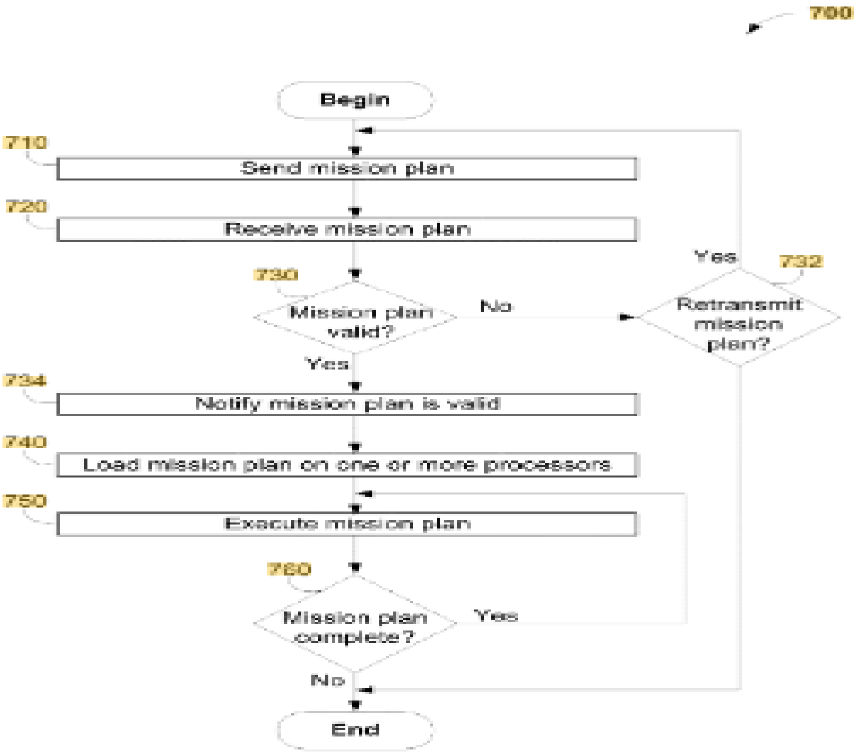


Fig.1.11. Is a flowchart depicting an exemplary way to execute a mission plan.

The system includes methods and a device for controlling a UAV. A UAV can provide a networked situational awareness capability for a military unit such as a platoon or division. The UAV may guard areas and/or provide reconnaissance, surveillance and target acquisition (RSTA).

The control of the UAV can be carried out both on board the aircraft and on the ground. UAV management may include a strategic or command and mission (CME) component and a tactical or technical management manager (TCM) component. The CME component provides high-level control and can be shared between the ground-based CME (GCME) located at the ground control point and the on-board or "flight" CME (FCME) located on the UAV. The TCM component may contain both ground and flight components or may contain only the flight TCM component (FTCM).

FTCM can provide tactical control over UAVs, while FCME and GCME can provide strategic control. Thus, the GCME is the primary controller of the aircraft; it provides mission control (communications, sensors and mission plan control) to the UAV via FCME and/or FTCM commands. The FTCM can process commands directly or relay commands to other vehicle components. FTCM has the ability to support teams on critical and non-critical paths to ensure message priority and security.

The UAV can be equipped with sensors and cameras that transmit data to ground control in real time. In addition, the UAV can be equipped with weapons such as missiles or bombs. Ground control can remotely launch UAV weapons in real time. The UAV can be configured to be carried by one or more platforms (such as vehicles) or by people (such as soldiers or police officers). The UAV can carry one or more people in a hurry in a protective transport container, which protects the system from the effects of natural and man-made influences.

FTCM can provide various services and interfaces inside and outside the UAV, such as mission planning services, sensor management, and configuration utilities. The FTCM may allow the download of mission plans (flight plans, contingency flight plans, communication plans, speed commands and/or sensor plans) from ground control and verify that the downloaded mission plan is valid. For example, a mission plan may include one or more flight plans, each of which contains a list of one or more waypoints, timing requirements, sensor plans, and/or weapon plans. Waypoints and timing requirements can indicate where and when, respectively, the UAV must be to complete the task set by the mission plan. A sensor plan, communications plan, or weapons plan may specify how the sensors, communications equipment, or weapons on board the UAV, respectively, are to be deployed during the mission.

Software developed for use on board UAVs (FTCM, FCME, and VMS software components) and/or for use by ground control can be designed according to a segmented and multi-tiered architecture that provides the foundation for well-designed software. this promotes loose coupling between software objects. Examples of software objects are software objects, modules, functions, routines, code, computer data, data objects, databases, and/or data structures.

General drone control system. [4]

The system is individual to the drone control system, and more specifically, to the method of remote piloting of the aircraft. A drone is an unmanned aerial vehicle controlled by remote control. Drones are used for many military purposes as aerial targets and for other purposes such as reconnaissance.

This system is a remotely piloted aircraft or unmanned aerial vehicle that was originally designed to be an operational aircraft and to accommodate a pilot if necessary. In the related field of drone design and development, there are usually two different approaches.

The first approach, similar to this system, originated in the transformation of an existing aircraft into a drone. Converting traditional aircraft into drones required significant modifications to existing aircraft. you need the development of additional and unique hardware and software complexes both on the aircraft and on the ground station. In addition, many drone aircraft conversions also require the development of new or modified aircraft control laws. These control laws or algorithms are part of additional hardware/software systems on board the drone and are used to control the drone's operations. Primarily because of the aforementioned modifications, converting aircraft into drones often becomes expensive and technically difficult. For this reason, the conversion of each aircraft into a drone is actually a unique development. There were no universal drone conversion kits that could simply be used on multiple types of aircraft. Similarly, once an aircraft has been converted to a drone using traditional methods, converting the drone back to a serviceable aircraft becomes impractical and expensive because the modifications were quite extensive. That is, there are no known removable and portable aircraft-to-drone conversion packages that would simplify the conversion process.

A second approach to drone design and development originated in designing and manufacturing a drone from scratch. Therefore, these vehicles are not designed for the pilot to get motion sickness on board. These vehicles are commonly known as remotely piloted vehicles (RPVs) and unmanned aerial vehicles (UAVs). Both UAVs and UAVs are vehicles originally designed without a pilot on board.

A typical drone system consists of an aircraft or on-board system combined with a ground station. The ground station is adapted to ensure reliable control of the drone during the flight of the aircraft. In addition, the corresponding electrical or mechanical devices are also controlled through the ground station. These electrical/mechanical devices are used to actuate elements such as landing gear, flaps, slats, wheel brakes, brake brakes, nose wheel control, and many other electrical connections used to control an air vehicle.

The drone control system essentially consists of a ground station and an aircraft. An air vehicle in a disclosed drone control system is a functional aircraft. The ground station in this invention includes a replica of the cockpit and the aircraft with identical controls that are on the operational aircraft.

Control movements at the ground station are transferred by the operator from analog movements into digital signals and transmitted by the telemetry drone. Digital telemetry signals are used by simple mechanical actuators in the control bodies to accurately reproduce the output movement of the control bodies at the ground station. Thus, the response of the drone to the volumes of processing is exactly the same as that of the aircraft to the drone. The only difference is the replacement of the pilot with a set of mechanical drives. Information about the condition and characteristics of the aircraft or aircraft will be transmitted telemetrically using video communication, and not telemetric readings of the device itself.

Another goal of the system is to create a drone control system that transmits information about the position and characteristics of the aircraft from the drone to the ground station using telemetry video signals. Further, separate visual images of takeoff and landing will also be transmitted over a secondary video telemetry link.

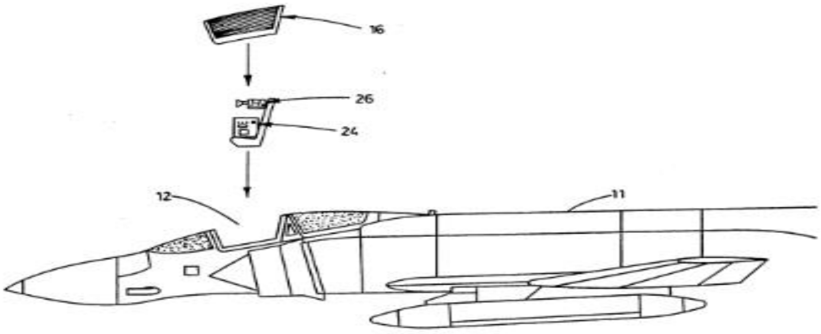


Fig.1.12. shows the configuration diagram of the aircraft, including the darkened lantern.

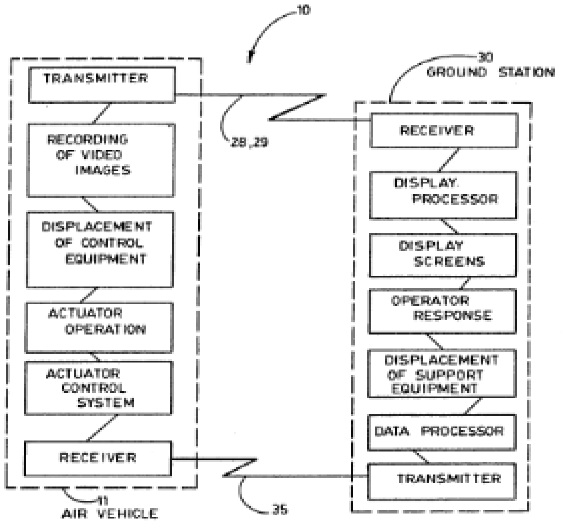


Fig.1.13. shows the information flow between the aircraft and the ground station.

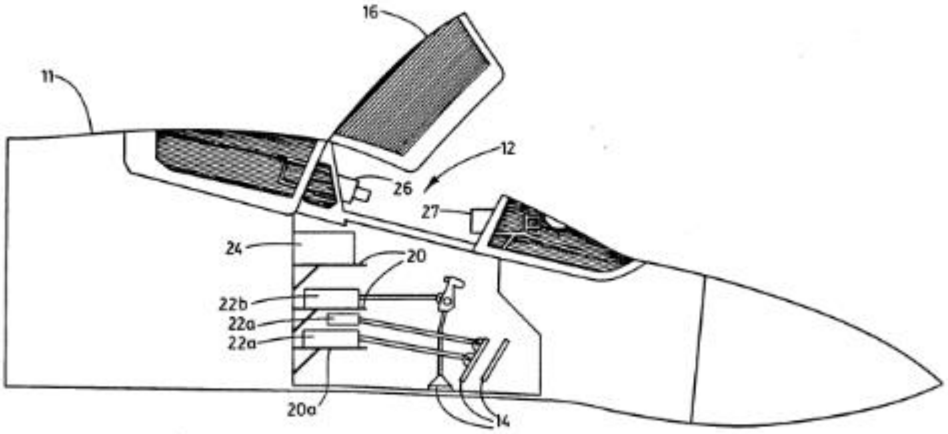


Fig.1.14. Illustrates the interior of an aircraft cockpit, control equipment, and various actuators.

