

Efficient model of PID controller of unmanned aerial vehicle

Arsen R. Petrosian¹, Ruslan V. Petrosian¹, Ihor A. Pilkevych² and Maryna S. Graf¹

¹Zhytomyr Polytechnic State University, 103 Chudnivsyka Str., Zhytomyr, 10005, Ukraine

²Korolyov Zhytomyr Military Institute, 22 Myru Ave., Zhytomyr, 10004, Ukraine

Abstract. The widespread use of digital signal processing distinguishes the modern stage of science and technology development. The use of digital processing for effective control of aircraft and other technical equipment remains relevant for many industries. The paper considers the problem of building a model of a digital PID regulator that can be used in unmanned aerial vehicles. It is proposed to base the regulator on the methods of digital filtering (realization of the differential component using a digital filter with a finite impulse response). The calculation of digital filter coefficients is performed using a genetic algorithm. This approach increases the model's accuracy in calculating PID-regulator coefficients using classical PID-regulator methods. The Python programming language for realizing the proposed method has been developed. The effectiveness of the developed model is demonstrated in simulation.

Keywords: UAV, PID controller, model optimisation algorithm, digital filter, genetic algorithm

1. Introduction

The task of effective control of technological processes, robotic systems, aircraft and other technical means remains relevant for many industries. Unmanned Aerial Vehicles (UAVs) or drones are gaining popularity, especially recently. They are autonomous or semi-autonomous aircraft without a pilot on board, which can perform various tasks.

UAVs have a wide range of applications. They are used for commercial purposes, such as: aerial photography, mapping, monitoring industrial and natural complexes, delivering goods and much more. Drones are also actively used for scientific and research purposes, in agriculture, in the military sphere of activity, to quickly obtain information about the consequences of emergencies, as well as for entertainment purposes.

The advantages of UAVs are their manoeuvrability, ability to operate in hard-to-reach places, low cost of operation compared to manned aircraft, and ability to solve tasks without threat to human life and health.

Overall, UAVs have great potential, and it is expected that their popularity will continue to increase in the future. Moreover, consequently, the drone flight environment is becoming increasingly complex. For example, it may involve various moving obstacles.

✉ xckaytz@gmail.com (A. R. Petrosian); e-rvs@ukr.net (R. V. Petrosian); igor.pilkevich@meta.ua (I. A. Pilkevych); graf.maryna@gmail.com (M. S. Graf)

ORCID [0000-0003-0960-8461](https://orcid.org/0000-0003-0960-8461) (A. R. Petrosian); [0000-0002-0388-8821](https://orcid.org/0000-0002-0388-8821) (R. V. Petrosian); [0000-0001-5064-3272](https://orcid.org/0000-0001-5064-3272)

(I. A. Pilkevych); [0000-0003-4873-548X](https://orcid.org/0000-0003-4873-548X) (M. S. Graf)



© Copyright for this paper by its authors, published by Academy of Cognitive and Natural Sciences (ACNS). This is an Open Access article distributed under the terms of the Creative Commons License Attribution 4.0 International (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

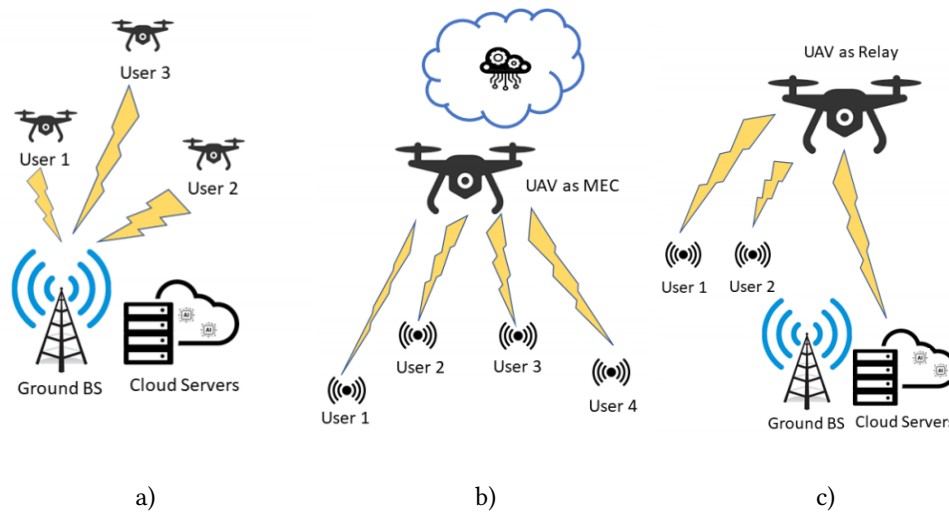


Figure 1: UAV-enabled edge computing architecture: a) The scenario UAVs have computation task; b) The scenario UAVs as a MEC; c) The scenario UAVs as a Relay.

A microcontroller plays the role of the central controlling device in a UAV. It is responsible for control and data processing. UAV control can be conditionally divided into three components: route planning, navigation, flight stabilisation, and data processing into two components: data of the performed task (photo, video, etc.), data for navigation and stabilisation of the UAV (accelerometer, gyroscope, GNSS, etc.). Microcontrollers usually have limited resources compared to more powerful computers. They have limited processing power and RAM for data storage. This limits the complexity of algorithms that can be executed on board unmanned aircraft and the processing of large amounts of data.

The use of cloud computing and cloud storage greatly enhances the UAV control and data processing capabilities. However, there are limitations in using cloud computing, such is the delay in UAV data transmission. In this regard, edge computing and efficient UAV control algorithms have become important.

2. Architecture overview

There are several scenarios for the application of UAVs in an edge-computing architecture (figure 1) [26].

The first scenario (figure 1a) is the most common. In this case, the UAV solves the task of a specific user, and edge computing and cloud computing are used to process the data obtained using the drone. This scenario is used when the UAV has insufficient RAM and computational resources to accomplish the task. Also, moving the computation to the servers reduces the battery power consumption, which increases the drone's flight time. The second (figure 1b) and third (figure 1c) scenario is used to organise the computation of ground-based users. In the

second scenario, the UAV raises the MEC (mobile edge computing) server to a certain altitude. In the third scenario, the UAV serves as a central repeater to transmit data to the MEC servers.

Edge computing is a distributed computing paradigm that brings computation and data storage closer to the data source. This reduces response time and network bandwidth.

Edge computing infrastructure is placed between cloud computing and edge devices (M2M, IoT, CPS, etc.). There are many different types of edge devices that utilise edge computing, so the architecture of edge computing may vary slightly. However, in general, it can be visualised as follows (figure 2) [16].

According to this figure, it can be seen that the farther the computation occurs from the device, the longer the response time of the system.

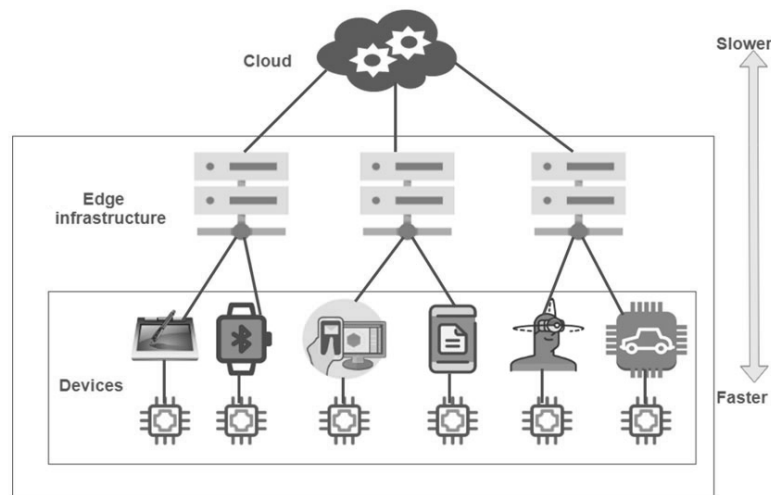


Figure 2: Edge computing architecture.

Modern flight controllers based on PX4 [21] and ArduPilot [1] firmware allow you to organise information exchange with external devices (ground control stations, servers, etc.). MavLink (Micro Air Vehicle Link) is used as an exchange protocol. The MavLink protocol is an open standard not bound to any specific data transmission medium.

Based on MavLink, an API is provided to create applications that can send requests to the flight controller to control the drone, get information about the flight status, capture data from sensors, and others.

Figure 3 shows an example of organising complex computations in unmanned aviation systems based on the “Pixhawk” flight controller (PX4 autopilot) and Raspberry Pi single-board computer (SBC) [21]. Thus, it can be noted that the computations required to solve a specific task are distributed between the flight controller, intermediate computer and cloud servers.

Let us briefly look at an example of using such an architecture. Many companies are responsible for the condition of power transmission networks. Inspection and maintenance of power lines are aimed at minimising the risk of blackouts; and extending the life of cables. Inspection is carried out in different ways: manual inspection; using a helicopter based on aerial photography. Recently, UAVs [24] have been used to inspect power transmission facilities. In

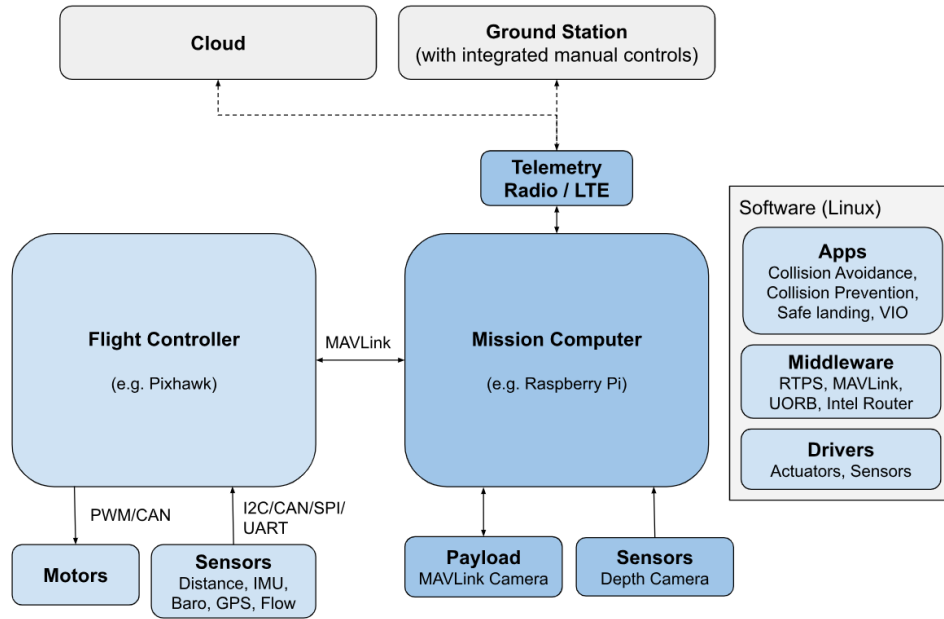


Figure 3: PX4 system architecture.

this paper, a UAV scheme based on an open-source platform is proposed to ensure the autonomy of the drone for power line inspection.

Figure 4 shows the general view of the system (numbers indicate the technologies used) [24]: (1) cloud services provide the drone with geolocation data of nearby power transmission towers, which are used for navigation; (2) acquisition and processing of sensor data to pinpoint power lines for drone alignment and cable inspection; (3) trajectory planning for cable capture; (4) following the line for cable inspection using on-board sensors. As we can see, the architecture of this system corresponds to figure 3.

When solving such tasks using UAVs, there may be non-standard situations; for example, on the way may appear broken wires of power lines; birds, etc., so the UAV must be able to avoid such obstacles [7] and the reaction time should be as short as possible. Such data processing should be done on the UAV side, and efficient control algorithms should be used. One of them is the PID controller algorithm [17, 19].

3. Theoretical background

In general, the control system for any object has the form shown in figure 5.

where $W_{pid}(p)$ – PID control transfer function, $W_o(p)$ – control object transfer function, $x(p)$, $y(p)$, $e(p)$, $u(p)$ – input action, output signal, error signal, control signal, respectively.

The time-domain control algorithm is implemented by the expression (1) [17]:

$$u(t) = K_p e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt}, \quad (1)$$

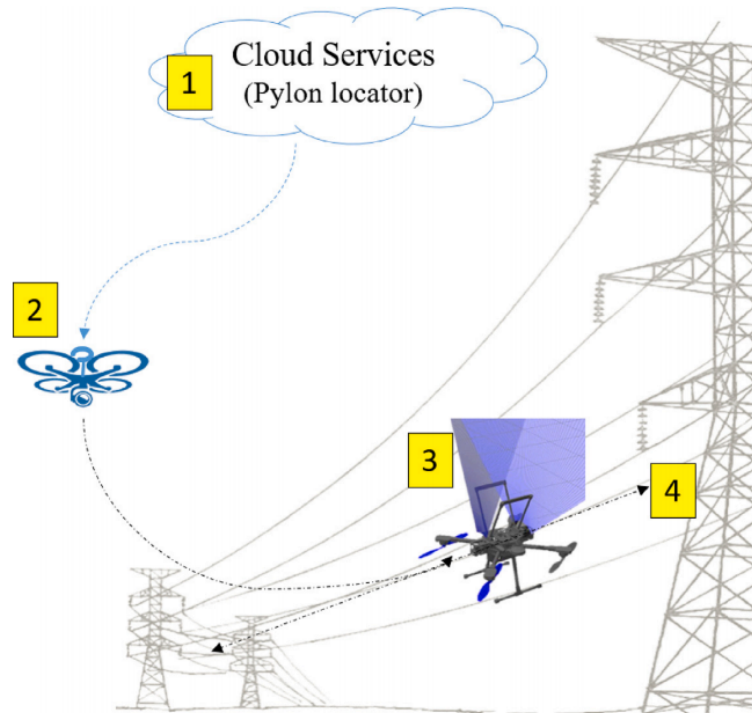


Figure 4: Conceptual overview of the proposed framework.

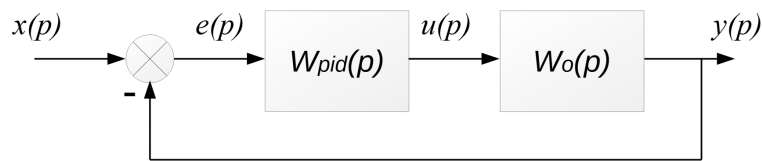


Figure 5: Facility management system.

where K_p , T_i , T_d – proportional factor, constant of integration and constant of derivation of the controller, respectively.

In some cases, the following expression is used (2):

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}, \quad (2)$$

where K_p , K_i , K_d – PID controller coefficients.

