Group Flight Automation Using Tello EDU Unmanned Aerial Vehicle

Olha Pohudina
department of Information Technology of Design
National Aerospace University «KhAI»
Kharkiv, Ukraine
0000-0001-5689-2552

Mykhailo Kovalevskyi
department of Information Technology of Design
National Aerospace University «KhAI»
Kharkiv, Ukraine
0000-0001-8311-2304

Mariia Pyvovar
department of Information *Technology of Design*
National Aerospace University «KhAI»
Kharkiv, Ukraine
0000-0002-2136-233X

Abstract – UAVs are expected to play a key role in the future, providing support to various services to the population. Already today, their use helps to solve a huge range of scientific and applied problems related to ecology, meteorology, geology, zoology, agriculture, climate study, mineral exploration and more. Increasing the number of civilian, military and sports UAVs significantly complicates the problem of flight safety in a single airspace. The factors that have the strongest impact on the safety and effectiveness of UAVs include: destruction in contact with the ground; harm to life and health of people or their property on the ground; collisions in the air with other aircrafts. That is why the study of group flight parameters, especially in automatic and semi-automatic mode is an actual task. The problem of group flight automation based on the use of the popular Tello EDU UAV training model is solved in the work. The architecture, algorithm and model of information exchange in flight control are formed. UAV flight testing was performed and additional recommendations for further organization of experiments in the group were developed.

Keywords – unmanned aerial vehicles, group flight, automatic flight, software architecture, Tello EDU

I. INTRODUCTION

Today, for unmanned aerial vehicles (UAVs), such as Tello [1], Parrot AR.Drone [2] and Crazyflies [3], there are a large number of information systems that can automate their behavior in the environment [4-6]. As a result, such software packages are able to create subroutines to control individual UAVs. However, to control the UAV group, it is necessary to solve completely new tasks that existing automation packages cannot solve, including: calculation of the physical space required for the operation of a separate UAV, search for a communication channel for a separate UAV, the need to maintain a safe distance from other moving objects.

UAV groups are considered in studies [7-8] but in fact, even with software packages that can support multiple UAVs, the documentation focuses on a single UAV. There are studies of specialized institutions [9-10], but they are based on expensive or commercially unavailable solutions.

There are open solutions for group flight research, such as Bitcraze Crazyflie 2.0 [3, 11], but the limitation is the external position tracking system and hardware supply problems for our region.

Among the existing UAVs that can be used to create a group flight mockup there is Tello EDU from DJI. Despite the fact that this UAV is a closed and commercial project, there is a library that provides the ability to program a group flight [12-13]. In addition, Tello EDU UAVs are quite inexpensive, and their design ensures safe use in laboratory premises under conditions of enough lighting and other not so high requirements.

Therefore, the paper proposes to explore the capabilities of the Tello EDU UAV for various flight purposes [14-15] and to form the parameters of the mockup of software and hardware for the organization of a group flight.

II. FORMULATION OF THE PROBLEM

A. Overview of the Tello EDU UAV Model

Ryze Tello is a high-tech UAV with Intel's 14-core processor, DJI flight control system, visual positioning system, auto take-off and landing function, altitude holding function, five programmable flight modes, built-in 5 MP HD camera, electronic image stabilization (EIS), 720P HD first-person flight (FPV), recording of short videos, maximum flight distance – 100 m, maximum flight time – 13 minutes, as well as the ability to program UAVs using the built-in library [16].

Restrictions on use include: collector motors (ie low efficiency, low speed, overheating, rapid wear), no storage for video and photo materials (recorded material is stored on the resources of the connected smartphone or computer).

The main advantage is that Tello EDU contains built-in tools that allow you to program the UAV and its own flight, as well as make an organized flight in a group with other Tello EDU UAVs.

The Tello SDK is a programming interface that allows you to automate the flight of one or a group of Tello EDU UAVs. The Tello SDK allows you to create applications for a variety of operating systems and platforms. It contains commands that can be called from Python, Scratch or Swift. These commands provide direct access to the UAV.

The Tello SDK includes three basic command types:

- control commands: returns «ok» if the command was successful; returns «error» or an informational result code if the command failed;

- set commands: returns «ok» if the command was successful; returns «error» or an informational result code if the command failed;

- read commands: returns the current value of the sub-parameter.