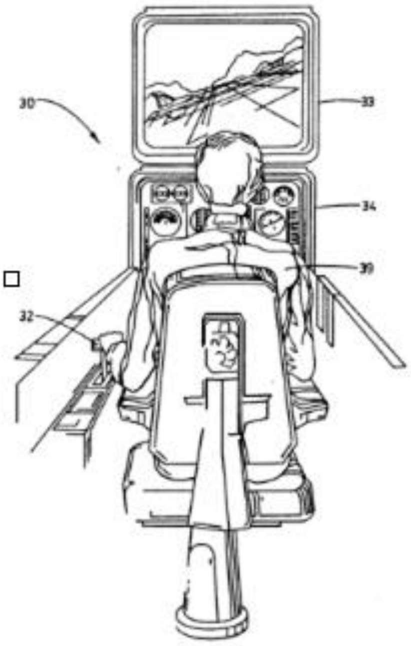


Fig.1.15. shows the view of the ground station and auxiliary equipment.

Fig.1.16. A top view of the ground station and auxiliary equipment is presented.

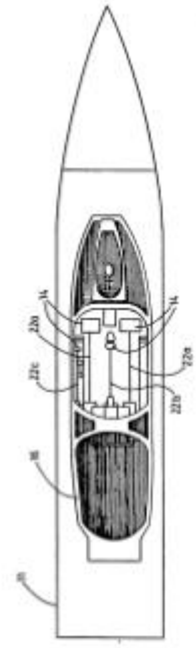


Fig.1.17. Illustrates a top view of an aircraft cockpit, control equipment, and a variety of executive mechanisms.

This system is a universal drone control system or, alternatively, a method of remote piloting of an aircraft. The Generic Drone Control System is a technically simple approach to turning airplanes into drones. The disclosed drone control system essentially includes a converted aircraft or other aerial vehicle and a ground station from which the drone is remotely controlled. As part of the drone control system, many means for transmitting information and data between the ground station and the drone are also disclosed.

Referring to the drawings, (fig. 1.12) and (fig. 1.13) this invention is shown, which contains an aircraft or an air vehicle (11), a ground station (30), a means of transmitting information from an aircraft to a ground station and a means of transmitting information from the ground station to the aircraft. In particular, the drone is converted from an operational aircraft (11). The aircraft (11) has a cabin (12) containing operational control equipment (14).

(Fig. 1.13), (Fig. 1.15) and (Fig. 1.16) show a ground station (30) that contains a duplicate cockpit having auxiliary equipment (32), identical control equipment (14) in the cockpit of the aircraft in operation (11). The ground station (30) is adapted to accommodate the operator (39) or other means of controlling the drone. The ground station (30) additionally includes means for displaying the visual image signals (28) received from the aircraft or aircraft (11) to the control means. This means for displaying video image signals (28) preferably includes multiple screens (34) of high resolution.

The means of transmitting information from the aircraft (11) to the ground station (30) is shown in (Fig.1.3) and (Fig.1.4), contains a plurality of video cameras (26, 27) for collecting video information both from outside the aircraft cabin (12) and from devices (not shown) located inside the cabin (12). In addition, one or more transmitters operatively connected to the video cameras (26,27) are used to send information collected from the video cameras (26,27) to the ground station (30). Further, at the ground station are receiver and display processors adapted to receive, process and display information transmitted from the aircraft.

The means of transmitting information from the ground station (30) to the aircraft (11) or the aircraft includes a transmitter, a receiver and a processor. In the best embodiment, control movements at the ground station (30) are converted from analog movements into digital flight control signals (35) and transmitted telemetrically to the aircraft (11). The telemetry flight control signals (35) are then used to control the control equipment (14) on the aircraft (11).

Since the cockpits and controls of the aircraft are designed for a certain range of population, a common set of actuators (22) is used to provide the full range of controls on any fighter or attack aircraft, essentially replacing the pilot. Thus, the overall drone control system (10), except for a few minor elements, will be completely useful and transferable to many types of aircraft. No permanent modifications to the existing aircraft are required. This, in turn, will allow the aircraft to remain fully manned without degrading systems or performance.

1.2 Object of research.

In this work, we will take the UAV of the quadcopter type as the object of research.

A quadcopter is an aircraft with four carrying propellers (Fig. 2.1). Two pairs of screws have opposite directions of rotation, which allows in normal mode

to extinguish dynamic moments created by screws. Among the many modern aircraft, the quadcopter belongs to unmanned aerial vehicles (UAVs), relatively cheap and easy to design. Quadcopters are able to fly in bad weather, hang in the air for a long time, monitor objects and perform many other tasks. They have found their application in rescue operations, in agriculture, in military affairs and in many other areas. For quadcopters, the tasks of route planning and management are relevant. These tasks have many different options, which take into account the limited resources of modern UAVs, and the need to take into account possible obstacles, for example, when organizing flights in rough terrain or urban environments, and accounting for weather conditions (in particular, wind conditions).

Conclusion to the chapter 1

The patents reviewed show different approaches to remote control of UAVs, focusing on reliability, safety and efficiency. The implementation of automatic control systems and backup flight plans ensures a high level of reliability and autonomy in UAV control, which is a key factor for successful use in various missions.

SECTION 2

MATHEMATICAL MODEL OF UAV MOVEMENT

2.1 Mathematical model of a quadcopter.

The mathematical model of the quadcopter will be obtained from the solid body motion model, taking into account the specific type of forces and moments generated by the aircraft's engines. Schematically, the quadrocopter is shown in (Fig.2.1). The moving coordinate system is chosen so that the x' and y' axes are directed to pairs of opposite rotors, and the z' axis is perpendicular to the plane in which the rotors are located.

Quadcopter engines form traction forces f1, f2, f3, f4 directed along the z axis of the moving coordinate system. The traction forces of the screws create torques M1, M2, M3, M4, directed along the x' and y axes, and the screws - dynamic moments Md1, Md2, Md3, Md4, directed along the z' axis.

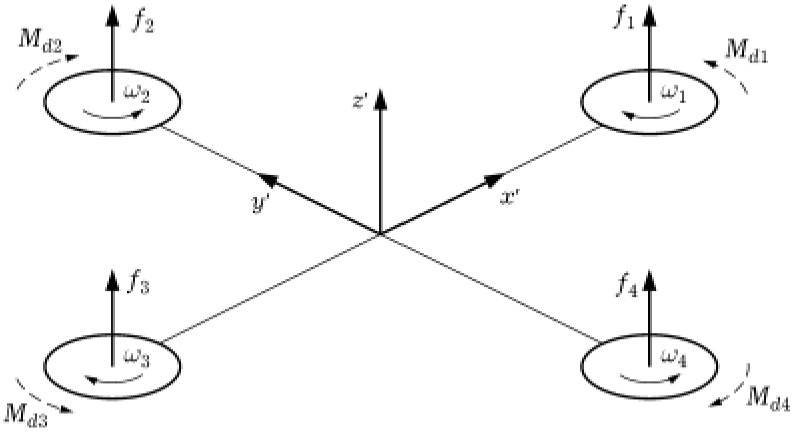


Fig.2.1. Diagram of a quadcopter

The power of each i-th rotor and the corresponding dynamic moment are determined by the speed ω_i of the rotor rotation. We use the following model (2.7):

$$f_i = k\omega_i^2, \quad M_{d,i} = b\omega_i^2, \tag{2.1}$$

Where k, b - Proportionality coefficients reflecting the features of the rotors. From these ratios it follows that the dynamic moments of the rotors are proportional to the traction forces.

Md,i = (b/k)fi. Therefore, it is convenient instead of the speeds w; as management to consider forces fi thrust of the rotors.

The total torque Mf of all four rotors can be written by the formula:

$$\boldsymbol{M}_f = l(f_2 - f_4)\boldsymbol{k}'_x + l(f_3 - f_1)\boldsymbol{k}'_y,$$

(2.2)

where L is the distance from the center of mass to the center of the rotor; k'x and k'y are the coordinates of the connected coordinate system directed along the corresponding axes.

The dynamic moments of the rotors give the vector Md, directed along the axis Oz' of the bound coordinate system, which, taking into account the directions of rotation of the rotors, can be written in the form:

$$\boldsymbol{M}_d = \frac{b}{k}(f_1 - f_2 + f_3 - f_4)\boldsymbol{k}'_z.$$

(2.3)

Finally, the total force of the four rotors is written as:

$$\boldsymbol{F} = (f_1 + f_2 + f_3 + f_4)\boldsymbol{k}'_z.$$

(2.4)

Thus, in the general model of the motion of a rigid body in the case of a quadcopter, we have:

$$F = f_1 + f_2 + f_3 + f_4, \quad M_x = (f_2 - f_4)l, \quad M_y = (f_3 - f_1)l, \quad M_z = \frac{b}{k}(f_1 - f_2 + f_3 - f_4).$$

(2.5)

The mathematical model of the motion of a rigid body in Krylov angles has the following form:

$$\begin{cases} \ddot{x} = \dfrac{F}{m}(-\cos\gamma\ \cos\psi\ \sin\vartheta + \sin\gamma\ \sin\psi); \\ \ddot{y} = \dfrac{F}{m}(-\cos\psi\ \sin\gamma - \cos\gamma\ \sin\psi\ \sin\vartheta); \\ \ddot{z} = -g + \dfrac{F}{m}\cos\gamma\ \cos\vartheta; \\ \dot{\psi} = \dfrac{\sin\gamma}{\cos\vartheta}\omega_y + \dfrac{\cos\gamma}{\cos\vartheta}\omega_z; \\ \dot{\vartheta} = -\omega_y\cos\gamma + \omega_z\sin\gamma; \\ \dot{\gamma} = \omega_x - \omega_y\sin\gamma\ \mathrm{tg}\,\vartheta - \omega_z\cos\gamma\ \mathrm{tg}\,\vartheta; \\ \dot{\omega}_x = \dfrac{M_x}{I_x} - \dfrac{\omega_y\omega_z(I_z - I_y)}{I_x}; \\ \dot{\omega}_y = \dfrac{M_y}{I_y} - \dfrac{\omega_z\omega_x(I_x - I_z)}{I_y}; \\ \dot{\omega}_z = \dfrac{M_z}{I_z} - \dfrac{\omega_x\omega_y(I_y - I_x)}{I_z}. \end{cases}$$

(2.6)

2.2 Movement of the quadcopter in the vertical plane.

If the quadcopter is moving in the vertical plane, we will choose a fixed coordinate system so that the movement occurs in the Oxz plane, i.e. for such motion, we have y = 0. Additionally, we assume that the condition ψ = 0 is fulfilled.

Taking into account the conditions y = 0 and ψ = 0, from the second equation of the system (2.6) we find that γ = 0, and with the help of the condition ψ = 0, from the fourth equation we find that ω_z = 0.

Also, from the sixth equation of the system, having already taken into account γ = 0 and ω_z = 0, we get ω_z = 0. From the seventh and ninth equations, we conclude that Mx = Mz =0.

The listed ratios allow you to simplify the system by reducing the number of equations to four:

$$\begin{cases} \ddot{x} = -\dfrac{F}{m}\sin\vartheta; \\ \ddot{z} = -g + \dfrac{F}{m}\cos\vartheta; \\ \dot{\vartheta} = -\omega_y; \\ \dot{\omega}_y = \dfrac{M_y}{I_y}. \end{cases}$$

(2.7)

The simplified control model includes variables F and My according to the formulas (form. 2.5).

In this condition Mx = Mz = 0 the control is imposed by the following connections:

$$f_2 - f_4 = 0, \quad f_1 - f_2 + f_3 - f_4 = 0.$$

(2.8)

These connections mean that the system (form. 2.6) actually has two-dimensional control. For convenience, we will replace the controls, considering:

$$u_1 = f_2, \quad u_2 = f_3 - f_1.$$

(2.9)

Then F = 4u1, My = u2. Note that the values of u1, u2, taking into account the control connections f1, f2, f3, f4, are uniquely restored. As a result of replacing the controls, we arrive at the following system, which describes the movement of the quadcopter in the vertical plane:

$$\begin{cases} \ddot{x} = -\dfrac{4u_1}{m}\sin\vartheta; \\ \ddot{z} = -g + \dfrac{4u_1}{m}\cos\vartheta; \\ \dot{\vartheta} = -\omega_y; \\ \dot{\omega}_y = \dfrac{lu_2}{I_y}. \end{cases}$$

(2.10)

Movement of the quadcopter in the horizontal plane.

The movement of the quadcopter in the horizontal plane means that the z coordinate of its position does not change during the movement, i.e. z = c = const. We will additionally impose a restriction on the orientation of the quadcopter as ϑ = 0.

From the third equation of the system (2.6) we find:

$$F\cos\gamma = mg.$$

(2.11)

From the fifth equation of this system, taking into account that ϑ = 0, we find:

$$\omega_y\cos\gamma - \omega_z\sin\gamma = 0.$$

(2.12)

The equation (2.12) connects three phase variables. It is better to consider the left side of the equation as a new phase variable. In this regard, we will replace the phase variables ω_x, ω_z with new ones μ, ν according to the formulas:

$$\begin{pmatrix}\mu\\ \nu\end{pmatrix} = \begin{pmatrix}\cos\gamma & -\sin\gamma\\ \sin\gamma & \cos\gamma\end{pmatrix}\begin{pmatrix}\omega_y\\ \omega_z\end{pmatrix}.$$

(2.13)

Taking into account the transformations and substitutions, we will get the derivatives of the new variables through the system:

$$\begin{cases}\dot\mu = \dfrac{M_y}{I_y}\cos\gamma - \dfrac{M_z}{I_z}\sin\gamma - \left(\dfrac{I_x - I_z}{I_y}\cos^2\gamma - \dfrac{I_y - I_x}{I_z}\sin^2\gamma + 1\right)\nu\omega_x;\\ \dot\nu = \dfrac{M_y}{I_y}\sin\gamma + \dfrac{M_z}{I_z}\cos\gamma - \left(\dfrac{I_x - I_z}{I_y} - \dfrac{I_y - I_x}{I_z}\right)\nu\omega_x.\end{cases}$$

(2.14)

The constraint (form. 2.12) after changing the variables takes the form u = 0. It follows from this constraint that u = 0, i.e.:

$$\frac{M_y}{I_y}\cos\gamma - \frac{M_z}{I_z}\sin\gamma - \left(\frac{I_x - I_z}{I_y}\cos^2\gamma - \frac{I_y - I_x}{I_z}\sin^2\gamma + 1\right)\nu\omega_x = 0.$$

(2.15)

Equations (2.11) and (2.15) are control restrictions. Thus, in this case, management is two-dimensional. We replace the management:

$$u_1 = \frac{M_x}{I_x}, \quad u_2 = \frac{M_y}{I_y}\sin\gamma + \frac{M_z}{I_z}\cos\gamma.$$

(2.16)

We get the following system:

$$\begin{cases} \ddot{x} = g\,\mathrm{tg}\,\gamma\,\sin\psi; \\ \ddot{y} = -g\,\mathrm{tg}\,\gamma\,\cos\psi; \\ \dot{\psi} = \nu; \\ \dot{\gamma} = \omega_x; \\ \dot{\omega}_x = u_1 - \dfrac{I_z - I_y}{I_x}\nu^2\cos\gamma\,\sin\gamma; \\ \dot{\nu} = u_2 - \left(\dfrac{I_x - I_z}{I_y} + \dfrac{I_y - I_x}{I_z}\right)\nu\,\omega_x\sin\gamma\,\cos\gamma. \end{cases}$$

(2.17)

2.3 A simplified mathematical model of the movement of a quadcopter in the vertical plane.

For a quadcopter with four propellers in the vertical plane, the mathematical model will take into account the thrust of all four propellers and their effect on the movement. Let's assume that these propellers are located symmetrically with respect to the center of mass of the quadcopter.

Main variables and parameters: (x) - horizontal coordinate of the quadcopter. (z) is the vertical coordinate of the quadcopter.

(Theta) - Angle of inclination of the quadcopter relative to the horizontal (pitch). (m) is the mass of the quadcopter. (g) – acceleration of free fall. (I) - the moment of inertia of the quadcopter relative to its center of mass. (T_1, T_2, T_3, T_4) - thrusts of four propellers. (L) is the distance from the center of mass to each propeller (it is assumed that the propellers are located symmetrically at the distance (L)).

Equation of motion: Dynamics in the horizontal plane (x axis):

$$\dot{x} = v_x$$

$$\dot{v_x} = \frac{(T_1 + T_2 + T_3 + T_4)\sin(\theta)}{m}$$

(2.18)

Dynamics in the vertical plane (axis (z)):

$$\dot{z} = v_z$$

$$\dot{v_z} = \frac{(T_1 + T_2 + T_3 + T_4)\cos(\theta) - mg}{m}$$

(2.19)

Angular movement (axis θ):

$$\dot{\theta} = \omega$$

$$\dot{\omega} = \frac{L(T_1 + T_2 - T_3 - T_4)}{I}$$

(2.20)

State vector: Enter the system state vector:

$$\mathbf{X} = \begin{bmatrix} x \\ v_x \\ z \\ v_z \\ \theta \\ \omega \end{bmatrix}$$

(2.21)

System of differential equations: The evolution of the state of the system over time can be written in the form:

$$\begin{cases} \dot{x} = v_x \\ \dot{v}_x = \frac{(T_1+T_2+T_3+T_4)\sin(\theta)}{m} \\ \dot{z} = v_z \\ \dot{v}_z = \frac{(T_1+T_2+T_3+T_4)\cos(\theta)-mg}{m} \\ \dot{\theta} = \omega \\ \dot{\omega} = \frac{L(T_1+T_2-T_3-T_4)}{I} \end{cases} \tag{2.22}$$

Controlling influences: T_1, T_2, T_3, T_4

For a complete description of the model, it is also necessary to take into account the thrust limitations of the propellers, the minimum and maximum values of T_min and T_max, as well as possible additional moments related to aerodynamic effects and air resistance.

These equations are the basis for modeling the movement of a quadcopter with four propellers in the vertical plane. The system of differential equations allows you to numerically study the behavior of the quadcopter and develop algorithms for controlling its movement.

A simplified mathematical model of the movement of the quadcopter in the horizontal plane.

Consider the following aspects: coordinate systems, forces and moments acting on the quadcopter, and equations of motion.

Coordinate systems: Inertial coordinate system (earth system): (O_X Y Z). The (X) axis points north, (Y) east, (Z) up. The coordinate system with the quadcopter: (O_x y z). Main variables: (X, Y) – coordinates of the quadcopter in the horizontal plane (in the inertial system). - Yaw angle, which describes the orientation of the quadcopter around the vertical axis (Z). (u, v) - horizontal components of the speed of the quadcopter in the bound coordinate system. (r) - Angular speed around the axis (Z) (Yaw rate). - Pitch angle. - Roll angle.

Kinematic equations: Kinematic equations relate the speed and position of the quadcopter:

$$\begin{cases} \dot{X} = u\cos(\psi) - v\sin(\psi) \\ \dot{Y} = u\sin(\psi) + v\cos(\psi) \\ \dot{\psi} = r \end{cases}$$

(2.23)

where: $\dot{X}$ и $\dot{Y}$ - Speed of change of coordinates in the inertial system. $\dot{\psi}$ - Yaw rate.

Dynamic Equations: The main forces and moments acting on the quadcopter include thrust from each rotor and aerodynamic drag. Moments include risk moments created by rotors. The dynamic equations describe the movement of the quadcopter under the influence of these forces and moments:

$$\begin{cases} m(\dot{u} - vr) = -mg\sin(\theta) \\ m(\dot{v} + ur) = mg\cos(\theta)\sin(\phi) \\ I_z\dot{r} = N \end{cases}$$

(2.24)

where: (m) the mass of the quadcopter. (g) – acceleration of free fall. (I_z) - the moment of inertia of the quadcopter around the (Z) axis. (N) is the moment of searching.

Forces and Moments: Let us denote the total thrust (T) and the moments about the axes (x), (y), and (z) as (L), (M), and (N), respectively:

$$\begin{cases} T = T_1 + T_2 + T_3 + T_4 \\ L = d(T_2 - T_4) \\ M = d(T_3 - T_1) \\ N = k(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{cases}$$

(2.25)

where: (T_i) - thrust of the rotors. (d) is the distance from the center of mass of the quadcopter to the rotor. (k) - the coefficient associated with moments from the rotation of the rotors. - Rotor angular velocity (i).

The complete system of equations: By combining the kinematic and dynamic equations, we get the complete system of equations for the movement of the quadcopter in the horizontal plane:

$$\begin{cases} \dot{X} = u\cos(\psi) - v\sin(\psi) \\ \dot{Y} = u\sin(\psi) + v\cos(\psi) \\ \dot{\psi} = r \\ m(\dot{u} - vr) = -mg\sin(\theta) \\ m(\dot{v} + ur) = mg\cos(\theta)\sin(\phi) \\ I_z\dot{r} = N \\ T = T_1 + T_2 + T_3 + T_4 \\ L = d(T_2 - T_4) \\ M = d(T_3 - T_1) \\ N = k(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{cases}$$

(2.26)

Control actions: The control of the quadcopter is carried out by changing the rotation speeds of the rotors, which allows you to adjust the thrust (T), moments (L, M, N), (θ, ϕ), as well as horizontal movement.

Conclusion to Chapter 2

Relevance: The mathematical model of quadcopter motion presented in this section is important and relevant for several reasons:

Advances in Unmanned Aerial Vehicle (UAV) technology: Quadcopters have become an integral part of many industries. Therefore, the development of an accurate mathematical model for their movement is critical for the further improvement of these technologies.

Control algorithms Mathematical models: are the basis for the development of quadcopter control algorithms. In particular, they allow creating stabilization and navigation systems that ensure safe and efficient flight.

Disadvantages: Despite its relevance, the presented model has several disadvantages and limitations:

Simplification and Idealization: Many aspects of the model are simplified or idealized. For example, aerodynamic effects, turbulence, wind loads and other external factors that can significantly affect the movement of the quadcopter are not taken into account.

Model linearity: In some equations, a linear approximation is used, which may not be accurate enough to describe the behavior of the quadcopter in real conditions, where systems may behave non-linearly. The model implies some control restrictions that may not take into account all possible options for maneuvering the quadcopter.

Orientation to ideal conditions: The model assumes ideal conditions where all rotors operate flawlessly and symmetrically. In fact, there may be failures or asymmetries in the operation of the rotors, which must also be taken into account.

Necessity of numerical methods: For the practical application of the model, it is necessary to use numerical methods to solve differential equations, which require additional computing resources and can be complex in real time.

Conclusion: The presented mathematical model of the movement of the quadcopter is an important basis for understanding the movement of this UAV and the development of control algorithms. It takes into account the main forces and moments acting on the quadcopter and allows modeling its movement in the vertical and horizontal planes. However, for a more accurate description of the movement of the quadcopter, it is necessary to consider additional factors and improve the model so that it is more realistic and applicable to real operating conditions.