Thus, the PID controller includes three components: proportional, integral and differential. The proportional component generates a control signal counteracting the deviation (mismatch) of the output signal from the set value. The greater the mismatch, the greater the impact on the control object. If the output signal is equal to the set value, then the error signal is zero, and therefore, the control action of the proportional component is zero. The integrator is used to

eliminate the static error. The differentiating component takes into account the rate of change of the output signal, which allows you to get better control of the object by predicting the output value of the signal [17].

There are several groups for assessing the quality indicators of object management: direct, root, frequency, and integral. In practice, direct quality indicators have found the greatest application. This is because direct indicators of the quality of object management are determined directly by the transient characteristic [17]. The following quality indicators can be distinguished:

- steady-state output value;
- static error;
- regulation time;
- overshoot;
- attenuation rate;
- etc.

The choice of control quality indicators depends on the task in which the PID controller is used.

To ensure the required performance of regulation, it is necessary to calculate the coefficients of the PID controller.

There are many methods for calculating quality indicators. One of the first methods for calculating the parameters of PID controllers was proposed by Ziegler J. and Nichols N. [22]. This technique does not give excellent results, but it is elementary. Therefore, it is still often used in practice. After calculating the parameters of the regulator, manual adjustment is required to improve the quality of regulation.

In [22], other methods for calculating the parameters of PID controllers are also considered, namely Chien-Hrones-Reswick and Kuhn. Relay methods are also widely used [5, 8].

If the methods considered were developed relatively long ago, then the methods below are quite recent.

In [10, 27], methods for optimising the parameters of a PID controller using a genetic algorithm are considered. In these works, the choice is analysed: fitness functions, the main operators of the genetic algorithm, and quality indicators.

In [11], the possibility of using neural networks to optimise the PID controller coefficients was considered. The main focus of the article is on the analysis of the stability of such systems.

Many works are related to determining the optimal parameters of the PID controller for specific control objects [10, 11, 23, 25].

A large number of jobs linked to the development of the digital PID [2–4, 9, 12–14]. However, in fact, all the articles cited can be divided into two groups.

The first group of works [2, 3, 9, 14] is based on expression (3) given in [4]:

$$u(n) = K_p e(n) + K_{id} \sum_{k=0}^n e(k) + K_{dd}(e(n) - e(n-1)), \quad (3)$$

where T_k – sampling period, $K_{id} = K_i T_k$, $K_{dd} = K_p / T_k$.

Expression (3) is often written in a recurrent form to reduce computational costs (4):

$$u(n) = u(n-1) + K_p(e(n) - e(n-1)) + K_{id}e(n) + K_{dd}(e(n) - 2e(n-1) + e(n-2)). \quad (4)$$

The second group of works [12, 13] is based on the expression (5):

$$u(n) = u(n-1) + K_1e(n) + K_2e(n-1) + K_3e(n-2), \quad (5)$$

where $K_1 = K_p + K_i + K_d$, $K_2 = -K_p - 2K_d$, $K_3 = K_d$.

The analysis showed that expressions (4) and (5) are practically identical (the control signal $u(n)$ depends on the last three readings of the error signal). The main difference between them is that expression (4) allows you to determine the coefficients of a digital PID controller based on an analogue prototype. For expression (5), the coefficients K_1 , K_2 , K_3 must be selected when manually adjusting the control system. It may seem that these coefficients depend on the controller coefficients K_p , K_i , K_d so that they can be calculated, but this is not the case. This is easy to see if you pay attention to the fact that these coefficients K_1 , K_2 , K_3 do not consider the sampling rate.

Considering the above, digital and analogue PID controllers are not considered different entities. Therefore, methods for calculating the controller coefficients are considered regardless of whether digital or analogue. However, as the analysis has shown, there are at least two implementations of a digital PID controller, so the calculation methods must take into account the structure of the controller. In addition, as will be shown later, the digital controller (4) is not a complete analogue of the classic analogue PID controller. Thus, the problem of the algorithm for optimising the digital PID controller is relevant.

4. Results

This section will discuss the implementation of a digital PID controller. The controller will be based on a digital filter [20]. The method for calculating the filter coefficients will be performed using a genetic algorithm [6, 15].

The purpose of the work is to refine the digital PID controller model by using a digital filter.

To achieve the goal in the work, the following tasks must be solved: to analyse digital filters; suggest a filter type for the PID controller; choose a fitness function to optimise the filter coefficients; calculate the filter coefficients for the PID controller model; draw conclusions from the results.

4.1. Digital filter

Digital signal processing is used wherever necessary to perform tasks such as filtering, compressing, recovering, controlling, and measuring a signal: audio, video, or any signal from any source.

Filtering is the most common digital processing task, implemented using digital filters: filters with a finite impulse response (FIR filters); filters with infinite impulse response (IIR filters). In

general, a digital filter is understood as a hardware or software implementation of a mathematical algorithm, the input of which is a digital signal, and the output is another digital signal modified by the filter.

The primary operations of information filtering include noise suppression, smoothing, prediction, differentiation, signal separation, etc.

The main advantages of digital filters over analogue filters:

- may have parameters that are impossible to implement in analogue filters, for example, linear phase response;
- does not require calibration because its performance does not depend on the destabilising factors of the external environment, for example, temperature;
- input and output data can be saved for later processing;
- accuracy of digital filters is limited by the capacity of the filter coefficients.
- can be easily rearranged to filter a different frequency range, for example, by changing the data sampling rate.

In general, the digital filter is described by the following expression (6):

$$y(n) = \sum_{k=0}^{K-1} b_k \cdot x(n-k) + \sum_{m=0}^{M-1} a_m \cdot y(n-m), \quad (6)$$

where b_k , a_m – filter coefficients; $x(n)$, $y(n)$ – input and output signal; K , M – number of filter coefficients b_k , a_m respectively.

Expression (6) is also called IIR filter. Such a filter is often used when you must filter with a minimum number of arithmetic operations. If all the coefficients a_m are equal to zero, then such a filter is called an FIR filter. In this case, the digital filter will be described by the following difference expression (7):

$$y(n) = \sum_{k=0}^{K-1} h(k) \cdot x(n-k), \quad (7)$$

where $h(k) = b_k$ – impulse response of an FIR filter.

The amplitude-frequency response (AFR) of such a filter will have the following form (8):

$$H(\omega) = \sum_{k=0}^{K-1} h(k) \cdot e^{-j\omega k}, \quad (8)$$

where ω – circular frequency.

In many digital signal processing applications, the use of FIR filters is preferable because they have the following advantages:

- filter group delay constant (linear phase FIR filters);
- FIR filters are always stable.

Table 1

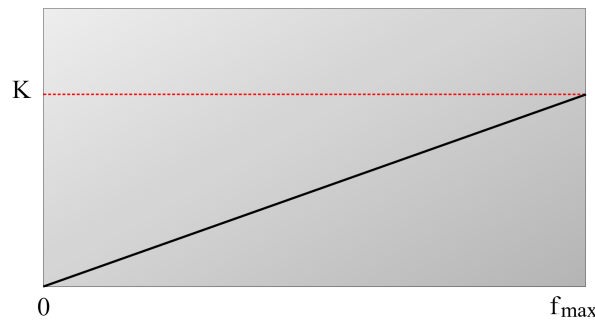
Types of linear phase filters and their characteristics.

