CONTROL SYSTEM TUNING FOR UNMANNED ROTORCRAFT USING PARTICLE SWARM OPTIMIZATION ALGORITHM

Łukasz Kiciński
Faculty of Power and Aeronautical Engineering, Warsaw University of Technology
Warsaw, Poland

Sebastian Topczewski
Faculty of Power and Aeronautical Engineering, Institute of Aeronautics and Applied Mechanics Warsaw University of Technology
Warsaw, Poland

ABSTRACT

This paper presents an alternative approach to designing automatic flight control systems for unmanned rotorcraft using a Linear Quadratic Regulator (LQR) controller optimized via the Particle Swarm Optimization (PSO) algorithm. The research formulates control system design as a constrained optimization problem, where the cost function is defined as the sum of control errors over a finite simulation period, given a predefined desired flight trajectory. The proposed control system was developed for the ARCHER unmanned rotorcraft, designed at the Warsaw University of Technology. The tuning and optimization process was conducted using a linear vehicle model, enhanced with actuator saturation constraints to reflect flight control system limitations. The optimized weighting parameters were then tested on a fully nonlinear rotorcraft model developed in FLIGHTLAB software.

NOTATION

*

Symbols

c_1, c_2 scaling coefficients in the PSO algorithm

$\Delta\mathbf{u}$ increment of the control vector

$\Delta\mathbf{x}$ increment of the state vector

$e(t)$ control error

$\mathbf{g}$ global best solution among all particles

$J(\mathbf{K})$ cost function dependent on the gain matrix $\mathbf{K}$

$\mathbf{K}$ static state feedback gain matrix

$\mathscr{X}$ set of feasible solutions

N number of particles in the PSO algorithm

$O_b x_b y_b z_b$ body-fixed coordinate system

$O_g x_g y_g z_g$ gravitational coordinate system fixed to the body

$O_n x_n y_n z_n$ inertial coordinate system fixed to the Earth

$\mathbf{P}$ solution to the algebraic Riccati equation

$\mathbf{p}_n^{(i)}$ best solution found so far by the n-th particle in the i-th PSO iteration

$\mathbf{Q}$ weighting matrix for the control error in the LQR problem

$\mathbf{R}$ weighting matrix for the control effort in the LQR problem

$\mathbf{r}_p$ reference state trajectory

$\mathbf{u}(t)$ control vector

$\mathbf{u_0}$ control vector at equilibrium conditions

$\mathbf{v}_n^{(i)}$ velocity of the n-th particle in the i-th PSO iteration

$\mathbf{x}$ state vector

$\mathbf{x}_0$ state vector at equilibrium conditions

x_a, x_b cyclic pitch components of the main rotor blades [%]

x_c collective pitch of the main rotor blades [%]

x_r collective pitch of the tail rotor blades [%]

$\mathbf{x}_i$ current solution of the optimization problem in the i-th iteration

$\mathbf{x}_n^{(i)}$ position of the n-th particle in the i-th PSO iteration

$\mathbf{y}(t)$ output vector

σ_{in} inertia coefficient of the PSO algorithm

*

Acronyms

ARE Algebraic Riccati Equation

LQG Linear Quadratic Gaussian

LQR Linear-Quadratic Regulator

MIMO Multiple Input-Multiple Output

PID Proportional-Integral-Derivative

PSO Particle Swarm Optimization

UAV Unmanned Aerial Vehicle

VTOL Vertical Take-off and Landing

INTRODUCTION

Unmanned aerial vehicles (UAVs) have become a significant component in modern aerial operations, offering advantages in terms of mission flexibility, safety, and operational cost. Among various UAV configurations, rotorcrafts such as helicopters stand out due to their vertical takeoff and landing (VTOL) capabilities, hover flight, and maneuverability in confined environments. These features make unmanned helicopters especially suitable for tasks such as aerial surveillance, search and rescue, precision delivery, and environmental monitoring.

However, the design and implementation of flight control systems for unmanned helicopters remains a challenging task. The nonlinear dynamics, strong coupling between control channels, sensitivity to external disturbances, and rapid dynamic responses demand advanced control strategies. Moreover, the lack of onboard pilots requires the control system to ensure full autonomous flight stability and trajectory tracking under various flight conditions.

There are many possible approaches to the problem of designing an automatic control system for an unmanned helicopter, as evidenced by the variety of technical papers addressing this issue. A literature review of methods for designing autopilots for unmanned helicopters is presented below.

A reliable approach to designing automatic flight control systems for unmanned helicopters involves conducting an identification experiment that allows the dynamics of the analyzed aircraft to be described in hover conditions. Then, using the obtained model, it is possible to design a PID controller structure and subsequently tune it to achieve satisfactory performance for helicopter position control. This approach is described in detail in the Ref. 1, and it enabled the designers to achieve satisfactory performance under near-hover conditions.

An alternative approach to the synthesis of control laws for unmanned helicopters may be based on the design of an H_∞ control loop, the aim of which is to minimize the influence of disturbances in the system on the aircraft's response. This method not only guarantees the stability of the designed control system but also provides an effective solution for controlling systems with multi-dimensional dynamics (MIMO), where decoupling of the dynamics is not possible. The design of a static H_∞ controller for an unmanned helicopter is presented in the Ref. 2.

Another method that can be applied is the previously mentioned approach based on the Linear-Quadratic Regulator (LQR) theory, which also works very well in controlling MIMO systems. Based on a linear model of the helicopter's dynamics, the LQR controller generates optimal control for an infinite time horizon. The existence of an analytical solution to the posed problem and the high effectiveness of LQR regulation in controlling multi-dimensional dynamics make this method a popular choice in the aerospace industry. An example of using the LQR method to control an unmanned helicopter is described in the Ref. 3.

The control system design methods mentioned above rely heavily on a mathematical representation of the helicopter's flight dynamics. An alternative approach to control law synthesis that does not necessarily depend on a mathematical model of the system is the use of fuzzy inference systems. These systems can be designed based on expert knowledge and provide a suitable approach for designing nonlinear control laws. The article Ref. 4 presents the process of designing a fuzzy inference system for controlling an unmanned rotorcraft. Another example of using fuzzy systems in the control of an unmanned helicopter is given in the Ref. 5, which describes the integration of a Mamdani-type controller with Takagi-Sugeno inference to control the forward flight speed of an unmanned helicopter.

An alternative to the methods presented in the above examples is the use of the Model Predictive Control (MPC) method. While it depends heavily on the quality of the mathematical representation of the system, it also offers many formulations and provides the designer with a wide range of tuning parameters that shape the response of the controlled object. An example of applying predictive control to the flight control of an unmanned helicopter is described in the Ref. 6. The paper also presents the use of Laguerre functions to generate control signals in the system and discusses the impact of control signal constraints on the nature of the system's response.

The research presented in this paper aims to demonstrate a method for automatic tuning of an LQR controller using heuristic optimization techniques, exemplified by the Particle Swarm Optimization (PSO) algorithm. The proposed approach formulates the LQR weight selection as a constrained optimization problem and evaluates the resulting control performance through simulation studies of an automatic flight

control system. The novelty of this work lies in the development of a tuning framework that reduces the dimensionality of the optimization space compared to traditional structured controller design methods.

HELICOPTER MODEL

Description of the control plant

The control plant using is the unmanned helicopter *ARCHER*. This aircraft was developed at the Division of Automation and Aeronautical Systems, which is part of the Faculty of Power and Aeronautical Engineering at Warsaw University of Technology. *ARCHER* is an acronym for the full name of the moving platform: *Autonomous Reconfigurable Compound Helicopter for Education and Research*, which was introduced in the Ref. 7. In its simplest version, the aircraft is a rotorcraft designed in a classical configuration, using a main rotor and a tail rotor to control its state. Additionally, the modular design of *ARCHER* allows for easy expansion of its functionality by attaching wings or pusher propellers, which enables higher forward flight speeds and increases the range of the platform. A photo of the helicopter in flight in its classical configuration is shown in Figure 1.

For the purposes of this work, the ARCHER helicopter is considered in its basic form, which consists of a fuselage, two motors, a main rotor, a tail rotor, and a fixed skid-type landing gear. Both rotors are powered by separate DC motors, which eliminates the need for a complex transmission system with a main gearbox. The main rotor has teetering hub, a popular solution in small helicopters. The main rotor blades are designed using the NACA23012 airfoil, while the tail rotor uses the symmetric NACA0015 airfoil. The main rotor of the ARCHER helicopter rotates in a clockwise direction.

The basic mass and geometric characteristics of the ARCHER helicopter are presented in Table 1.

Table 1. Basic mass and geometric characteristics of the ARCHER helicopter

Parameter	Value	Unit
Main rotor diameter	1.78	m
Number of main rotor blades	2	-
Main rotor speed	1500 – 2000	RPM
Tail rotor diameter	0.158	m
Number of tail rotor blades	2	-
Tail rotor speed	7200	RPM
Take-off weight (TOW)	8	kg

FLIGHTLAB Software Description

FLIGHTLAB, developed by Advanced Rotorcraft Technology Inc., is a widely used tool for the analysis of flying vehicle dynamics, with a particular focus on rotorcraft modeling. Its

Figure 1. Photo of the *ARCHER* helicopter in flight (Ref. 8)

modular structure enables the parallel analysis of the state of individual components of the entire aircraft.

In the FLIGHTLAB environment, the model of the flying object is built by separately defining modules responsible for simulating the behavior of particular helicopter components. Once the program modules are defined, the next step is to specify the mutual relations between them and the signals exchanged. Selecting the number of elements and defining the connections within the program workspace is the main challenge in helicopter modeling using this software.

In addition to implementing classical helicopter flight dynamics equations (derived in the Ref. 9), the software supports analysis of main rotor deformations, includes unsteady aerodynamic effects, considers flow interference phenomena around the structure, and even allows modeling of tip vortices shed from the rotor blades.

Full Model of the ARCHER Helicopter

In this paper, the FLIGHTLAB environment was used to create a detailed model of the ARCHER helicopter, which serves as a platform for testing the performance of the designed control system.

To model the main rotor, the blade element method was used to determine aerodynamic loads. Given the blade airfoil profiles, it was possible to accurately describe the lift force, drag force, and pitching moment of the main rotor as nonlinear functions of dynamic pressure, angle of attack, and Mach number. These characteristics were obtained via detailed CFD analysis, then tabulated and implemented in the rotor dynamics simulation. The Peters-He model, introduced in the Ref. 10, was used to compute induced velocity.

For the tail rotor, momentum theory was used to determine thrust. This method was chosen due to its computational efficiency.

The fuselage was modeled as a rigid body with six degrees of freedom. Its mass properties—including center of gravity position, inertia tensor, and total mass—were included in the numerical model. As in the case of the main rotor, aerodynamic loads acting on the fuselage were determined using

CFD analysis and tabulated for implementation in the simulation software. In this configuration, aerodynamic forces and moments were described as nonlinear functions of dynamic pressure, angle of attack, and sideslip angle. The fuselage model also includes the locations of onboard sensors.

The numerical model of the helicopter also incorporates landing gear dynamics, which enables the analysis of landing maneuvers and ground contact. Skid friction is also taken into account.

To simulate environmental conditions during flight, FLIGHTLAB allows users to apply the 1959 ARDC atmospheric model. Turbulence effects were neglected in this study.

The final ARCHER helicopter model, composed of the components described above, is characterized by 26 state variables:

- helicopter position in the inertial coordinate system $O_n x_n y_n z_n$ (3)
- linear velocities in the body-fixed frame $O_b x_b y_b z_b$ (3)
- integrals of linear velocities in the body-fixed frame $O_b x_b y_b z_b$ (3)
- helicopter attitude angles (3)
- angular velocities in the body-fixed frame $O_b x_b y_b z_b$ (3)
- integrals of angular velocities in the body-fixed frame $O_b x_b y_b z_b$ (3)
- induced velocity components from the Peters-He model (6)
- induced velocity from the tail rotor (1)
- tail rotor pitch angle (1)

In the commonly accepted flight dynamics formalism, the control input vector u for the presented case can be described by Equation 1:

$$\mathbf{u} = [x_a, x_b, x_c, x_r]^T \tag{1}$$

where:

- x_a – longitudinal cyclic pitch of the main rotor
- x_b – lateral cyclic pitch of the main rotor
- x_c – collective pitch of the main rotor
- x_r – collective pitch of the tail rotor

Linear Model of Helicopter Dynamics

Due to the complexity of helicopter dynamics modeling provided by the FLIGHTLAB numerical approach, it is possible to simplify this description into a linear dynamic model. The linear model was derived under the following assumptions:

- only the following variables are used to describe the helicopter's motion in space:
  - $[X, Y, Z]^T$ – position in the inertial frame $O_n x_n y_n z_n$
  - $[\Phi, \Theta, \Psi]^T$ – attitude angles
  - $[U, V, W]^T$ – linear velocities in the body-fixed frame $O_b x_b y_b z_b$
  - $[P, Q, R]^T$ – angular velocities in the body-fixed frame $O_b x_b y_b z_b$
- at the operating point, the helicopter is hovering at 11.47 ft altitude, which corresponds to a skid height of 10 ft above ground level
- the state vector components can be expressed as the sum of the equilibrium value and the deviation, as in Equation 2:

$$\mathbf{x} = \mathbf{x}_0 + \Delta\mathbf{x} \tag{2}$$

- the control vector components can be expressed as the sum of the equilibrium input and the deviation, as in Equation 3:

$$\mathbf{u} = \mathbf{u}_0 + \Delta\mathbf{u} \tag{3}$$

The values of control inputs and attitude angles required to maintain equilibrium in the scenario were determined using a trim analysis performed in FLIGHTLAB. These correspond to the initial conditions for all simulations described in this work. The equilibrium state and control vectors are defined by Equations 4 and 7.

$$\mathbf{x}_0 = [X_0, Y_0, Z_0, \Phi_0, \Theta_0, \Psi_0, U_0, V_0, W_0, P_0, Q_0, R_0]^T \tag{4}$$

where:

$$\begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -3.43 \end{bmatrix} [\text{m}], \quad \begin{bmatrix} \Phi_0 \\ \Theta_0 \\ \Psi_0 \end{bmatrix} = \begin{bmatrix} 0.0698 \\ -0.0087 \\ 0 \end{bmatrix} [\text{rad}] \tag{5}$$

$$\begin{bmatrix} U_0 \\ V_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [\text{m/s}], \quad \begin{bmatrix} P_0 \\ Q_0 \\ R_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [\text{rad/s}] \tag{6}$$

$$\mathbf{u}_0 = [x_{a_0}, x_{b_0}, x_{c_0}, x_{r_0}]^T \tag{7}$$

where:

$$\begin{bmatrix} x_{a_0} \\ x_{b_0} \\ x_{c_0} \\ x_{r_0} \end{bmatrix} = \begin{bmatrix} 50.3 \\ 50.4 \\ 64.5 \\ 59.4 \end{bmatrix} [-] \tag{8}$$

The linear model of the unmanned helicopter dynamics is given by Equations 9 and 10:

$$\dot{\mathbf{x}} = \mathbf{A}\Delta\mathbf{x} + \mathbf{B}\Delta\mathbf{u} \tag{9}$$

$$\mathbf{y} = I\mathbf{x} \tag{10}$$

where:

- $\Delta\mathbf{x} = [x, y, z, \phi, \theta, \psi, u, v, w, p, q, r]^T$
- $\Delta\mathbf{u} = [\Delta x_a, \Delta x_b, \Delta x_c, \Delta x_r]^T$

CONTROL METHODOLOGY

Linear-Quadratic Regulator principles

Linear-Quadratic Regulator is a popular full-state feedback method of designing multidimensional automatic flight control systems. Presented framework can be applied to fully controllable and observable dynamical systems, assuming that full state vector can be measured in flight conditions. It relies on computing **K** gain matrix, which constitutes control law presented in the equation 11.

$$\mathbf{u} = -\mathbf{K}\mathbf{x} \tag{11}$$

where:

- $\mathbf{u} \in R^m$ is the control vector
- $\mathbf{x} \in R^n$ is the state vector
- $\mathbf{K} \in R^{m \times n}$ is the gain matrix obtained during designing LQR regulator

The formulation of the LQR control relies on the assumption, that gain matrix **K** is computed by minimization of the quadratic cost index defined on infinite time horizon, described by the equation 12.

$$J(\mathbf{K}) = \frac{1}{2}\int_0^\infty (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})dt = \frac{1}{2}\int_0^\infty \mathbf{x}^T(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K})\mathbf{x}dt \tag{12}$$

where:

- $\mathbf{Q} \in R^{n \times n}$ is a diagonal semi-positive defined weight matrix
- $\mathbf{R} \in R^{m \times m}$ is a diagonal positive defined weight matrix

The cost function, described in the equation 12, is minimized with respect to the gain matrix **K**. Additionally, the **Q** and **R** diagonal matrices can be modified by the control system designer in order to modify the cost function and to differentiate the impact of the particular state vector and control vector elements on the cost function value. Consequently, the choice of the weights in **Q** and **R** is crucial in order to provide satisfying control action in the designed system.

There exists an analytical solution of the presented optimization problem, which transforms the minimization of the integral cost function to the solution of the Algebraic Ricatti Equation (ARE), presented in the equation 13.

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q} = 0 \tag{13}$$

The analytical solution to the LQR problem is the gain matrix that satisfies condition 12, given in the form $\mathbf{K} = -\mathbf{R}^{-1}\mathbf{B}\mathbf{P}$, where the matrix **P** is obtained from solving the algebraic Riccati equation 13. The existence of an analytical solution to the LQR problem for such a defined case allows for a simple and effective method of designing control laws for MIMO-type dynamic systems. The first numerical algorithm for solving the ARE was proposed by David Kleinman in 1968 in the Ref. 11. However, nowadays many computational packages offer built-in solvers for the Riccati equation, such as the `K = lqr(A, B, Q, R)` function available in MATLAB.

The described method is dedicated only to dynamic systems in which, similarly to the pole placement method, full measurement of the instantaneous state vector x is possible. It requires the system to be fully controllable and observable, which further limits its practical applicability (Ref. 12). Its main drawback is that the LQR method is dedicated to linear systems, which means that applying it to nonlinear system control is not optimal. Nevertheless, the LQR controller remains a popular choice in the synthesis of control laws for automatic flight control systems, which is exemplified by the Ref. 13 and Ref. 14.

The method allows for effective and simple implementation in automatic flight control systems. Its performance strongly depends on the selection of the diagonal elements of the matrices **Q** and **R**, whose values allow the designer to weight the influence of individual components of the state and control vectors on the cost function value 12. The selection of these values is usually based on the experience of the autopilot system designer. However, heuristic optimization algorithms, such as the Particle Swarm Optimization (PSO) algorithm or the Bee Algorithm, can be used to determine the weighting values, which constitutes a modern approach to the problem. The presented framework is implemented in the Ref. 15.

It is also possible to extend the applicability of the LQR method to systems in which full observability is ensured, but measuring all components of the state variables is not feasible. This relies on using a Kalman filter as a state observer within the feedback loop of the dynamic system, which enables effective implementation of the LQR controller. The approach that combines an LQR controller with a Kalman filter acting as a state observer is known as LQG and can be used in various applications, including aerospace systems presented in the Ref. 16 and Ref. 17.

Particle Swarm Optimization tuning

Heuristic optimization algorithms are methods inspired by natural or social processes that find approximate solutions to

optimization problems, especially when traditional methods fail. They are used in cases where the objective function is highly nonlinear, discontinuous, or has many local extrema. These algorithms aim to find approximate solutions to optimization problems through an iterative process, where the main task is to explore the domain of the cost function using intuitive strategies.

In contrast to classical gradient-based methods, heuristic optimization algorithms operate on a set of potential solutions rather than a single point in the domain of the cost function. Another key difference is that heuristic algorithms use probabilistic methods to process candidate solutions, while gradient-based methods use deterministic techniques. It is also worth mentioning that heuristic optimization methods require only the computation of the cost function value $J(x_i)$, and not its gradient or the inverse of the Hessian matrix.

There are many algorithms that fall under the philosophy of heuristic methods. These include, among others, evolutionary algorithms, which are popular in engineering and design. Their operation is inspired by natural selection, crossover, and mutation, which are processes occurring in nature to ensure the survival of organisms best adapted to their environment.

A modern heuristic optimization algorithm is the Particle Swarm Optimization algorithm (PSO), developed in 1995 and introduced in the Ref. 18, which emerged during simulations of social behaviors conducted by James Kennedy and Russell Eberhart. The logic behind the simulated social mechanism is based on the observation that three main factors influence human decision-making:

- a component reflecting each individual's personal experience,
- a component representing decisions similar to those made by the most successful individuals,
- a random component simulating luck or unpredictable life events.

This very general decision-making scheme can conceptually describe the behavior of animal swarms searching for food or suitable environmental conditions for survival.

Applying this simple decision-making model can be highly effective in finding approximate solutions to optimization problems. The algorithm conceptually simulates the search of a cost function's domain by a finite number of particles, whose positions are updated at each iteration according to the principles listed above. A more detailed computational scheme of the algorithm is presented below.

1. **Initialization**: A set of N particles is created and randomly distributed within the feasible solution space. Each particle n has a position $\mathbf{x}_n$ and a velocity $\mathbf{v}_n$.
2. **Fitness Evaluation**: For each particle, the value of the cost function $J(\mathbf{x}_n)$ is calculated, determining the quality of its position.
3. **Update of Best Values**:
   - If the new position $\mathbf{x}_n$ is better than the particle's previous best position $\mathbf{p}_n$, then $\mathbf{p}_n$ is updated.
   - If the new position $\mathbf{x}_n$ is the best among all particles, then the global best position $\mathbf{g}$ is updated.
4. **Stopping Condition**: Check whether the best position $\mathbf{g}$ satisfies the stopping criterion or whether the maximum number of iterations has been reached.
5. **Velocity and Position Update**: If the stopping condition is not met, each particle's velocity and position are updated using the following equations:

$$\mathbf{v}_n^{(i+1)} = \sigma_{in}\mathbf{v}_n^{(i)} + c_1 r_1 \cdot (\mathbf{p}_n - \mathbf{x}_n^{(i)}) + c_2 r_2 \cdot (\mathbf{g} - \mathbf{x}_n^{(i)}) \quad (14)$$

$$\mathbf{x}_n^{(i+1)} = \mathbf{x}_n^{(i)} + \mathbf{v}_n^{(i+1)} \quad (15)$$

   where:
   - σ – inertia coefficient,
   - c_1, c_2 – attraction coefficients toward the particle's best-known solution and the global best solution,
   - r_1, r_2 – random numbers in the range $[0, 1]$,
   - $\mathbf{p}_n$ – best position found so far by particle n,
   - $\mathbf{g}$ – best global position found so far by the swarm.

Steps 2–5 are repeated iteratively until the algorithm terminates upon meeting the stopping criterion. PSO is characterized by its simplicity of implementation and broad applicability in global optimization.

By analyzing the velocity update formula 14, one can observe an analogy to a group decision-making model:

- $c_1 r_1 \cdot (\mathbf{p}_n - \mathbf{x}_n^{(i)})$ models the influence of each individual's personal experience on the optimization direction of particle n,
- $c_2 r_2 \cdot (\mathbf{g} - \mathbf{x}n^{(i)})$ models the influence of the most successful individual (i.e., the one that found the best cost function value) on the optimization direction of particle n,
- the term $\sigma_{in}\mathbf{v}_n^{(i)}$ introduces an inertia component that determines the impact of the previous velocity on the next velocity. This component was proposed in 1998 to improve the convergence of the computational procedure, which is precisely described in the Ref. 19.

The algorithm's flowchart is presented in Figure 2.

The PSO algorithm is a very effective tool for finding approximate solutions to optimization problems characterized by strong nonlinearity of the cost function. It provides an effective mechanism for exploring the domain of a minimized

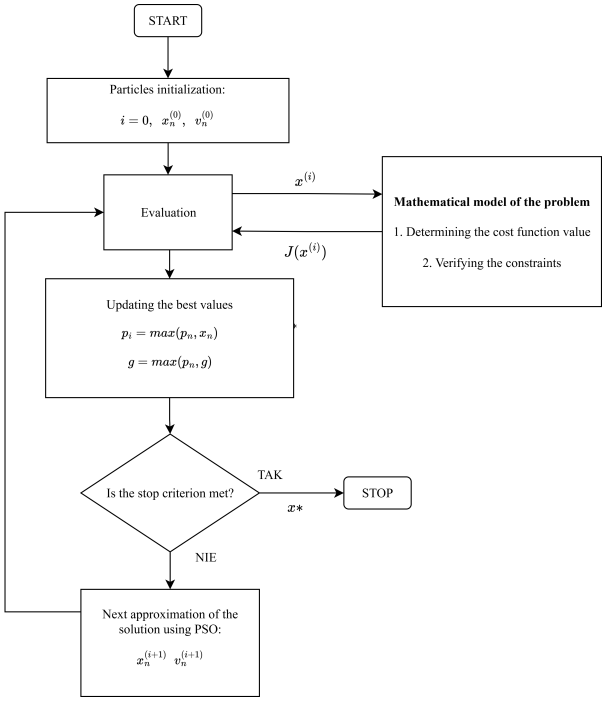

Figure 2. Computational flowchart of the PSO algorithm

function, but it is not well suited for rapidly solving local optimization tasks. To utilize optimization in design, a good solution might be to use a hybrid algorithm. In such an approach, the first stage searches for an approximate minimum of the cost function, while the second stage applies classical gradient-based methods to determine an accurate solution to the posed problem. The formulation of the modified algorithm is given in the Ref. 20.

DESIGNED CONTROL SYSTEM

Flight Control System Structure for the ARCHER Unmanned Helicopter

The structure of the automatic flight control system for the ARCHER unmanned helicopter is based on a single feedback loop, in which the controller, operating in state feedback configuration, implements the control law designed using the LQR method 11.

The state feedback loop in the proposed control system is realized via an IMU sensor, which consists of three inertial accelerometers and three MEMS gyroscopes. The measurement unit allows for the acquisition of three acceleration components in the sensor-fixed coordinate system and three angular velocity components, which enables the estimation of the full set of state variables described by Equation 16:

$$\mathbf{x} = [X, Y, Z, \Phi, \Theta, \Psi, V_x, V_y, V_z, P, Q, R]^T \tag{16}$$

where:

- $[X, Y, Z]^T$ – helicopter position in the inertial reference frame
- $[\Phi, \Theta, \Psi]^T$ – helicopter orientation angles
- $[V_x, V_y, V_z]^T$ – linear velocities in the body-fixed reference frame
- $[P, Q, R]^T$ – angular velocities of the helicopter motion

The input signal to the system is the vector $\mathbf{r}_p$, which contains the reference trajectory signals in the state space for the controlled helicopter. In the next step, the system computes the difference between the reference signal and the estimated state, and the resulting control error is fed into the LQR-designed control law.

The control signal generated by the LQR controller is then applied to the helicopter actuators, which activate the control effectors of the aircraft. The block diagram of the proposed automatic flight control system is shown in Figure 3.

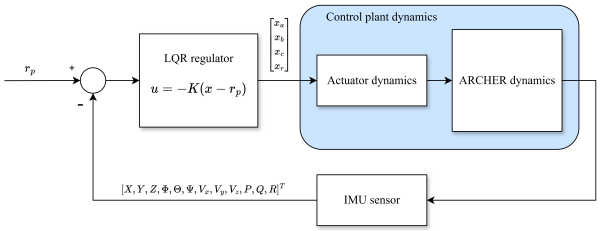

Figure 3. