UDC 004.042

**I. Akhaladze, O. Lisovychenko**

# USE OF SERVERLESS FUNCTIONS IN THE ALGORITHM FOR CALCULATING THE TARGET POINT OF THE TRAJECTORY UNDER DYNAMIC LOADING

*Abstract:* Implemented architecture using server-free functions for the analysis of the exit to the target point of the trajectory of a swarm member in conditions of dynamic loading with lower energy consumption. Using the approach, implementing the mathematical apparatus of calculating the target point and validating the output at this point, in terms of the task allows more efficient use of energy at the synchronization node by 37.5%, and since the calculations are transferred to the cloud, costs in other parts energy consumption for data transmission. This allows you to significantly continue the work of the swarm in conditions of limited energy resources and increase the flight time of the swarm.

*Keywords:* trajectory endpoint calculation, downtime, calculation call discreteness, serverless functions, power consumption, SLA

## Introduction

Calculating the trajectory target point to synchronize more than one drone combined in a swarm is a task that is not difficult in terms of the cyclomatic complexity of the algorithm, but this task is subject to high requirements for speed and energy consumption.

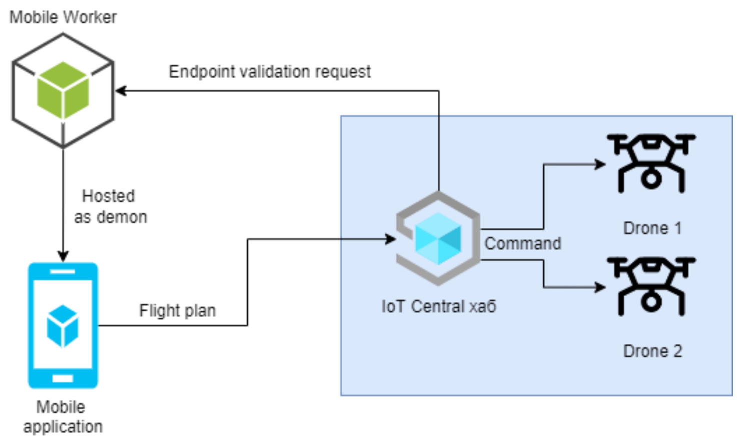

*Figure 1.* Architectural scheme of the system of synchronization of drone trajectories with calculation on the side of the mobile application

The call to validate the drone's exit to the end point of the trajectory is sent to a mobile application that performs a mathematical calculation (Fig. 1). The frequency of the calculation call is equal to the number of synchronization points of the flight plan multiplied by the number of swarm members and requires continuous operation of the daemon on the side of the mobile application, which leads to 30% of energy consumption of the mobile device. Since the number of calls to the calculation method depends on the complexity of the flight plan and the number of members of the swarm, the statistics of the calculation will be very defragmented. That is, to analyze the current state of each drone in the swarm with a certain time discreteness, you need a constant cost of energy to ensure the availability of analysis service at any time. The problem that requires a solution is the high energy consumption at the decision node (mobile application) for the support and hosting of the service, which encapsulates the logic of calculating the end point of the trajectory, and the elimination of energy costs when idle this service [3]. Transferring these calculations to the cloud will reduce energy consumption at the decision node for calculations and eliminates the cost of downtime [2]. Energy losses of the node decisions will be made only for data transportation, and the mathematical apparatus itself will work as a cloud service. In addition, the use of serverless functions and their implementation through trajectory-type decomposition, where the calculation of the endpoint of elementary trajectory sections is performed by each individual serverless function, can reduce code connectivity, simplify maintenance, and use caching on functions. function with the same parameters several times, instead of executing the result will be calculated only once.

## Problem statement

It is necessary to propose an approach to reduce energy consumption at the decision-making node to calculate the target point of the trajectory and eliminate energy consumption when idle service that implements the mathematical apparatus of calculations.

## Solving the tasks

The proposed approach to the use of serverless functions [5] to implement the validation of the output to the target point of the trajectory based on the analysis of telemetry data at a discrete time is shown in Fig. 2.

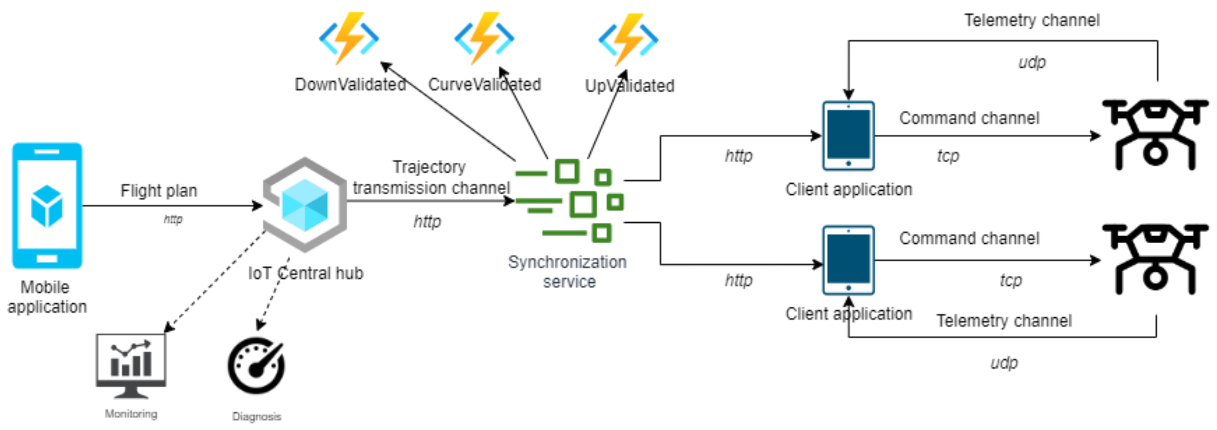

*Figure 2.* Architectural scheme of the system of synchronization of drone trajectories

The mobile application serves as a trigger to start testing two drones of the synchronous flight plan. The IoT Hub performs the function of routing and collecting logs for monitoring and post-flight diagnostics. The synchronization service receives the flight plan, divides it into simple trajectories, and creates synchronization points on their borders. In addition, the service with a given discreteness receives the telemetry data of each member of the swarm and analyzing them, validates the output to a given point of the trajectory. Trajectory endpoint calculation tasks perform server-free functions. The client application transforms the trajectories broken by the synchronization service into drone commands of a certain model. Since the API of interaction with a drone of a certain model is specific, for each model of a drone with a certain API it is necessary to implement a separate isolated instance of the application [4]. In addition, the application acts as a proxy and sends to the synchronization service telemetry data in real time for analysis [6].

To test the system, we use the following scenario: two drones are included in the swarm through the IoT hub, working out a deterministic flight plan with three trajectories simultaneously and in isolation. Rise to a height of 1m in 3 seconds, hang for 2 seconds, make a circular horizontal arc of 120 degrees in 5 seconds, hang for 4 seconds, and go down to 1m in 2 seconds.

The use of serverless functions to solve this problem is the most rational in terms of energy consumption [1]. Let's decompose the scenario described in the given task on the time indicator:

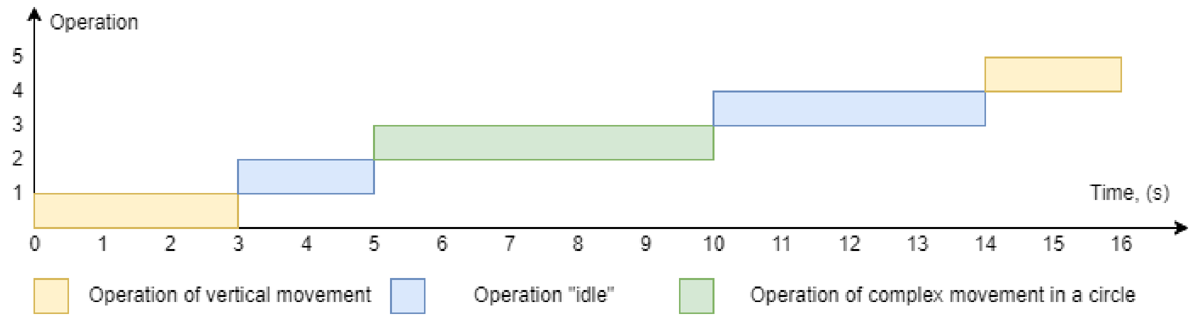

*Figure 3.* Decomposition of the flight plan over time

From fig. Figure 3 shows that the downtime of the flight plan is the time of irrational use of energy at the decision-making node. The multiplicity of energy consumption is directly proportional to the current number of drones in the swarm. The other three trajectories change the position of the drone in space, so you need to validate the exit to a given point at each discrete point in time. To do this, use the telemetry data of each drone at each such time. The data analysis will then consist of data on the initial state of the drone (its position in the case of simple trajectories), executed by the command with its parameters, and the current state of telemetry in text format. The logic of the analysis will work according to the following algorithm:

1. Determining the end point of the trajectory based on the initial state of telemetry and the executed command
2. Comparison of the current state of drone telemetry and calculated telemetry at the end point of the trajectory

Since the complexity of the trajectories is limited only by the capabilities of the drone, the load on the mathematical apparatus of telemetry data analysis will be defragmented. For this example, 3 serverless functions are implemented:

1. UpValidated
2. CurveValidated
3. DownValidated

Also, the selected discreteness of access to these functions in 1 second. The graph of the distribution of load time on serverless functions is as follows:

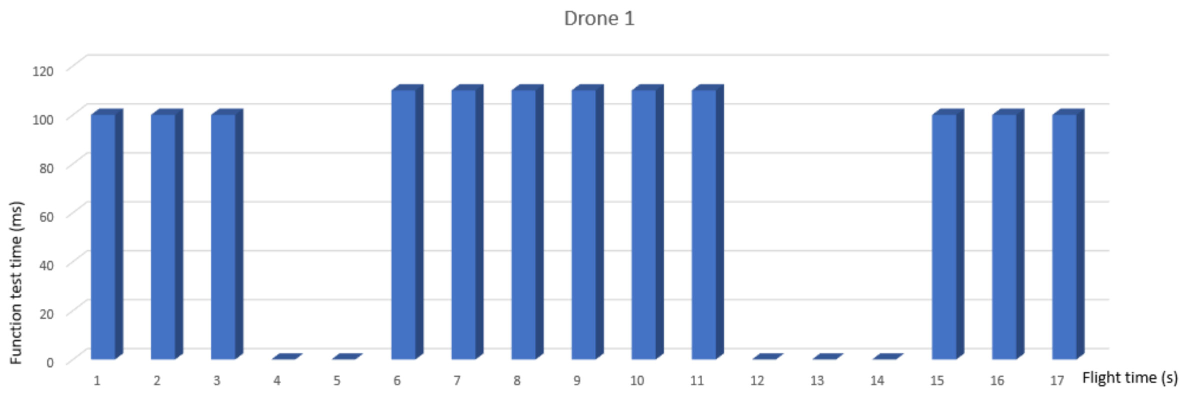

*Figure 4.* Distribution of the call of analysis functions by one drone

Histogram visualization in Fig. 4 shows the process of no calls to the serverless function from 3 to 6 and from 11 to 15 seconds. That is, the percentage of downtime when working out a given trajectory will be:

$$\eta = \frac{T}{} = \frac{6}{16} * 100\% = 37.5\ \% \qquad (1)$$

where,

$T$ - the total downtime of the billing service

$T$ - flight plan execution time

Having obtained the result from formula 1, we see that when performing a given trajectory, 37.5% of the time the service does not perform calculations, and therefore, this is the percentage of energy that is not used in the calculation due to serverless functions. In addition, because the calculations are performed in the cloud, the energy costs for the calculations of the trajectory endpoint at the decision node are reduced to zero, the real energy costs will be equal only to the costs for data transmission.

Simulating the operation of a swarm of 9 drones, we wash the following distribution of the frequency of calls to serverless functions (Fig. 3):

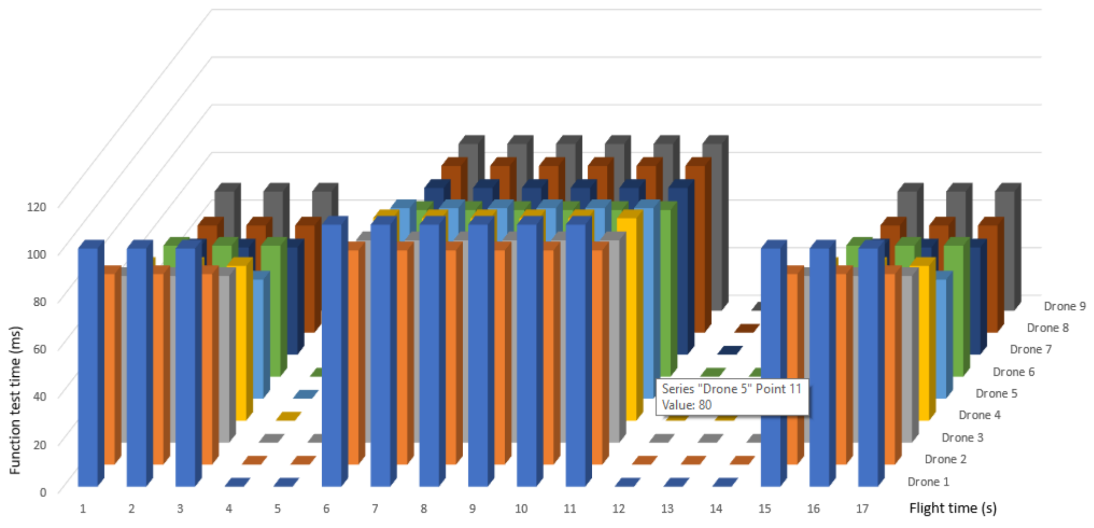

*Figure 5.* Distribution of calls server calls by a swarm of nine drones

In fig. Figure 5 shows that the load is fragmented in the presence of nine drones in the swarm and a simple set of trajectories, and the complexity of the trajectories will only increase the nonlinearity of the load. In such conditions, it is impossible to predict the requirements for computing power required for the analysis of telemetry data and energy consumption, so the use of serverless functions is the most rational.

There is a requirement for speed to the functions themselves [8], which must work out and return the result as quickly as possible (the time of work of the function should not exceed the set discreteness of the analysis of telemetry data). Studies of the maximum allowable time of serverless functions for the analysis of telemetry will be carried out in the following works in the conditions of entering the swarm of up to 9 drones.

Implementing feedback using server-free functions not only allows you to granulate mathematical operations of analysis [7], and more efficient use of energy resources but also opens the possibility of implementing more complex analysis operations, such as video streaming. The possibility of using video stream analysis mechanisms will be explored in the next paper.

## Conclusions

The need to validate the exit to the end point of the trajectory of each drone leads to the cost of energy of the mobile application, which is spent on hosting a copy of the worker who performs calculations. Transferring these calculations to the cloud allows you to eliminate these costs in case of downtime and reduce up to 5% of the data spent on data transmission, and all calculations are performed in the cloud.

The use of serverless functions to implement the mechanism of calculating the target point of the trajectory and validation of the output of each member of the swarm to the target point is optimal in terms of energy consumption under dynamic load. When solving the