Filter type	Impulse response	Number of impulse response coefficients	Amplitude-frequency response
I	symmetrical	odd	$H(\omega) = \sum_{k=0}^{(K-1)/2} a(k) \cdot \cos(\omega k)$
II	symmetrical	even	$H(\omega) = \sum_{k=1}^{K/2} b(k) \cdot \cos(\omega(k - 1/2))$
III	antisymmetric	odd	$H(\omega) = \sum_{k=1}^{(K-1)/2} c(k) \cdot \sin(\omega k)$
IV	antisymmetric	even	$H(\omega) = \sum_{k=1}^{K/2} d(k) \cdot \sin(\omega(k - 1/2))$

For FIR filters to be a linear phase, the impulse response must be symmetric or antisymmetric [20]. In this case, four types of FIR filters are possible (table 1).

In this case $a(0) = h(\frac{K-1}{2})$, $a(k) = 2h(\frac{K-1}{2} - k)$, $c(0) = 0$, $c(k) = 2h(\frac{K-1}{2} - k)$, $k = 1, 2, 3, \dots, \frac{K-1}{2}$, $b(k) = 2h(\frac{K}{2} - k)$, $d(k) = 2h(\frac{K}{2} - k)$, $k = 1, 2, 3, \dots, \frac{K}{2}$.

An example of an antisymmetric FIR filter with an even number of coefficients is a differentiating filter, the AFR of which corresponds to figure 6.

**Figure 6:** AFR of the differentiating term.

4.2. Development of a PID controller model

Let us define the transfer function of the analogue PID controller. For this, it is necessary to perform the Laplace transform of formula (2) with zero initial conditions $u(0) = 0$. As a result, we get the following expression:

$$W_A(p) = K_p + K_i \frac{1}{p} + K_d p, \quad (9)$$

where p – Laplace operator.

If in expression (9) we substitute $p = j\omega$, then we obtain an expression for the frequency response of the analogue PID controller (10):

$$W_A(\omega) = K_p - K_i \frac{j}{\omega} + jK_d\omega. \quad (10)$$

Let us look at the frequency response of the regulator at $K_p = 10$, $K_i = 1$, $K_d = 1$ (figure 7).

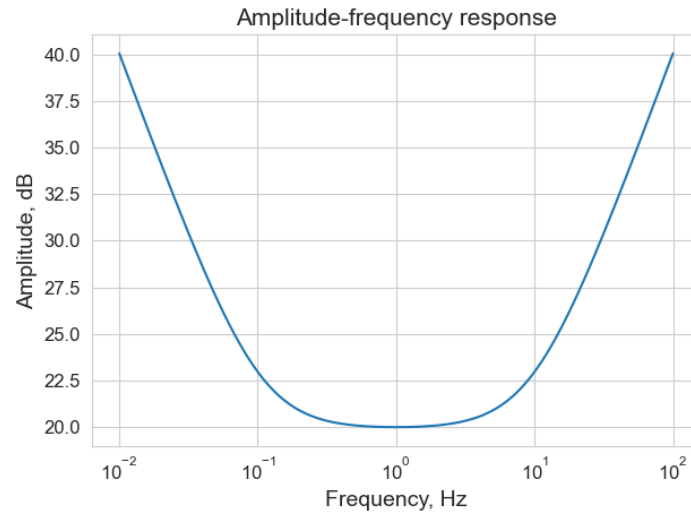


Figure 7: AFR of the PID controller.

The AFR of the analogue PID controller shows that the integrating component has an effect in the low-frequency range and the differentiating component in the high-frequency range.

Now, let us compare the AFR of the differentiating components of the analogue and digital PID controllers.

First, let us write the frequency response of the derivative component of the analogue PID controller. It can be seen from formula (10) that its frequency response is determined by the expression (11):

$$D_A(\omega) = K_d\omega. \quad (11)$$

Now, let us determine the AFR of the differentiating component of the digital PID controller. From expression (3), it can be seen that in the time domain, the differentiating component has the following form (12):

$$d_D(n) = K_{dd}(e(n) - e(n-1)). \quad (12)$$

In the brackets of expression (12), there is a digital filter of the first order. Therefore, the frequency response can be determined from the formula (8). As a result, we will have the following expression (13):

$$D_D(\omega) = 2K_{dd}\sin\left(\frac{\omega T_k}{2}\right). \quad (13)$$

Figure 8 shows the AFR of both regulators. For convenience of comparison, the frequencies are normalised (the sampling rate is taken equal to one), and the coefficients are taken equal to $K_d = 1$, $K_{dd} = 1$.

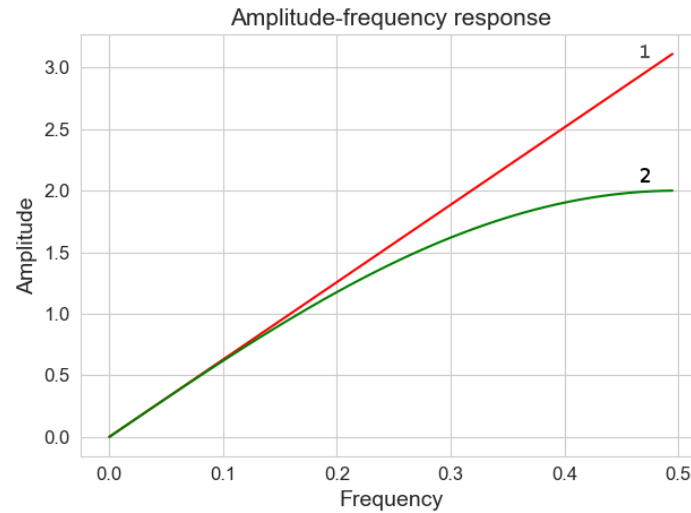


Figure 8: AFR of the differentiating component of PID controllers: 1 – analogue; 2 – digital.

The AFR of the differentiating component of the analogue PID controller corresponds to figure 6, but the digital one does not. Such a term works as a differentiating one only for 1/3 of the initial interval. However, in this area, its influence is minimal because the integrating and proportional components prevail there. Figure 9 shows the relative error. As you can see from the figure, the maximum error is more than 35%.

To eliminate this drawback, it is necessary to set a new model of the differentiating component. One of the options is using digital processing methods, namely, using the previously mentioned FIR filter with a linear phase.

From table 1, it follows that there are four types of such filters. Filters of types III and IV have an imaginary part of the AFR. However, the type III filter cannot always be used as such. The reason is that the value of the transmission coefficient at the maximum frequency will be equal to zero $H(\omega_{max}) = 0$ regardless of the filter coefficients (table 1). Such a filter can be used as a differentiating filter only in the initial section. We need to use the entire range, so for our task, the best solution would be to use a type IV filter.

From expression (7) and table 1, it follows that the differentiating component for a digital PID controller can be represented in the form of expression (14):

$$y(n) = \sum_{l=0}^{L-1} h(l) \cdot (x(n-l) - x(n+l-2L+1)), \quad (14)$$

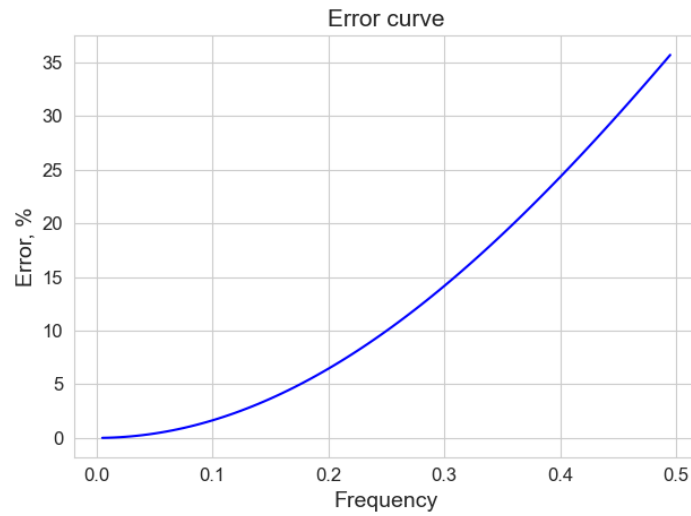


Figure 9: Relative error of the differentiating component of AFR of the digital PID controller.

where L – number of independent coefficients.