B. Software Architecture for Automatic Flight of the UAV Group

Own program contains four files:

- command.txt – contains text commands to be sent to the UAVs;
- stats.py – contains the definition and implementation of the Stats class required to collect statistics;
- tello.py – contains the definition and implementation of the Tello class required to communicate with the UAVs;
- main.py – contains the main program that reads and sends commands and receives responses from the UAVs.

Thus, the program contains two classes: Stats; Tello. Both classes are derived from the object class, as by default all classes in Python are inherited from the object class.

The Stats class diagram is shown in Fig. 1. The Stats class has 6 fields:

- command – stores the text of the sent command;
- id – command ID;
- start_time – start time of sending the command to Tello EDU;
- end_time – end time after receiving a response from Tello EDU;
- duration – total time of executing the command;
- response – stores the text of the response received after sending the command.

Also the Stats class has 6 methods. Stats class methods:

- __init__(self, command, id) – class constructor with parameters. Initializes all fields of the class with initial values.

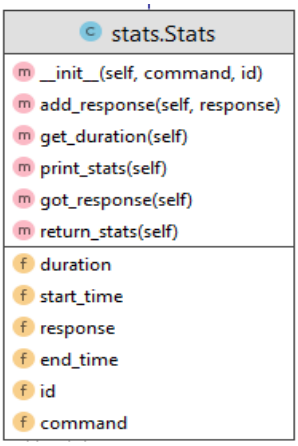

Fig. 1. The Stats class diagram

- get_duration(self) – getter for the duration field. Calculates and returns time to execute a command.
- add_response(self, response) – gets the response and writes it in the corresponding field.
- got_response(self) – checks whether a response has been received from the UAV.
- print_stats(self) – displays all statistics with the values of all fields.
- return_stats(self) – returns all statistics with the values of all fields as a string.

The Tello class diagram is shown in Fig. 2.

The Tello class has 10 fields:

- local_ip – stores the IP address of the local computer;
- local_port – stores the port of the local computer;
- log – stores statistics as a list of values;
- receive_thread – thread to receive a response from Tello EDU;
- MAX_TIME_OUT – maximum waiting time for the command;
- response – stores the text of the response received after sending the command;
- tello_address –pair of values (IP address, port);
- tello_ip – stores the IP address of the Tello EDU;
- tello_port – stores the port of the Tello EDU;
- socket – socket needed to create a UDP connection.

The Tello class has the following 5 methods:

- __init__(self) – class constructor. Initializes all fields of the class with initial values.
- send_command(self, command) – sends the command to the specified address. If there is an error sending the command – tries to send it again.
- _receive_thread(self) – private method that waits for a response and writes it to the corresponding field. Runs as a background thread.
- get_log(self) – getter for the log field. Returns statistics as a list of values.
- on_close(self) – describes the actions required when closing a UDP connection.

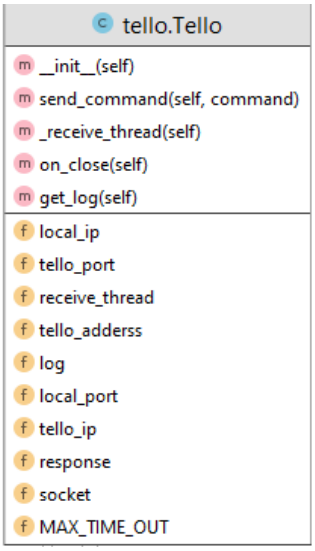

Fig. 2. The Tello class diagram

C. UAV Communication With the Application

In order for the UAV and the application to be able to communicate with each other, all of the Tello SDK commands must be wrapped into Python functions. After that, special commands for UAV control become available and can be used with any algorithmic constructions (cycles, if-else, etc.) and data structures (lists, dictionaries, etc.) of the programming language. The general communication scheme of one UAV Tello EDU with the application is shown in Fig. 3.

Consider the communication of the Tello EDU UAV group with a computer. It is impossible to directly connect all UAVs to a PC. It is necessary to use an auxiliary link that will connect the group and the application. Such a link is a router. The program will send commands to the router, which, in turn, will send these commands to all UAVs in the swarm. The answers will be sent in the same way, only in reverse order – from UAVs to the computer via a router (Fig. 4).

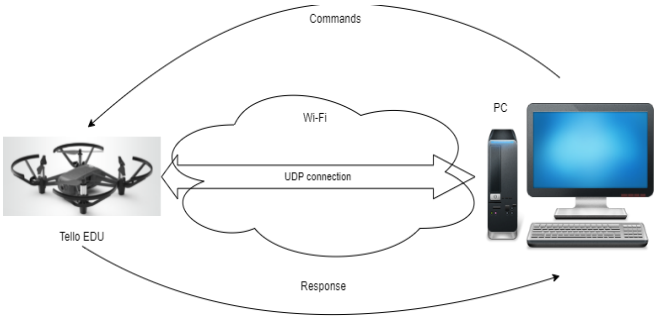

Fig. 3. General communication scheme of one UAV Tello EDU with a PC

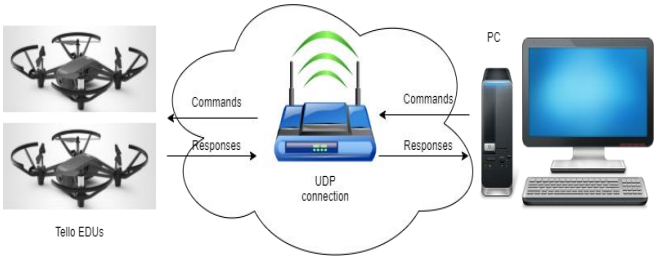

Fig. 4. General communication scheme of a swarm of UAVs Tello EDU with a PC

III. ALGORITHMIZATION OF UAV GROUP FLIGHT

Consider a general flight algorithm of a group of UAVs.

The first step is to import modules: socket, threading, time. Modules are needed for: UAV identification in the network, background thread processing, timer operation.

Next, we assign each UAV to a separate IP address (identify and initialize variables) and PC ports to transmit information to and from the UAV. Define variables for sockets.

The next step is to bind to the local address and port.

Then define a function that will send commands to all UAVs in the group and display information about the status of sending the command.

Next, define the function that will receive responses from the UAV.

The block of definitions is completed, ie the basic algorithm begins (fig. 5).

Step 1. Create and start a listening thread that runs in the background. This utilizes our receive functions and will continuously monitors the network for incoming messages from UAVs.

Step 2. Defining variables for setting the flight path. In the current algorithm, this is the box movement of the UAV.

Step 3. Put the Tello EDU into a command mode.

Step 4. All UAVs take off.

Step 5. Loop for flight by each leg of the box.

Step 6. All UAVs land and notification about successful mission completion is displayed.

This program was tested on two UAVs. Then the program was modified for two more UAVs (Fig. 6). In addition, other UAV movements from the Tello SDK were used, namely:

- up x – ascend to «x» cm;
- down x – descend to «x» cm;
- left x – fly left for «x» cm;
- right x – fly right for «x» cm;
- forward x – fly forward for «x» cm;
- back x – fly backward for «x» cm;
- cw x – rotate «x» degrees clockwise;
- ccw x – rotate «x» degrees counterclockwise;
- flip x – flip in «x» direction.

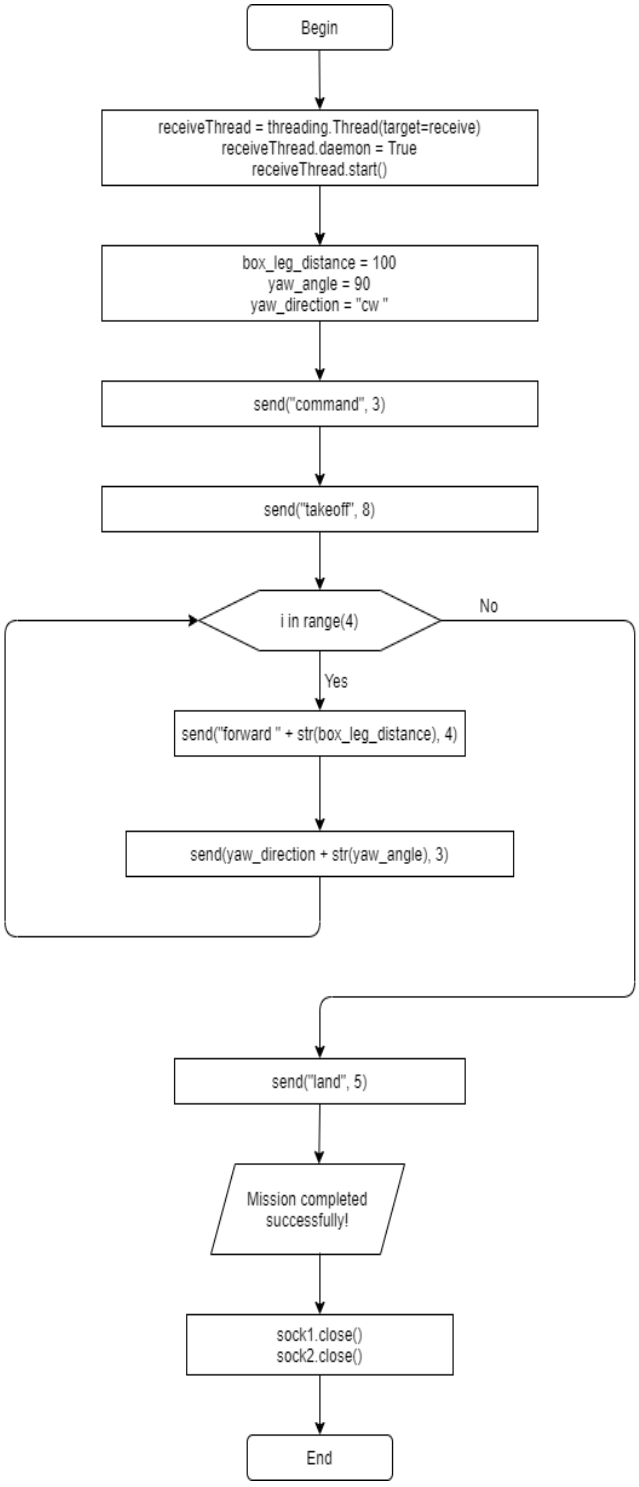

Fig. 5. Block diagram of the algorithm

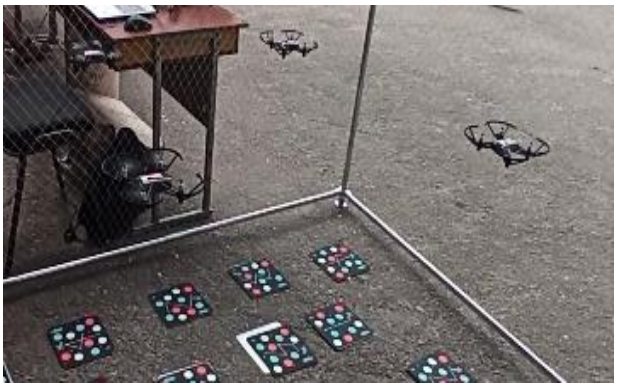

Fig. 6. An example of an experiment with four UAVs

As a result of a series of tests in the group of UAVs, the following was found: first, lighting, battery charge and surface under the aircraft much affect the accuracy of UAV positioning during flight; secondly, there are some types of movement that lead to greater positioning error than others. The following errors were revealed: forward and backward movement gives less error (1-10 cm) than right and left movement (5-31 cm). When performing more complex movements (such as a flip), the error reached up to 63 cm.

A series of experiments was performed: movement according to the same flight program of all UAVs, movement according to different UAV programs, movement according to external marks, movement of the group with additional load in the form of LED illumination. To conduct experiments on the model and during training flights, additional software was used, which allows: to display the flight trajectory, generates a code for certain actions of the UAV. For individual UAVs, it is possible to perform command movements, which are set by the user with hand movements. In the future, it is planned to organize the formation of a flock of UAVs that can move behind a certain leader (a separate UAV), while maintaining a safe distance of the UAV[17]. However, the next after the Tello EDU UAV with DJI, which recently went on sale (Robomaster TT), will be used for this purpose. This UAV contains an integrated infrared distance sensor ToF and more advanced programming functions.

IV. Conclusion

When performing this work, the UAV Tello EDU was considered and the choice of this UAV for the organization of a group flight was justified.

Among the main advantages is the fact that the Tello EDU has the ability to be programmed. It can be programmed in Python, Swift and Scratch using a special command system (the Tello SDK connects to the UAV through Wi-Fi UDP port, allowing users to control the UAV with text commands).

That is, the Tello SDK and the Python programming language were used to automate the flight. A program has been developed for the UAV group to perform the flight. Software architecture classes are described for the proposed program. The model of information exchange between UAV and PC in case of communication of one and several UAVs is considered. Several algorithms with the implementation of different flight tasks are proposed. A series of indoor experiments was performed. Positioning errors are determined. As a result, the following recommendations were implemented to automate the flight of a UAV group based on the use of Tello EDU: the distance between the UAVs in a group flight should exceed 50 cm, and in flips – 100 cm; the duration of the flight in case of flips will be reduced (maximum, you can make 4 flips with a standard LiPo battery with a capacity of 1100 mAh).

The work is performed as a part of the applied study “Methods of swarm intelligence control for the effective use of unmanned aerial vehicles for civil and military use”.

References

[1] M. Karahan, H. Kurt and C. Kasnakoglu, “Autonomous Face Detection and Tracking Using Quadrotor UAV,” *IEEE 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 2020, pp. 1-4.

[2] H. Bouafif, F. Kamoun and F. Iqbal, “Towards a Better Understanding of Drone Forensics,” *International Journal of Digital Crime and Forensics*, 2020, no. 12(1), P. 35–57; doi:10.4018/ijdcf.2020010103.

[3] J. A. Preiss, W. Hönig, G. S. Sukhatme and N. Ayanian, “Crazyswarm: a large nano-quadcopter swarm,” *Proceedings of the International Conference on Robotics and Automation*, 2017, pp. 3299–3304. [Online]. Available: https://github.com/USC-ACTLab/crazyswarm.

[4] A. Cyba, H. Szolc and T. Kryjak, “A simple vision-based navigation and control strategy for autonomous drone racing,” arXiv preprint, 2021, [Online]. Available: arXiv.org/2104.09815.

[5] C. Caceres et al., “Simulation, model and control of a quadcopter AR drone 2.0,” *International review of mechanical engineering*, 2016, P. 432.

[6] W. Hönig and N. Ayanian, “Flying multiple UAVs using ROS,” *Robot Operating System (ROS)*, Springer, Cham, 2017, pp. 83-118.

[7] M. Schranz et al., “Swarm robotic behaviors and current applications,” *Frontiers in Robotics and AI*, 2020, no. 7., P. 36.

[8] M. Campion, P. Ranganathan and S. Faruque, “UAV swarm communication and control architectures: a review,” *Journal of Unmanned Vehicle Systems*, 2018, vol. 7, no. 2, pp. 93-106.

[9] M. Kasprzyk et al., “Application of a small UAV fleet for demonstration of optimized mission,” *AIAA Flight Testing Conference*, 2015, P. 3349.

[10] A.I. Hentati et al., “Simulation tools, environments and frameworks for UAV systems performance analysis,” *14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, IEEE, 2018, pp. 1495-1500.

[11] W. Giernacki et al., “Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering,” *22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, IEEE, 2017, pp. 37-42.

[12] M. V. Mamchenko, “Analysis of Control Channel Cybersecurity of the Consumer-Grade UAV by the Example of DJI Tello,” *Journal of Physics: Conference Serie*s, IOP Publishing, 2021, vol. 1864, no. 1, P. 12127.

[13] T. Saitoh et al., “Real-time breath recognition by movies from a small drone landing on victim’s bodies,” *Scientific reports*, 2021, vol. 11., no. 1, pp. 1-7.

[14] V. Kharchenko et al., “Green computing and communications in critical application domains: Challenges and solutions,” *The International Conference on Digital Technologies 2013*, IEEE, 2013, pp. 191-197.

[15] M. Uss et al., “Image informative maps for estimating noise standard deviation and texture parameters,” *EURASIP Journal on Advances in Signal Processing*, 2011, vol. 2011, pp. 1-12.

[16] DJI Ryze Tello UAV User Manual 2018. [Online]. Available: https://dl-cdn.ryzerobotics.com/downloads/Tello/20180404/Tello_User_Manual_V1.2_EN.pdf.

[17] O. Pohudina, D. Kritskiy, S. Koba and A. Pohudin, “Assessing Unmanned Traffic Bandwidth,” *Advances in Intelligent Systems and Computing*, Springer, Cham, 2020, pp. 447–458; doi: 10.1007/978-3-030-37618-5_38.