SECTION 3

DEVELOPMENT OF ALGORITHM AND PROGRAM FOR UAV TRAFFIC CONTROL. EXAMPLE OF CALCULATING THE MOVEMENT OF A UAV WITH REMOTE CONTROL

3.1. Development of a movement algorithm in a vertical plane in C#.

To model the motion of a quadcopter in the vertical plane in Unity 2D in C#, you need to consider the vertical forces and moments acting on the quadcopter. We will use Unity to render and model the quadcopter physics in the vertical plane.

Creating the algorithm and flowchart for Unity 2D to work while the character is moving involves several key steps. We will first consider the general algorithm and then present it as a block diagram.

Character movement algorithm in Unity 2D:

1. Initialization:

Loading scene.

Initialization of objects and variables.

Access to components such as Rigidbody2D.

2. Update input (Update):

Reading user input (keystrokes).

Update state variables based on input.

3. Physical update (FixedUpdate):

Calculation of forces and moments.

Updating the velocities and position of the object using the equations of motion.

Applying forces to Rigidbody2D.

4. Drawing:

Update character position and animation on screen.

5. End of frame:

Completion of the current frame and preparation for the next one.

To create a block diagram, you first need to define the main blocks that will be used:

1. The beginning.
2. Initialization.

3. User input.
4. Calculation of physics.
5. Restoring the position.
6. The end.
7. The beginning.

Here's what it might look like in flowchart form (Fig.3.1):

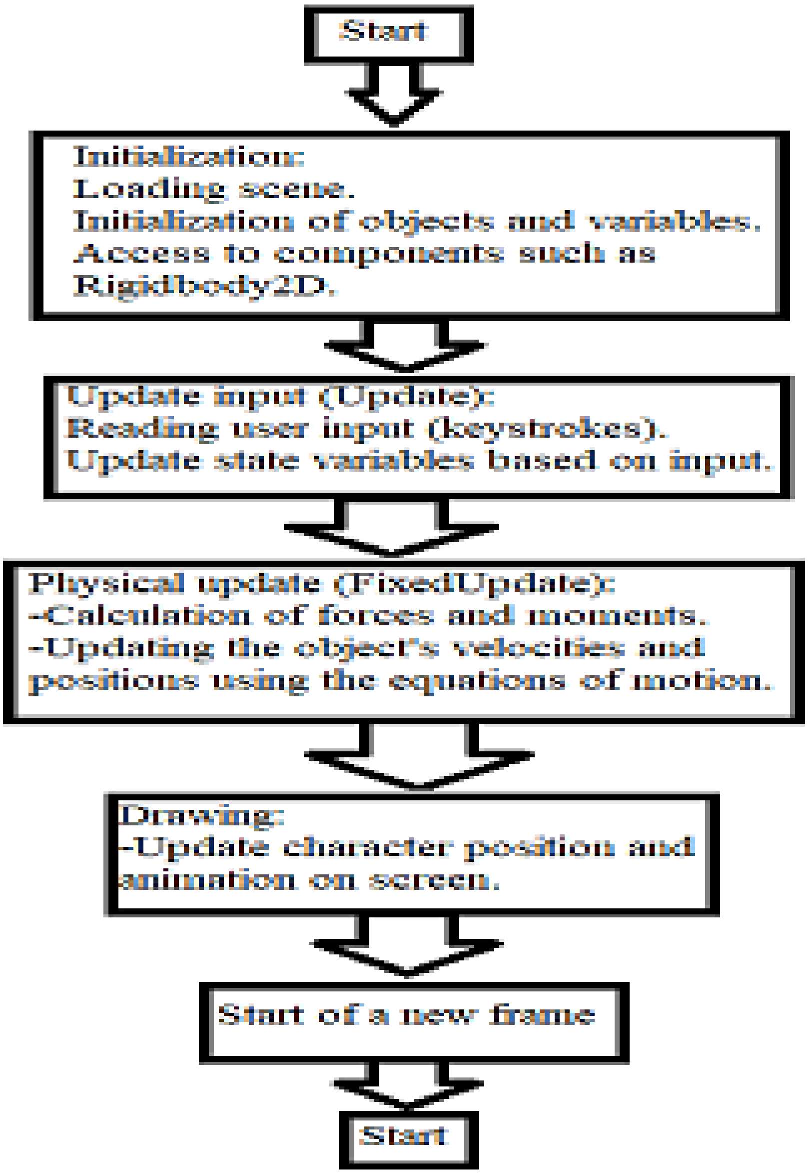


Fig 3.1

This algorithm covers the basic steps of processing user input, calculating physics, and updating the position of a Unity 2D character.

Example code for Body.cs:

First, let's set some variables for our object, let's call the script in which the variables that we will use as "Body" are written. (see Appendix A)

Constants and parameters:

G = 9.81 m/s² - acceleration of free fall

M = 1.0 kg - the mass of the quadcopter

I = 0.01 kg·m² - moment of inertia around the axis

D = 0.1 m - distance from the center UAV to the rotor

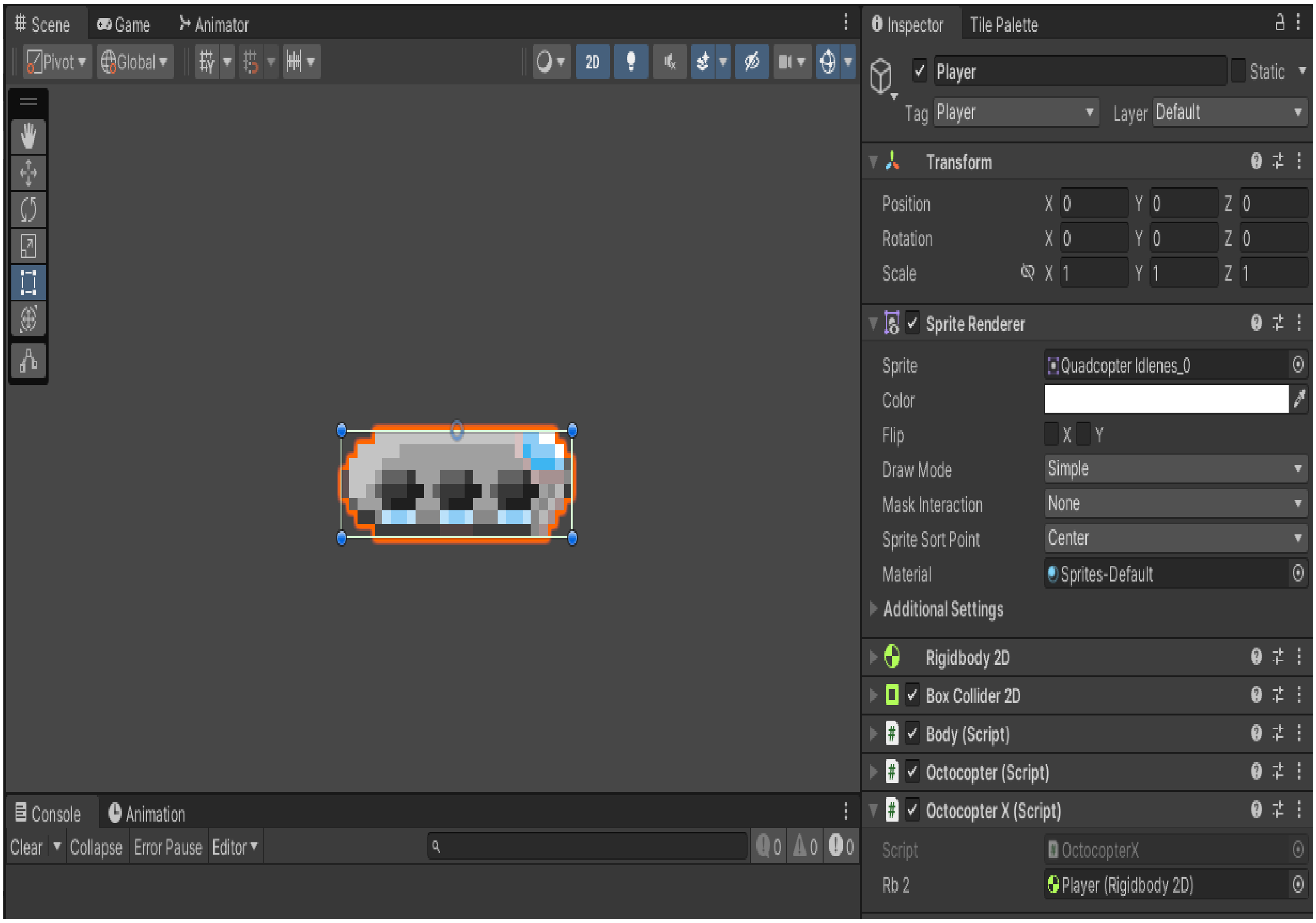
K = 0.0001 - rotor moment coefficient

Rotor rotation speeds:

$$T_1 = 3 \text{ rad/c}, T_2 = 3 \text{ rad/c}, T_3 = 3 \text{ rad/c}, T_4 = 3\text{rad/c} \tag{3.1}$$

Rotor thrusts:

$$\begin{aligned}
T_1 &= k \cdot \omega_1^2 = 0.0001 \cdot 3^2 = 0.0009 \text{ [H*m]} \\
T_2 &= k \cdot \omega_2^2 = 0.0001 \cdot 3^2 = 0.0009 \text{ [H*m]} \\
T_3 &= k \cdot \omega_3^2 = 0.0001 \cdot 3^2 = 0.0009 \text{ [H*m]} \\
T_4 &= k \cdot \omega_4^2 = 0.0001 \cdot 3^2 = 0.0009 \text{ [H*m]}
\end{aligned} \tag{3.2}$$



☑ Player Static
Tag Player Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

☑ Sprite Renderer
Sprite Quadcopter Idlenes_0
Color
Flip ☐ X ☐ Y
Draw Mode Simple
Mask Interaction None
Sprite Sort Point Center
Material Sprites-Default
Additional Settings

Rigidbody 2D
☑ Box Collider 2D
☑ Body (Script)
☑ Octocopter (Script)
☑ Octocopter X (Script)
Script OctocopterX
Rb 2 Player (Rigidbody 2D)

Fig.3.2. Object parameters

In (Fig. 3.2) shows the object (GameObject), its coordinates and position can be seen in the Inspector icon, scripts are also attached to it, Rigidbody2D is attached, and BoxCollider2D is attached.

3.2 Example of code calculation for Octocopter.cs:

Next, using the variables we get in the “Body” script, we can start calculating the movement in some axis, for us it is the Y(vertical) axis (see appendix B)

Total thrust and moments:

$$F_{\text{total}} = T_1 + T_2 + T_3 + T_4 = 0.0009 + 0.0009 + 0.0009 + 0.0009 = 0.0036 \quad [H]$$

(3.3)

The moment created by the opposite rotors:

$$M = d \cdot (T_2 - T_4) = 0.1 \cdot (0.0009 - 0.0009) = 0$$

(3.4)

Equation of motion:

Vertical acceleration:

$$\ddot{y} = \frac{F_{\text{total}} - m \cdot g}{m} = \frac{0.0036 - 1.0 \cdot 9.81}{1.0} = -9.8064\,\text{м}/\text{с}^2$$

(3.5)

Angular acceleration:

$$\ddot{\theta} = \frac{M}{I} = \frac{0}{0.01} = 0\,\text{рад}/\text{с}^2$$

(3.6)

Kinematic equations:

$$\begin{aligned} \dot{y} &= \dot{y} \\ \dot{\theta} &= \dot{\theta} \end{aligned}$$

(3.7)

State update (Euler method):

Vertical velocity update:

$$\dot{y}(t + \Delta t) = \dot{y}(t) + \ddot{y}(t) \cdot \Delta t$$

(3.8)

Angular velocity update:

$$\dot{\theta}(t + \Delta t) = \dot{\theta}(t) + \ddot{\theta}(t) \cdot \Delta t$$

(3.9)

Angle update:

$$\theta(t + \Delta t) = \theta(t) + \dot{\theta}(t) \cdot \Delta t$$

(3.10)

Calculation example

Assume that the initial values of velocities and angles are zero:

$$\begin{aligned} \dot{y}(0) &= 0\,\text{м}/\text{с} \\ \dot{\theta}(0) &= 0\,\text{рад}/\text{с} \\ \theta(0) &= 0\,\text{рад} \end{aligned}$$

(3.11)

Time step:

$$\Delta t = 0.01\,\text{с}$$

(3.12)

First Step Acceleration:

$$\ddot{y} = -9.8064\,\text{м}/\text{с}^2$$
$$\ddot{\theta} = 0\,\text{рад}/\text{с}^2$$

(3.13)

Update speeds:

$$\dot{y}(0.01) = 0 + (-9.8064) \cdot 0.01 = -0.098064\,\text{м}/\text{с}$$
$$\dot{\theta}(0.01) = 0 + 0 \cdot 0.01 = 0\,\text{рад}/\text{с}$$

(3.14)

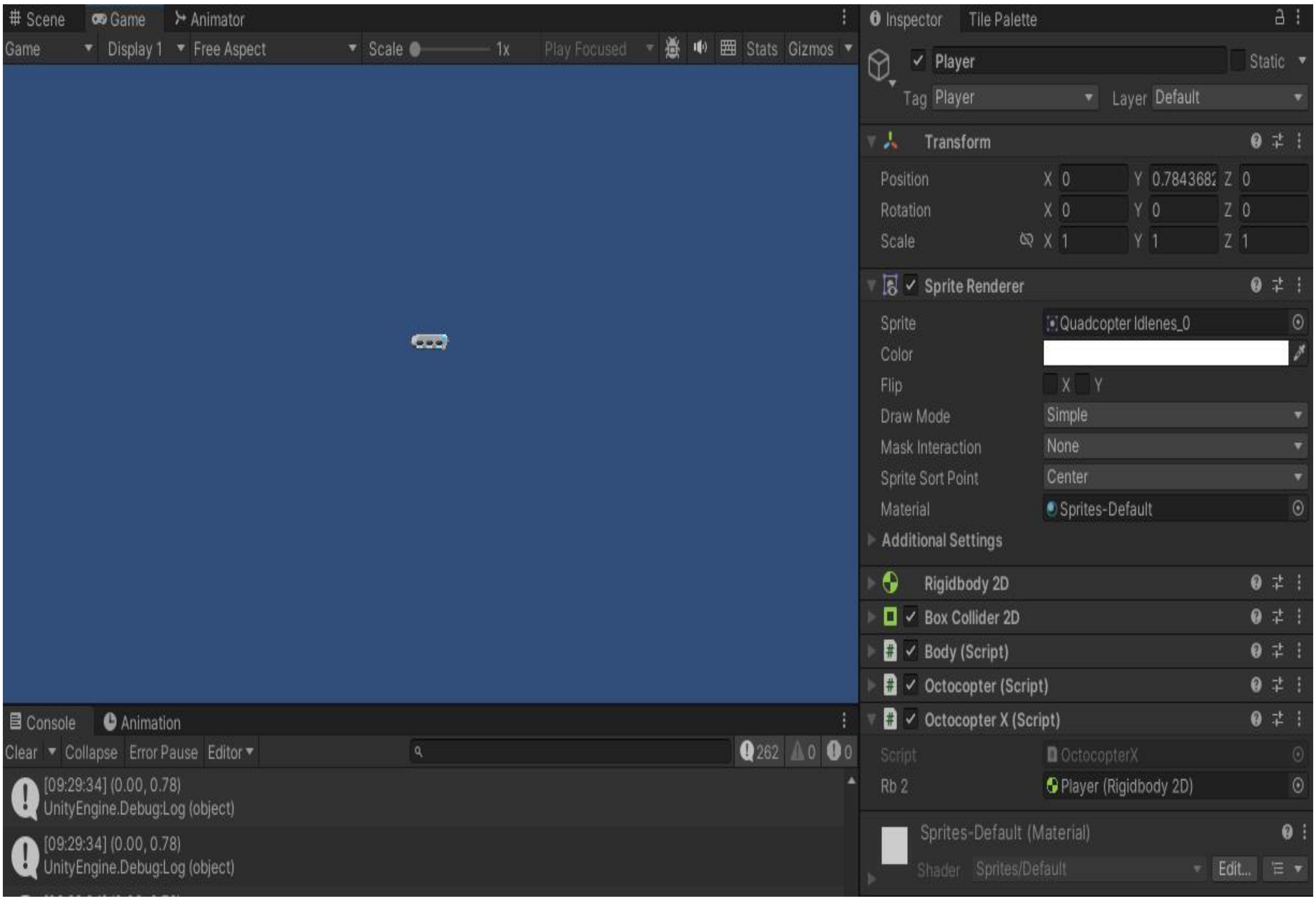
Angle update:

$$\theta(0.01) = 0 + 0 \cdot 0.01 = 0\,\text{рад}$$

(3.15)

Thus, after 0.01 seconds, the vertical velocity of the quadcopter will be - 0.098064 m/s, and the angle and angular velocity will remain zero.

In figures 3.3 - 3.5, we can see the compilation of the code, and the change of Y coordinates when the UAV moves vertically, in the Inspector, and in the console. The UAV positions change in the vertical plane by pressing the W, S keys. After compiling the code, the value for the variable (speedYDot) is determined. Then, in order for Unity to understand how much to move the object, we add to rb1.position (the current position of the object) Vector2 (two-dimensional position) and multiply by our variable speedYDot, and multiply by Time.fixedDeltaTime (Fixed interval in seconds in-game time)



```
[09:29:34] (0.00, 0.78)
UnityEngine.Debug:Log (object)
[09:29:34] (0.00, 0.78)
UnityEngine.Debug:Log (object)
```

Player
Tag Player
Layer Default

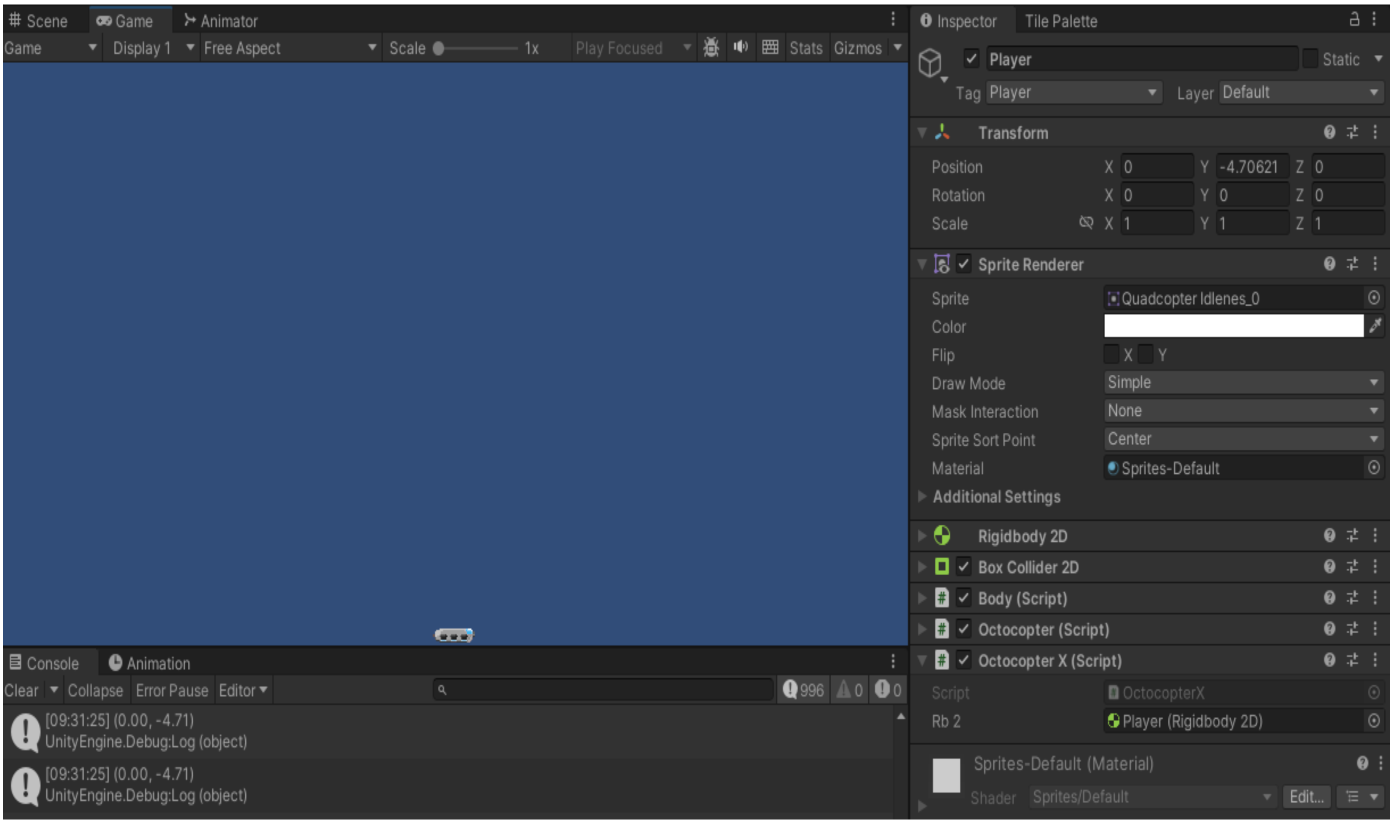
Transform
Position X 0 Y 0.7843682 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Sprite Renderer
Sprite Quadcopter Idlenes_0
Draw Mode Simple
Mask Interaction None
Sprite Sort Point Center
Material Sprites-Default

Octocopter X (Script)
Script OctocopterX
Rb 2 Player (Rigidbody 2D)

Sprites-Default (Material)
Shader Sprites/Default

Fig. 3.3



```
[09:31:25] (0.00, -4.71)
UnityEngine.Debug:Log (object)
[09:31:25] (0.00, -4.71)
UnityEngine.Debug:Log (object)
```

Player
Tag Player
Layer Default

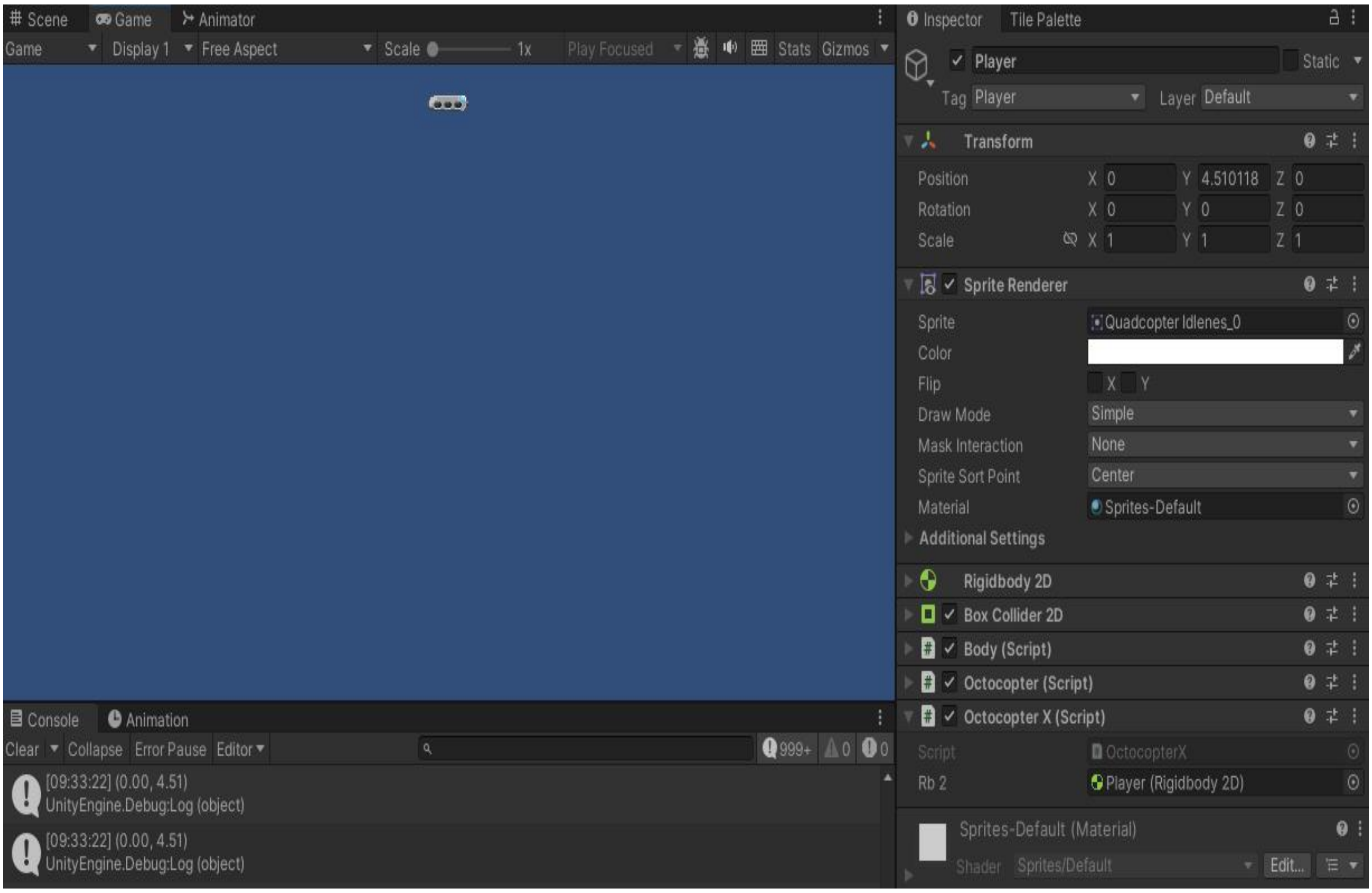
Transform
Position X 0 Y -4.70621 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Sprite Renderer
Sprite Quadcopter Idlenes_0
Draw Mode Simple
Mask Interaction None
Sprite Sort Point Center
Material Sprites-Default

Octocopter X (Script)
Script OctocopterX
Rb 2 Player (Rigidbody 2D)

Sprites-Default (Material)
Shader Sprites/Default

Fig. 3.4



Player

Tag Player

Layer Default

Transform

Position X 0 Y 4.510118 Z 0

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Sprite Renderer

Sprite Quadcopter Idlenes_0

Color

Flip X Y

Draw Mode Simple

Mask Interaction None

Sprite Sort Point Center

Material Sprites-Default

Octocopter X (Script)

Script OctocopterX

Rb 2 Player (Rigidbody 2D)

Sprites-Default (Material)

Shader Sprites/Default

```
[09:33:22] (0.00, 4.51)
UnityEngine.Debug:Log (object)
[09:33:22] (0.00, 4.51)
UnityEngine.Debug:Log (object)
```

Fig. 3.5

Example code for OctocopterX.cs :

After calculating the movement along the X axis, let's start calculating the movement along the X axis (see Appendix C)

Total thrust and moments:

$$F_{\text{total}} = T_1 + T_2 + T_3 + T_4 = 0.0009 + 0.0009 + 0.0009 + 0.0009 = 0.0036$$

(3.16)

The moment created by the opposite rotors:

$$M = d \cdot (T_2 - T_4) = 0.1 \cdot (0.0009 - 0.0009) = 0$$

(3.17)

Equation of motion:

$$\begin{array}{ll} \ddot{x} = -m \cdot \cos(\psi) & \ddot{x} = -1.0 \cdot \cos(0) = -1.0 \\ \ddot{y} = g \cdot \cos(\psi) \cdot \sin(\psi) & \ddot{y} = 9.81 \cdot \cos(0) \cdot \sin(0) = 0 \\ \dot{\psi} = \frac{N}{I} & \dot{\psi} = \frac{0}{0.01} = 0 \end{array} \tag{3.18}$$

Where *x* and *y* are the position coordinates, ψ is the course angle, *N* is the moment, *I* is the moment of inertia.

Kinematic equations:

$$\begin{array}{ll} \dot{x} = \dot{u} \cdot \cos(\psi) - \dot{v} \cdot \sin(\psi) & \dot{x} = 0 \cdot \cos(0) - 0 \cdot \sin(0) = 0 \\ \dot{y} = \dot{u} \cdot \sin(\psi) + \dot{v} \cdot \cos(\psi) & \dot{y} = 0 \cdot \sin(0) + 0 \cdot \cos(0) = 0 \\ \dot{\psi} = \frac{N}{I} & \dot{\psi} = 0 \end{array} \tag{3.19}$$

Where $\dot{u}$ and $\dot{v}$ are speeds along the axes.

Integration by the Euler method:

$$\begin{array}{ll} u_{t+1} = u_t + \ddot{u} \cdot \Delta t & u_{t+1} = 0 + (-1.0) \cdot 0.1 = -0.1 \\ v_{t+1} = v_t + \ddot{v} \cdot \Delta t & v_{t+1} = 0 + 0 \cdot 0.1 = 0 \\ \psi_{t+1} = \psi_t + \dot{\psi} \cdot \Delta t & \psi_{t+1} = 0 + 0 \cdot 0.1 = 0 \end{array} \tag{3.20}$$

Thus, after one time step $\Delta t = 0.1$, the state of the quadcopter will be as follows:

X-axis speed: -0.1

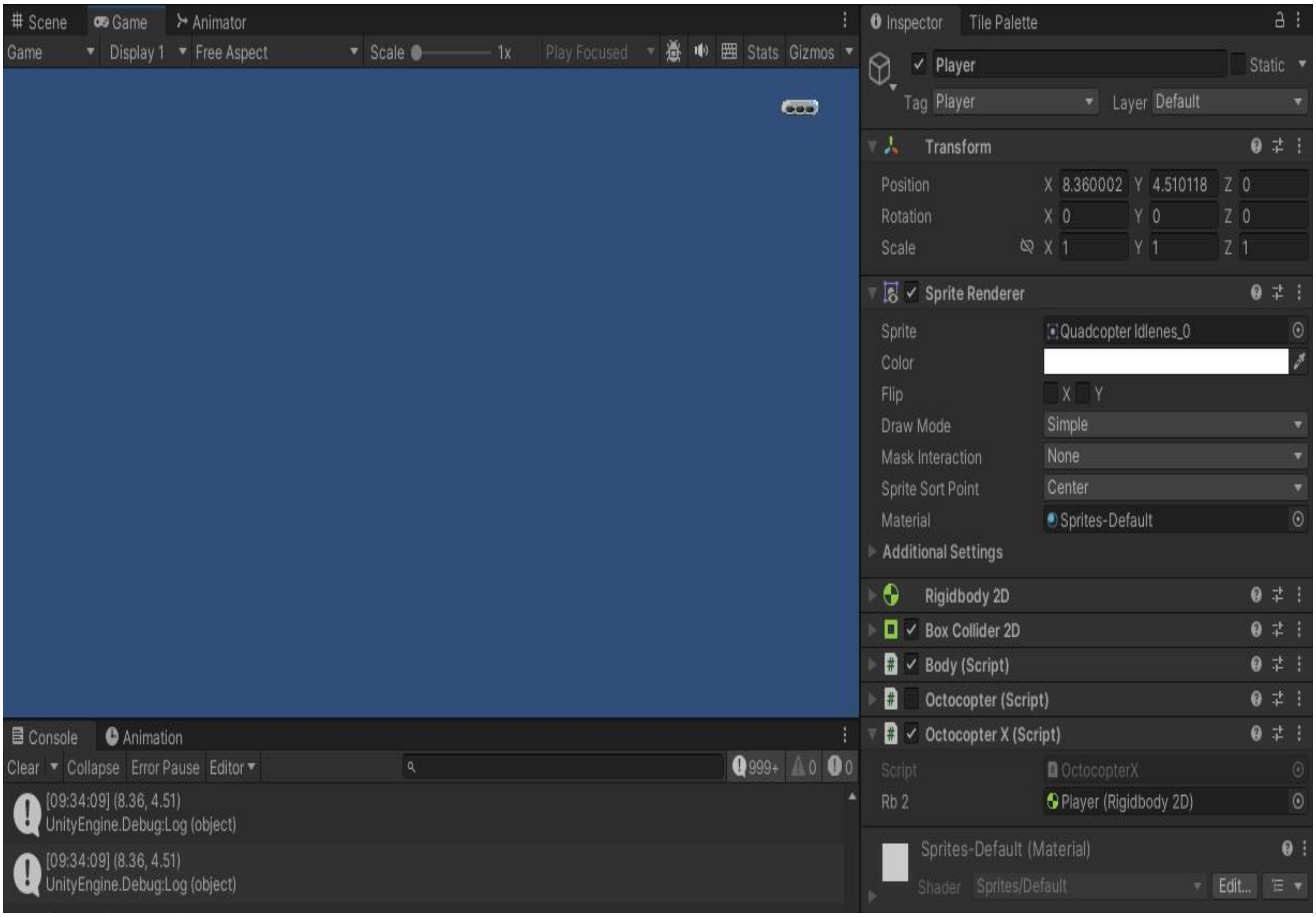
Y axis speed: 0

Heading angle: 0

In Figures 3.6 - 3.8, we can see the compilation of the code, and the change of Y and X coordinates, when the UAV moves along the vertical axis and horizontally, Inspector, and in the console.

As in the example with vertical calculation, the UAV position changes in the vertical plane when pressing the A, D keys. After compiling the code, the value for the variable (speedXDot) is determined. Then, in order for Unity to understand how much to move the object, we add Vector2 (two-dimensional position) to rb2.position (the current position of the object) and multiply by our speedXDot variable, and multiply by Time.fixedDeltaTime (Fixed interval in seconds in-game time)

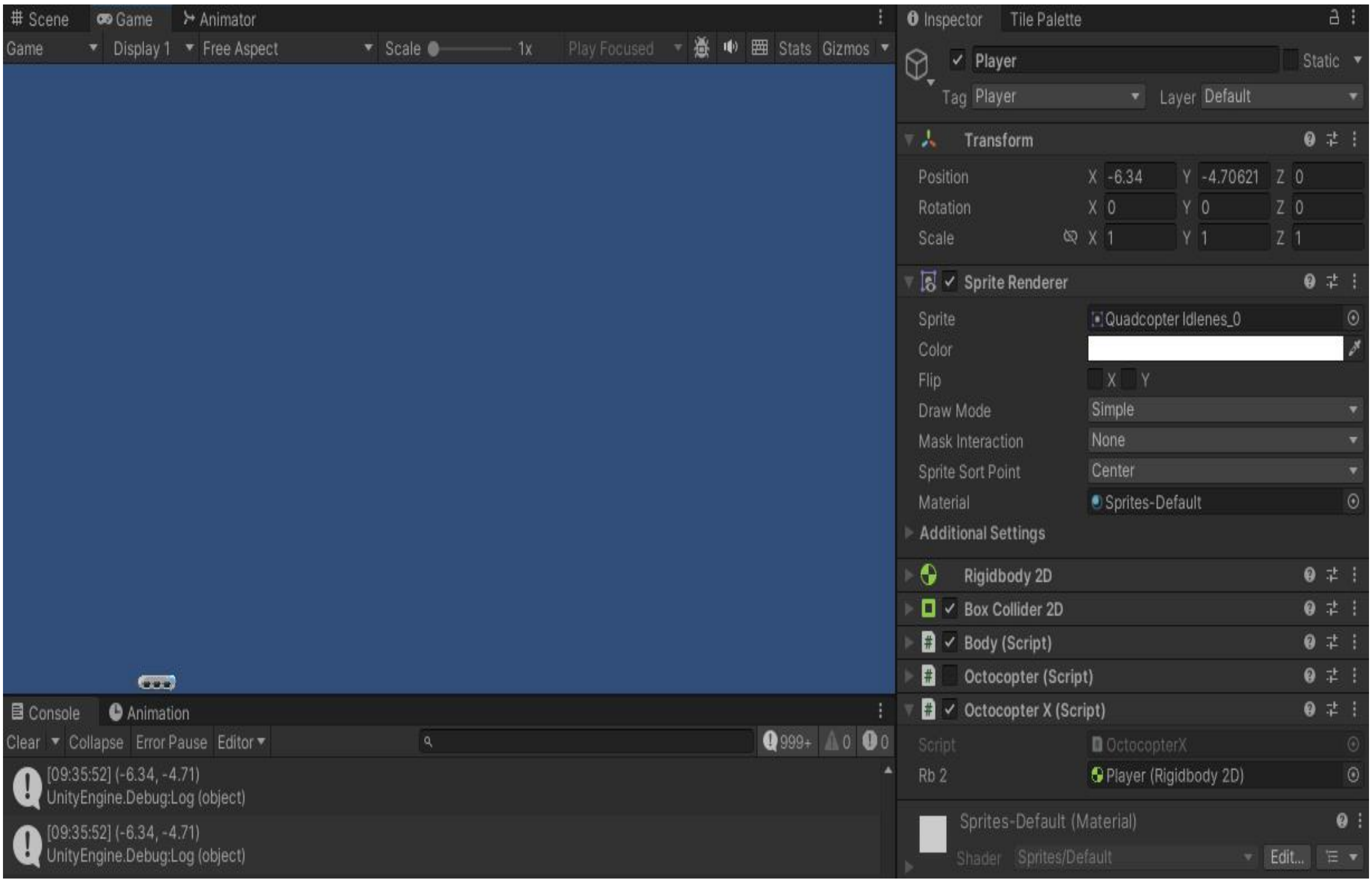
Explanation: Constants: The physical parameters of the quadcopter, such as mass, moment of inertia and distance to the rotors, are defined. Rotor rotation speeds: Initialized values for rotor rotation speeds. State variables: Quadcopter coordinates (speedY, angulVelocity, incline, speedX, rateYaw, Yaw). FixedUpdate method: Updates the state of the quadcopter based on the current rotor speeds and step time (using `Time.deltaTime`). Thrusts from each rotor, forces and moments acting on the quadcopter are calculated and its position and orientation are updated.



```
[09:34:09] (8.36, 4.51)
UnityEngine.Debug:Log (object)
[09:34:09] (8.36, 4.51)
UnityEngine.Debug:Log (object)
```

Player
Tag: Player
Layer: Default
Position: X 8.360002 Y 4.510118 Z 0
Rotation: X 0 Y 0 Z 0
Scale: X 1 Y 1 Z 1
Sprite: Quadcopter Idlenes_0
Color:
Flip: X Y
Draw Mode: Simple
Mask Interaction: None
Sprite Sort Point: Center
Material: Sprites-Default
Script: OctocopterX
Rb 2: Player (Rigidbody 2D)
Sprites-Default (Material)
Shader: Sprites/Default

Fig. 3.6



Player
Tag Player
Layer Default

Position X -6.34 Y -4.70621 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

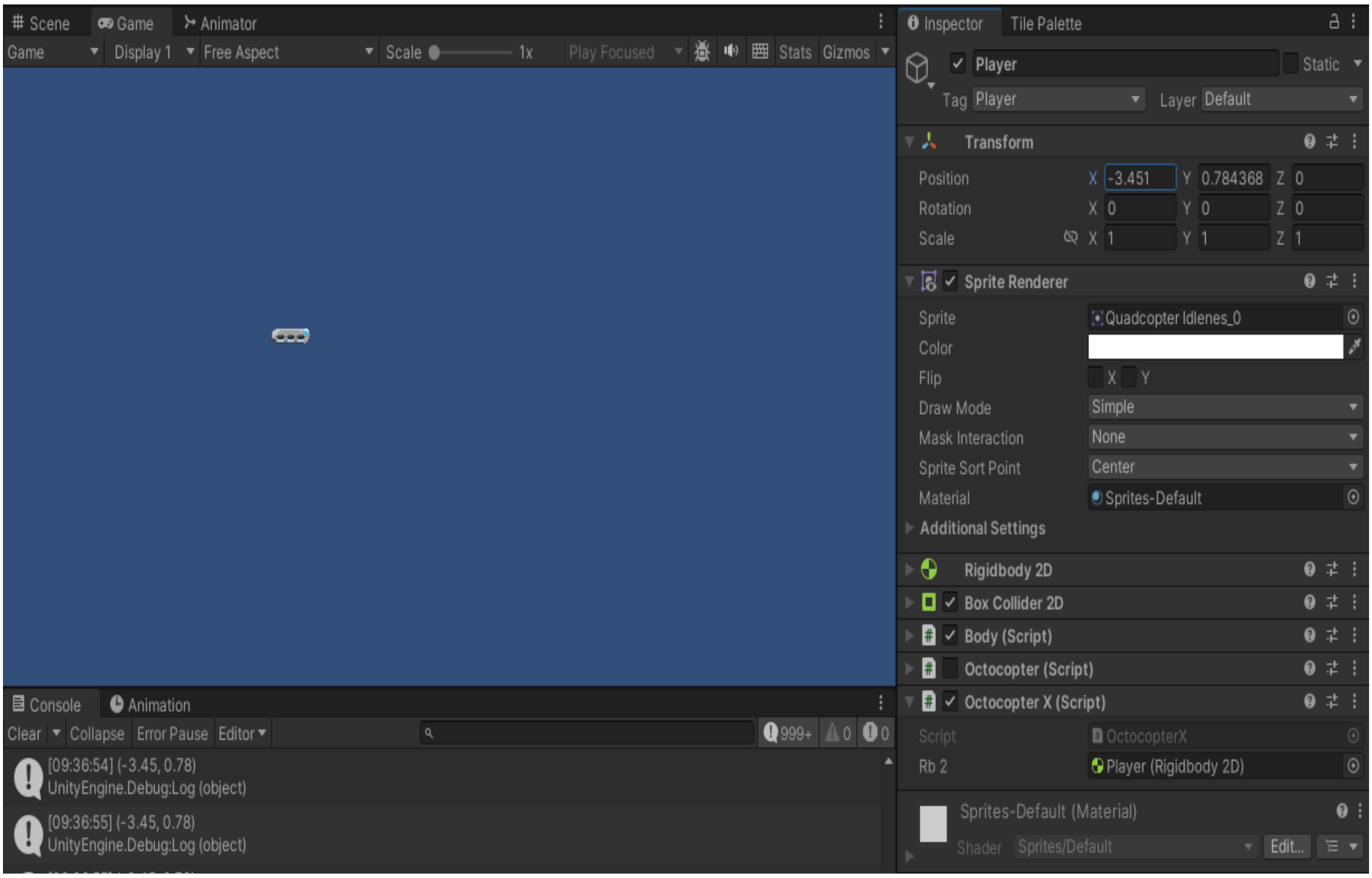
Sprite Quadcopter Idlenes_0
Color
Flip X Y
Draw Mode Simple
Mask Interaction None
Sprite Sort Point Center
Material Sprites-Default

Script OctocopterX
Rb 2 Player (Rigidbody 2D)

Shader Sprites/Default

```
[09:35:52] (-6.34, -4.71)
UnityEngine.Debug:Log (object)
[09:35:52] (-6.34, -4.71)
UnityEngine.Debug:Log (object)
```

Fig. 3.7



Player
Tag Player
Layer Default

Position X -3.451 Y 0.784368 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Sprite Quadcopter Idlenes_0
Color
Flip X Y
Draw Mode Simple
Mask Interaction None
Sprite Sort Point Center
Material Sprites-Default

Script OctocopterX
Rb 2 Player (Rigidbody 2D)

Shader Sprites/Default

```
[09:36:54] (-3.45, 0.78)
UnityEngine.Debug:Log (object)
[09:36:55] (-3.45, 0.78)
UnityEngine.Debug:Log (object)
```

Fig. 3.8

Conclusion to Chapter 3

Relevance: The development of an algorithm and program for controlling the movement of a quadcopter in the vertical and horizontal planes based on a mathematical model and its implementation in Unity 2D is extremely relevant in the modern context. UAVs (unmanned aerial vehicles).

In particular:

Expanding the capabilities of the UAV: The implementation of precise control of the movement of the quadcopter allows you to create more complex and efficient algorithms for performing various tasks.

Educational purposes: Unity is a popular platform for training developers, so the creation of similar models can be used for training courses and labs at universities and other educational institutions.

Algorithm testing: The model in Unity allows for numerical simulations, which is a safe and cost-effective way to test algorithms before they are used in real-world conditions.

Advantages:

Modeling in Unity: Using Unity to visualize and simulate quadcopter motion provides a user-friendly interface and powerful tools for development and testing.

Realistic control: Algorithms that take into account vertical forces, moments, rotor thrusts and kinematic equations allow you to create a realistic quadcopter control model.

Euler's Method: Using Euler's method for integration allows you to easily and quickly calculate new quadcopter positions and velocities, which is important for a real hour of simulations.

Disadvantages:

Model Simplification: A quadcopter model may be too simplistic and not take into account all the factors that affect its movement, such as aerodynamic effects, air turbulence, etc. This may lead to deviations from the actual behavior of the device.

Limitations of numerical integration methods: The Euler method used for integration has limited accuracy and can become unstable at larger hour steps. Other methods, such as the Runge-Kutta method, can provide greater accuracy and stability.

Lack of feedback: In the presented model, there are no feedback mechanisms that would allow the quadcopter to adjust its movement in real time based on external conditions or errors.

Scaling Difficulties: While the model is suitable for basic simulations, scaling to more complex scenarios and conditions may require significant refinement and optimization.

Conclusion: Overall, developing an algorithm and program to control the motion of a quadcopter in the vertical plane using Unity is an important and useful task. It provides a good basis for further development and improvement of UAV control algorithms. However, it is worth considering certain shortcomings and limitations of the model, as well as the possibilities of its further optimization and improvement to obtain more realistic and accurate results.

Conclusions

The development of remote control systems for quadcopters is an extremely important and relevant topic that includes various aspects such as reliability, safety, efficiency, mathematical modeling and software implementation. Successful implementation of UAV control systems contributes to their effective use in various fields and ensures a high level of autonomy.

1. Mathematical model of the movement of the quadcopter: the presented model is the basis for understanding the movement of the quadcopter and the development of control algorithms. It allows you to create stabilization and navigation systems that ensure safe and efficient flight.

2. Disadvantages of the model: Simplification of many aspects, such as aerodynamic effects and wind loads, can lead to deviations from the real behavior of the apparatus. The use of linear approximations and numerical methods to solve differential equations are additional limitations.

3. Algorithms and programs for controlling the movement of a quadcopter: The development of an algorithm and program for controlling the movement of a quadcopter in vertical and horizontal planes based on a mathematical model and its implementation in Unity 2D is extremely relevant. This allows you to create complex and efficient algorithms for various tasks, as well as use these models for educational purposes and testing.

4. Unity model benefits: Using Unity for modeling provides a user-friendly interface and tools for development and testing. Algorithms take into account the main forces and moments that allow you to create a realistic quadcopter control model.

5. Disadvantages of the Unity model: Simplifying the model can lead to deviations from the real behavior of the device. The limitations of numerical integration methods and the lack of feedback mechanisms can also affect the accuracy and efficiency of control.

The presented research and development points to the importance and relevance of quadcopter remote control systems. Mathematical models and software implementations, such as those in Unity 2D, provide the necessary tools to develop efficient and robust control algorithms. Despite some limitations and shortcomings, these studies provide a solid foundation for the further development of UAV technologies, ensuring accuracy, safety and autonomy in performing a variety of tasks. To achieve more realistic and accurate results, it is necessary to take into account additional factors and improve existing models, which will contribute to further progress in this field.

LITERATURE

[1] Remote control system of movement of pilotless aircraft. Inventor - Aleksandr Vasilyevich Koval, Denis Mykhailovych Matychyk, Nikolay Fedorovych Tupytsyn //Patent No. 81370 Ukraine, MKY B64C 13/00. – No. u201300897; Application 25.01.13; Publ. 25.06.13; Bul. No. 12. - 4 s. https://patents.google.com/patent/UA81370U/en?oq=%D0%9F%D0%B0%D1%82%D0%B5%D0%BD%D1%82+UA+81370+U

[2] Method for remote-controlling an unmanned aerial vehicle. //Patent of the invention USA US6377875B1 Designed by Inventor: Hans-Jürgen Schwerzler https://patents.google.com/patent/US6377875B1/en?oq=%D0%9F%D0%B0%D1%82%D0%B5%D0%BD%D1%82+%D0%A1%D0%A8%D0%90+6377875+%D0%BE%D1%82+23.04.2002%D0%B3

[3] Flight technical control management for an unmanned aerial vehicle. US Patent 20110035149A1 by Inventors: Kathryn McAndrew, Linda J. Watson, John Feist, A. Christopher Moorman. https://patents.google.com/patent/US20110035149A1/en?oq=US+2011%2f0035149+A1

[4] Generic drone control system. US Patent US5240207A by Inventors: David M. Eiband, Lynn R. Kern https://patents.google.com/patent/US5240207A/en?oq=%D0%9F%D0%B0%D1%82%D0%B5%D0%BD%D1%82+%D0%A1%D0%A8%D0%90+6377875+%D0%BE%D1%82+23.04.2002%D0%B3

[5] Chovancova A., Fico T., Chovanec L., Hubinsk y P. Mathematical Modelling and Parameter Identification of Quadrotor (a survey) // Procedia Engineering. 2014. Vol. 96. P. 172{181. DOI: 10.1016/j.proeng.2014.12.139

[6] Rinaldi F., Chiesa S., Quagliotti F. Linear Quadratic Control for Quadrotors UAVs Dynamics and Formation Flight // Journal of Intelligent & Robotic Systems. 2013. Vol. 70, no. 1-4. P. 203{220. DOI: 10.1007/s10846-012-9708-3

[7] Bouadi H., Simoes Cunha S., Drouin A., Mora-Camino F. Adaptive sliding mode control for quadrotor attitude stabilization and altitude tracking // Proc. 12th IEEE

international symposium on computational intelligence and informatics (CINTI 2011). Budapest: Hungary, 2011. P. 449{455. DOI: 10.1109/CINTI.2011.6108547

[8] Tayebi A., McGilvray S. Attitude stabilization of a four-rotor aerial robot // Proc. 43rd IEEE Conf. on Decision and Control. 2004. Vol. 2. P. 1216{1221. DOI: 10.1109/CDC.2004.1430207

Appendix A (script Body.cs)

```
using UnityEngine;

public class Body : MonoBehaviour
{
 // Constants
 public static float accelOfGravity = 9.81f; // Acceleration of gravity
 public static float mass = 1.0f; // Weight of the quadcopter
 public static float momentOfInertia = 0.01f; // Moment of inertia around the axis
 public static float diameter = 0.1f; // Distance from the center to the rotor
 public static float rootrTorqCoeff = 0.0001f; // Coefficient of moment of the rotor

 // Speeds of rotation of rotors
 public static float Rotor1 = 3f;
 public static float Rotor2 = 3f;
 public static float Rotor3 = 3f;
 public static float Rotor4 = 3f;

 // Thrust of rotors
 public static float forcThrust1;
 public static float forcThrust2;
 public static float forcThrust3;
 public static float forcThrust4;



 // Internal state variables
 public static float speedY = 0f; // Velocity along the y axis (vertical velocity)
 public static float speedYDot; // Scalar product
 public static float angleVelocity = 0f; // Angular velocity step
```

```
 public static float angleVelocityDot; // Scalar product
 public static float = 0f; // Angle (tilt)
 public static float inclineDot; // Scalar product

 public static float speedX = 0; // x-axis velocity (horizontal velocity)
 public static float speedXDot; // Scalar product
 public static float rateYaw = 0; // Angular velocity (deviation from course)
 public static float rateYawDot; // Scalar product
 public static float yaw = 0; // Course angle (deviation from course)
 public static float yawDot; // Scalar product

 public static float yDot; // Scalar product
 public static float xDot; // Scalar product

 // Total thrust and moments
 public static float totalForcThrust;
 public static float oppositeThrust;
 public static float positiveThrust;
 public static float momentN;

 // Update is called once per frame
 void Update()
 {
 // Calculate thrust from each rotor
 forcThrust1 = rootrTorqCoeff * Rotor1 * Rotor1;
 forcThrust2 = rootrTorqCoeff * Rotor2 * Rotor2;
 forcThrust3 = rootrTorqCoeff * Rotor3 * Rotor3;
 forcThrust4 = rootrTorqCoeff * Rotor4 * Rotor4;;
 }
}
```

Appendix B (script Octocopter.cs)

```
using UnityEngine;
using static TMPro.SpriteAssetUtilities.TexturePacker_JsonArray;

public class Octocopter : MonoBehaviour
{
 // Get a solid named "rb1"
 public Rigidbody2D rb1;
 // Getting a new vector
 Vector2 movementY;

 // Update is called once per frame
 void Update()
 {
 // Get the axis
 movementY.y = Input.GetAxisRaw("Vertical");

 // Total thrust and moments
 Body.totalForcThrust = Body.forcThrust1 + Body.forcThrust2 + Body.forcThrust3 + Body.forcThrust4;
 // Moment created by opposite rotors
 Body.oppositeThrust = Body.diameter * (Body.forcThrust2 - Body.forcThrust4);

 }
 private void FixedUpdate()
 {
 equationOfMotionY();
 kinematicsEquationY();
 MovementY();
```

```
}
 // Movement of the object
 rb1.MovePosition(rb1.position + movementY * Body.speedYDot *
Time.fixedDeltaTime);
 Debug.Log(rb1.position);
 }
 void equationOfMotionY ()
 {

 // Equations of motion
 Body.speedYDot = (Body.totalForcThrust - Body.mass * Body.accelOfGravity) /
Body.mass;
 Body.angulVelocityDot = Body.oppositeThrust / Body.momentOfInertia;
 }

 void kinematicsEquationY ()
 {
 // Equations of kinematics
 Body.yDot = Body.speedY;
 Body.inclineDot = Body.angulVelocity;
 }

 void MovementY()
 {
 // Update the state (Euler integration)
 Body.speedY += Body.speedYDot * Time.fixedDeltaTime;
 Body.angulVelocity += Body.angulVelocityDot * Time.fixedDeltaTime;
 Body.incline += Body.inclineDot * Time.fixedDeltaTime;
 }
}
```

```
using UnityEngine;
using static UnityEditor.PlayerSettings;

public class OctocopterX : MonoBehaviour
{
 // Get a solid called "rb2"
 public Rigidbody2D rb2;

 // Getting a new vector
 Vector2 movementX;

 // Update is called once per frame
 void Update()
 {
 // Get the axis
 movementX.x = Input.GetAxisRaw("Horizontal");

 // Total thrust and moments
 Body.totalForcThrust = Body.forcThrust1 + Body.forcThrust2 + Body.forcThrust3 +
Body.forcThrust4;
 Body.oppositeThrust = Body.diameter * (Body.forcThrust2 - Body.forcThrust4);
 Body.positiveThrust = Body.diameter * (Body.forcThrust1 - Body.forcThrust3);
 Body.momentN = Body.rootrTorqCoeff * ((Body.forcThrust1 * Body.forcThrust1) -
(Body.forcThrust2 * Body.forcThrust2) + (Body.forcThrust3 * Body.forcThrust3)
 - (Body.forcThrust4 * Body.forcThrust4);
 }

 void FixedUpdate()
```

```
 {
 equationOfMotionX();
 kinematicsEquationX();
 MovementX();
 rb2.MovePosition(rb2.position + movementX * Body.speedXDot *
Time.fixedDeltaTime);
 Debug.Log(rb2.position);
 }

 void equationOfMotionX()
 {
 // Equations of motion
 Body.speedXDot = -Body.mass * Mathf.Cos(Body.yaw);
 Body.speedYDot = Body.accelOfGravity * Mathf.Cos(Body.yaw) *
Mathf.Sin(Body.yaw);
 Body.rateYaw = Body.momentN / Body.momentOfInertia;
 }

 void kinematicsEquationX()
 {
 // Equations of kinematics
 Body.xDot = Body.speedX * Mathf.Cos(Body.yaw) - Body.speedY *
Mathf.Sin(Body.yaw);
 Body.yDot = Body.speedX * Mathf.Sin(Body.yaw) + Body.speedY *
Mathf.Cos(Body.yaw) ;
 Body.yawDot = Body.rateYaw;
 }

 void MovementX()
 {
```

```
 // Update the state (Euler integration)
 Body.speedX += Body.speedXDot * Time.fixedDeltaTime;
 Body.speedY += Body.speedYDot * Time.fixedDeltaTime;
 Body.rateYaw += Body.rateYawDot * Time.fixedDeltaTime;
 Body.yaw += Body.yawDot * Time.fixedDeltaTime;
 }
}
```