In this case, taking into account expression (3), the algorithm for implementing a digital PID controller will be described by the expression (15):

$$u(n) = K_p e(n) + K_{id} \sum_{k=0}^n e(k) + K_{dd} \sum_{l=0}^{L-1} h(l) \cdot (e(n-l) - e(n+l-2L+1)). \quad (15)$$

For simplicity, we call it PPID. By analogy, we can write an expression similar to the recurrence formula (4) or recurrently record only an integrating part (this will reduce the likelihood of overflow and reduce the number of arithmetic operations) in the following way (16):

$$u(n) = K_p e(n) + I(n) + K_{dd} \sum_{l=0}^{L-1} h(l) \cdot (e(n-l) - e(n+l-2L+1)), \quad (16)$$

where $I(n) = I(n-1) + K_{id}e(n)$ – integrating component.

To obtain the final model of the PPID controller, it is necessary to determine the coefficients $h(l)$, where $l = 1, 2, 3, \dots, L-1$.

To synthesise the filter (14), a fitness function is required. The synthesis of the filter with the best uniform approximation will be performed in the form of the problem of minimising the weighted Chebyshev norm (17):

$$e = \max \left(W(\omega) \left| H(\omega) - \hat{H}(\omega) \right| \right) \rightarrow \min, \quad (17)$$

where $H(\omega)$, $\hat{H}(\omega)$ – AFR of the approximated and approximating filters, respectively, $W(\omega)$ – weight function.

4.3. Experiments

To test the model of the digital PPID controller, we will carry out some experiments. Let us synthesise a digital FIR filter (14). There are many methods for their design in the scientific literature [20]. The most widely used are the classical methods for calculating FIR filters: weighing method; frequency sampling method; least squares method; and best uniform approximation method. The first two are not optimisation methods, but are fairly easy to use. The third and fourth methods are referred to as optimisation methods. The fourth method allows obtaining the best results, but, as a rule, it is impossible to determine analytically the function of the best uniform approximation. However, in this case, the synthesis of the FIR filter will be carried out using the genetic algorithm [18], which will allow us to obtain some advantages; for example, when searching for the values of the filter coefficients, we will take into account the effect of quantisation.

A genetic algorithm is a heuristic algorithm, a kind of evolutionary algorithm, with the help of which optimisation problems are solved using methods of natural evolution, similar to natural selection in nature [6, 15].

When solving a problem with a genetic algorithm, it is necessary to isolate the phenotype that determines the real object. In our case, the filter coefficients forming an species will act as a phenotype (figure 10).

$h(0)$	$h(1)$	$h(2)$	...	...	$h(L-1)$
--------	--------	--------	-----	-----	----------

Figure 10: The structure of the individual.

The fitness function will be described by expression (17). The AFR of the differentiating part of the analogue PID filter (11) at $K_d = 1$ (figure 8, graph 1) will act as the AFR of the approximated filter. The AFR of the approximating filter will be determined by a type IV filter (table 1).

The simulation was carried out using the Python programming language. To implement the genetic algorithm, it is necessary to tune the hyperparameters. In our case, they will have the following form:

```
POPULATION = 100 # number of individuals in the population
SURVIVOR = 0.2 # survival probability
MUTATION = 0.1 # possibility of mutating of an individual
GENERATIONS = 250 # maximum number of generations
```

Below is the fitness function code in the Python programming language, which corresponds to the expression (17), where *prototype* is an instance of the *PrototypeFIR* class of the filter being approximated (figure 6); *fir* – an instance of the *Fir1T* class approximating the filter (7).

```
# fitness function
def fitness(individual):
    fmax = prototype.getSamplingFrequency() / 2
    fir = fir1t.Fir1T(fmax, individual)
    emax = 0
    for fi in prototype.getReferencePoints():
        e = abs(prototype.getGain(fi) - fir.getGain(fi)) * prototype.getWeight(fi)
```

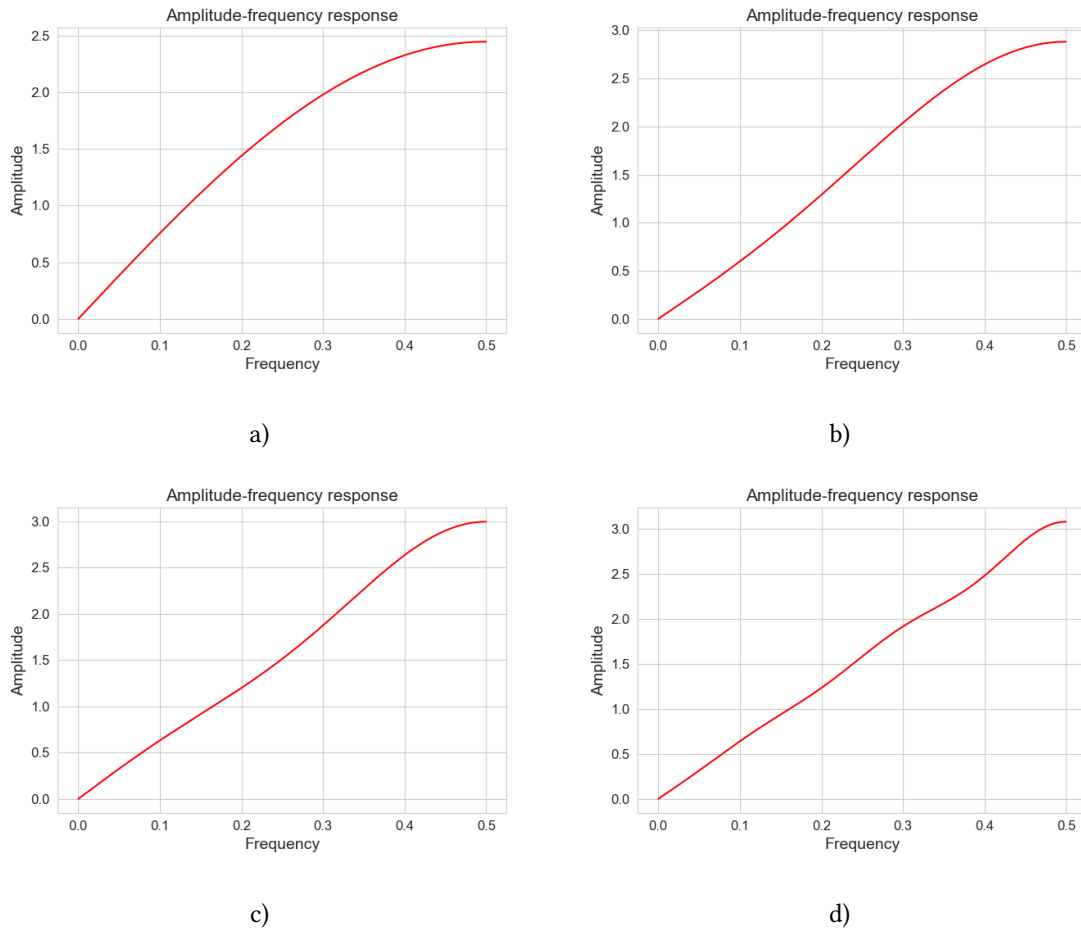


Figure 11: AFR of the differentiating component of the PPID controller: a) $L = 1$; b) $L = 2$; c) $L = 3$; d) $L = 6$.

```

if e > emax:
    emax = e
return emax,

```

Figure 11 shows the synthesis of a differentiating component PPID controller.

Table 2 shows the calculated coefficients of filters of different orders and also indicates the approximation error of the differentiating component of the PPID controller.

If we compare the proposed model of the PPID controller with the original digital PID controller (3) at $L = 1$, we can see that the expressions differ only by the factor $h(0)$ (table 2). However, due to him, the error was reduced by 15%. Figure 12 shows the relative error of this PPID controller.

Figure 13 shows the tendency of decreasing the error with an increase in the number of coefficients $h(l)$.

Let us compare the number of arithmetic operations (number of addition and multiplication

Table 2

FIR filter coefficients for the developed model of a digital PID controller.

L	$h(l)$	Error, %
1	1.22323099	22.32
2	-0.1310637, 1.30919328	8.40
3	0.05080995, -0.16309586, 1.28291046	4.77
4	-0.02893289, 0.06681262, -0.14717449, 1.27747904	3.25
5	0.01963494, -0.03861118, 0.05499173, -0.14416749, 1.27554355	2.44
6	-0.01467901, 0.02619673, -0.0291035, 0.05288374, -0.14306129, 1.27468666	1.95
7	0.0116237, -0.01945604, 0.01826656, -0.02751572, 0.05212775, -0.14253636, 1.27425729	1.62
8	-0.00959958, 0.01532948, -0.01266609, 0.01698343, -0.02694503, 0.05174926, -0.14223351, 1.2739692	1.38
9	0.00816843, -0.01257872, 0.00936974, -0.01156839, 0.01650426, -0.02665169, 0.0515316, -0.14203773, 1.27377827	1.20
10	-0.00710013, 0.01063027, -0.00727018, 0.0084291, -0.01118041, 0.01626373, -0.026466, 0.05138662, -0.14191183, 1.27366718	1.07
11	0.00627216, -0.00918011, 0.00583902, -0.00643919, 0.00810167, -0.01099786, 0.01613495, -0.02635945, 0.05128535, -0.1418134, 1.2735769	0.96

operations) for the classical PID model and the developed PPID model. When calculating, we will consider that the subtraction operation is equivalent to the addition operation because the addition operation can replace the subtraction operation with some opposite sign, and the division operation is equivalent to the multiplication operation. Expression (3) cannot be realised directly using a microprocessor (microcontroller), so it is converted into expression (4) but for adequate comparison with the developed PPID model, expression (3) is better written in this way:

$$u(n) = K_p e(n) + I(n) + K_{dd}(e(n) - e(n-1)), \quad (18)$$

where $I(n) = I(n-1) + K_{id}e(n)$ – integrating component.

Now, the classical model (18) has a form that is similar to the developed PPID model (16), but this is not enough for comparison because it is obvious that the original model is more straightforward, but has a significant error. The question arises: Can the error of the classical PID model be reduced? The answer is unambiguous – yes, it is possible. If you look at Figure 8, you can see that the initial section of the curve of the D-component is close to linear, and the error is less than 5%. For the PID controller to satisfy this requirement, it is necessary to lower the sampling period T_k (increase the sampling frequency) by 3-5 times. Figure 14 shows an example of increasing the sampling rate by a factor of 4.

This approach is used in many digital systems to reduce the error caused by the sampling step. Figure 15 shows the architecture of the control node in the PX4 autopilot [21]. The documentation states: “Most of the IMU drivers sample the data at 1kHz, integrate it and publish with 250Hz”[21], so the sampling and processing rate is four times higher than necessary.

Table 3 compares the number of arithmetic operations of the classical PID controller model (18) and the developed PPID controller model (16) for three cases. The results in the table show

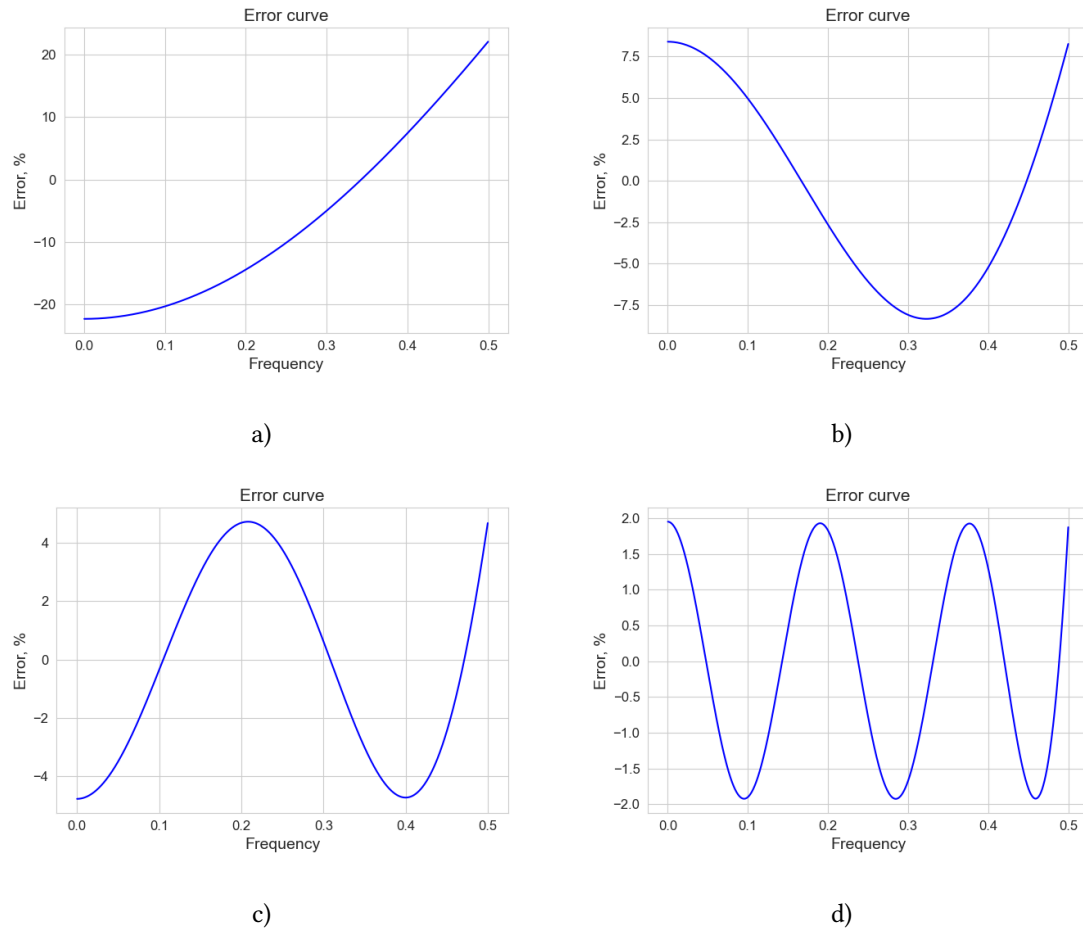


Figure 12: Relative error of the differentiating component of the AFR of the PID controller: a) $L = 1$; b) $L = 2$; c) $L = 3$; d) $L = 6$.

that with comparable errors of the D-component, the number of multiplication operations of the developed PPID regulator is almost two times less, and the number of addition operations is one and a half times less.

Another disadvantage of the classical PID controller model is the noise amplification at the output of the D-component. The reason for this effect is explained in figure 8. This effect is absent in the developed model of the PID regulator. It can be verified by using the following diagram (figure 16).

The input is the original signal $x(n)$ and noise $e_{in}(n)$, and the output is noise $e_{out}(n)$, after passing through the differentiating component.

Thus, it can be noted that $3 \leq L \leq 6$ is sufficient for UAV control in the developed PPID controller model.

In a classic digital PID controller, the amplitude-frequency response of the differential component is nonlinear. Because the initial section of the characteristic is quite linear, they try to

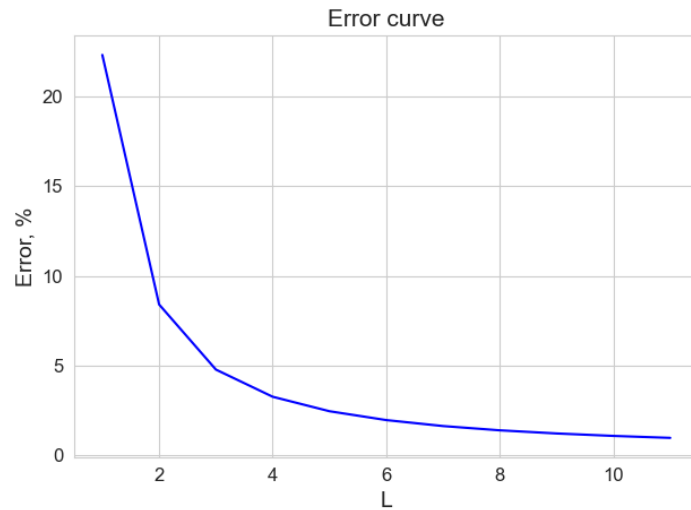


Figure 13: Changing error differentiating component with increasing L of PPID controller.

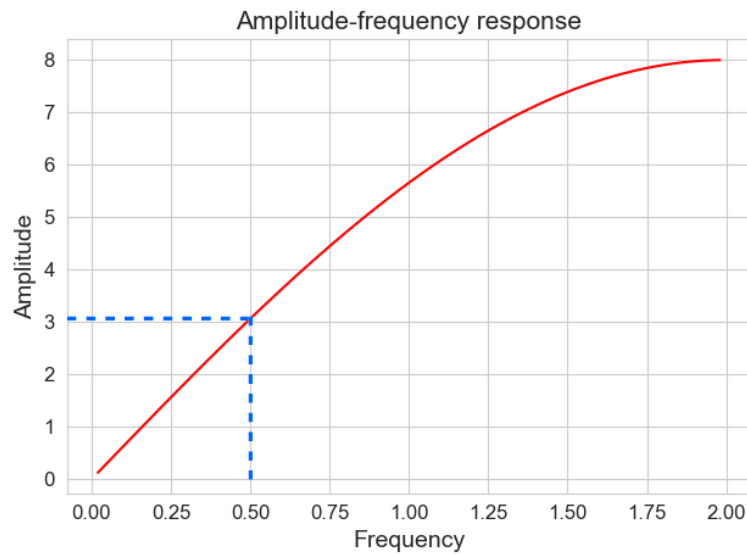


Figure 14: Frequency response of PID controller at four times increase of sampling frequency.

use this part of the characteristic in the PID controller, increasing the sampling frequency by 3-5 times. In this case, the error will not exceed 5%. However, this approach leads to additional costs of computing resources and increased noise levels.

In this paper, we propose another approach to eliminate the nonlinearity of the characteristic. The differential component is considered a nonrecursive digital filter, increasing the order of which reduces the nonlinearity of the amplitude-frequency response.

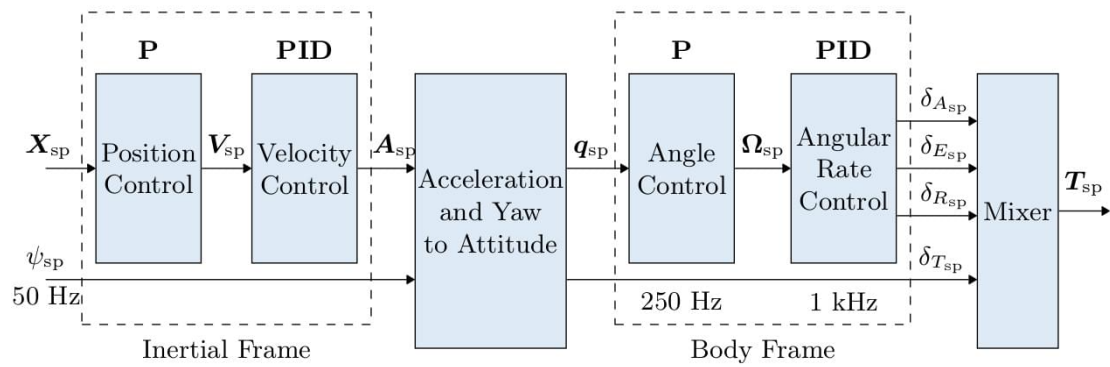


Figure 15: Multicopter control architecture.

Table 3

Comparison of the number of arithmetic operations in PID regulators.

PID					PPID			
Multiplication operations	Addition operations	Frequency multiplier	Error, %		Multiplication operations	Addition operations	L	Error, %
1	9	12	3	4.75	5	8	3	4.77
2	12	16	4	2.76	7	12	5	2.44
3	15	20	5	1.80	8	14	6	1.95

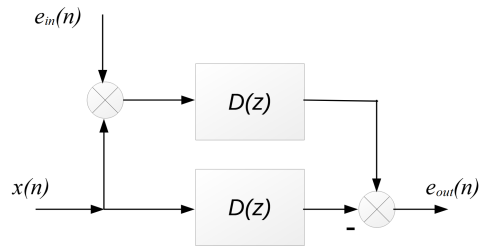


Figure 16: Testing the noise level at the output of the D-component.

5. Conclusions

The problem of building a model of a digital PID controller that can be used in unmanned aerial vehicles is considered.

The regulator is based on digital filtering methods. It is proposed to use an FIR filter with a linear phase of the IV type as a filtering device. This made it possible to fairly accurately approximate the differentiating component. So, for a classic digital PID controller, the introduction of one coefficient has reduced the relative frequency response error by 15%. In addition, the PID controller model was developed with the ability to use ready-made methods for calculating the PID controller coefficients.

The digital filter coefficients are calculated using a genetic algorithm. The phenotype is the filter coefficients. The Chebyshev norm was used as a fitness function.

The simulation results were carried out using the Python programming language.

Data for all filters up to 21 orders (up to 11 independent coefficients) has been analysed.

As shown in the work, for most control problems in robotic systems, it is sufficient to use filters with 3-6 independent coefficients.

Also, the analysis showed that with the same errors of the classical PID controller and the developed PPID controller, the cost of computational resources is approximately two times less.

Perspectives for further research consist of testing the proposed methods on a wider range of problems, studying the effects of finite bit depth, and analysing the structure of the PID controller.

References

- [1] ArduPilot Documentation, 2023. Available from: <https://ardupilot.org/ardupilot/>.
- [2] Atmel, 2016. AVR221: Discrete PID Controller on tinyAVR and megaAVR devices. Available from: http://ww1.microchip.com/downloads/en/AppNotes/Atmel-2558-Discrete-PID-Controller-on-tinyAVR-and-megaAVR_ApplicationNote_AVR221.pdf.
- [3] Baikar, P.M., 2014. Design of PID controller based information collecting robot in agricultural field. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, 3(8), pp.11013–11019. Available from: <https://doi.org/10.15662/ijareeie.2014.0308012>.
- [4] Bhandari, P. and Csurscia, P.Z., 2022. Digital implementation of the PID controller. *Software Impacts*, 13, p.100306. Available from: <https://doi.org/10.1016/j.simpa.2022.100306>.
- [5] Borase, R.P., Maghade, D.K., Sondkar, S.Y. and N., P.S., 2021. A review of PID control, tuning methods and applications. *International Journal of Dynamics and Control*, 9, pp.818–827. Available from: <https://doi.org/10.1007/s40435-020-00665-4>.
- [6] Carr, J., 2014. An introduction to genetic algorithms. Available from: <https://www.whitman.edu/documents/academics/mathematics/2014/carrjk.pdf>.
- [7] Graf, M.S., 2020. Unmanned aerial vehicle bypass training system. *Technical Engineering*, 2(86), pp.81–85. Available from: [https://doi.org/10.26642/ten-2020-2\(86\)-81-85](https://doi.org/10.26642/ten-2020-2(86)-81-85).
- [8] Hornsey, S., 2012. A Review of Relay Auto-tuning Methods for the Tuning of PID-type Controllers. *Reinvention: an International Journal of Undergraduate Research*, 5(2). Available from: <http://www.warwick.ac.uk/reinventionjournal/archive/volume5issue2/hornsey>.
- [9] Hou, D., 2015. PID Control on PIC16F161X by using a PID Peripheral. Available from: <http://ww1.microchip.com/downloads/en/AppNotes/90003136A.pdf>.
- [10] Jayachitra, A. and Vinodha, R., 2014. Genetic Algorithm Based PID Controller Tuning Approach for Continuous Stirred Tank Reactor. *Advances in Artificial Intelligence*, 2014. Available from: <https://doi.org/10.1155/2014/791230>.
- [11] Kadu, C. and Patil, C., 2016. Design and Implementation of Stable PID Controller for Interacting Level Control System. *Procedia Computer Science*, 79, pp.737–746. Proceedings

- of International Conference on Communication, Computing and Virtualization (ICCCV) 2016. Available from: <https://doi.org/10.1016/j.procs.2016.03.097>.
- [12] Maghsadghagh, A., 2016. Implementation of PID controller by microcontroller of PIC (18 Series) and controlling the height of liquid in sources. *Advances in Robotics & Automation*, 5(3), p.1000156. Available from: <https://doi.org/10.4172/2168-9695.1000156>.
- [13] Masade, S., Parmar, S. and Bhanushali, A.J., 2016. Speed Control for Brushless DC Motors using PID Algorithm. Available from: <https://ru.scribd.com/document/561125158/Speed-Control-for-Brushless-DC-Motor-Using-PID-Algorithm>.
- [14] Mohamed, N.M., Abdalaziz, A.A., Ahmed, A.A. and Ahmed, A.A., 2017. Implementation of a PID control system on microcontroller (DC motor case study). *2017 International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE)*, pp.1–5. Available from: <https://doi.org/10.1109/ICCCCEE.2017.7866088>.
- [15] Mutingi, M. and Mbohwa, C., 2017. *Grouping genetic algorithms. Advances and applications*. Springer Cham. Available from: <https://doi.org/10.1007/978-3-319-44394-2>.
- [16] Nikitchuk, T.M., Vakaliuk, T.A., Chernysh, O.A., Korenivska, O.L., Martseva, L.A. and Osadchyi, V.V., 2021. Architecture for edge devices for diagnostics of students' physical condition. In: S.O. Semerikov, ed. *Joint Proceedings of the Workshops on Quantum Information Technologies and Edge Computing (QuaInT+doors 2021)*, Zhytomyr, Ukraine, April 11, 2021. CEUR-WS.org, *CEUR Workshop Proceedings*, vol. 2850, pp.45–56. Available from: <http://ceur-ws.org/Vol-2850/paper3.pdf>.
- [17] Petrosian, A.R., Petrosian, R.V. and Pidtychenko, O.V., 2021. Optimization of the PID controller model based on a digital filter. *Scientific notes of Taurida National V.I. Vernadsky University. Series: Technical Sciences*, 32(71)(4), pp.129–134. Available from: <https://doi.org/10.32838/2663-5941/2021.4/20>.
- [18] Petrosian, R., Kuzmenko, O. and Petrosian, A., 2021. Method for calculating the FIR filter based on genetic algorithm. *International Scientific Journal "Computer Systems and Information Technologies"*, 1, pp.19–24. Available from: <https://doi.org/10.31891/CSIT-2021-3-3>.
- [19] Petrosian, R.V., Pilkevych, I.A. and Petrosian, A.R., 2023. Algorithm for optimizing a PID controller model based on a digital filter using a genetic algorithm. In: T.A. Vakaliuk and S.O. Semerikov, eds. *Proceedings of the 3rd Edge Computing Workshop, Zhytomyr, Ukraine, April 7, 2023*. CEUR-WS.org, *CEUR Workshop Proceedings*, vol. 3374, pp.97–111. Available from: <https://ceur-ws.org/Vol-3374/paper07.pdf>.
- [20] Proakis, J.G. and Manolakis, D.K., 2014. *Digital Signal Processing*. 4th ed. Pearson Education Limited.
- [21] PX4 User Guide, 2023. Available from: https://docs.px4.io/v1.alex_f_s500build/en/concept/architecture.html.
- [22] Sablina, G.V. and Markova, V.A., 2022. Tuning a PID Controller in a System with a Delayed Second-Order Object. *Optoelectronics, Instrumentation and Data Processing*, 58, pp.410–417. Available from: <https://doi.org/10.3103/S8756699022040112>.
- [23] Samakwong, T. and Assawinchaichote, W., 2016. PID Controller Design for Electro-hydraulic Servo Valve System with Genetic Algorithm. *Procedia Computer Science*, 86, pp.91–94. 2016 International Electrical Engineering Congress, iEECON2016, 2-4 March 2016, Chiang Mai, Thailand. Available from: <https://doi.org/10.1016/j.procs.2016.05.023>.

- [24] Schofield, O.B., Iversen, N. and Ebeid, E., 2022. Autonomous power line detection and tracking system using UAVs. *Microprocessors and Microsystems*, 94, p.104609. Available from: <https://doi.org/10.1016/j.micpro.2022.104609>.
- [25] Trafczynski, M., Markowski, M., Kisielewski, P., Urbaniec, K. and Wernik, J., 2019. A Modeling Framework to Investigate the Influence of Fouling on the Dynamic Characteristics of PID-Controlled Heat Exchangers and Their Networks. *Applied Sciences*, 9(5), p.824. Available from: <https://doi.org/10.3390/app9050824>.
- [26] Yazid, Y., Ez-Zazi, I., Guerrero-González, A., El Oualkadi, A. and Arioua, M., 2021. UAV-Enabled Mobile Edge-Computing for IoT Based on AI: A Comprehensive Review. *Drones*, 5(4), p.148. Available from: <https://doi.org/10.3390/drones5040148>.
- [27] Zhao, J. and Xi, M., 2020. Self-Tuning of PID Parameters Based on Adaptive Genetic Algorithm. *IOP Conference Series: Materials Science and Engineering*, 782(4), p.042028. Available from: <https://doi.org/10.1088/1757-899X/782/4/042028>.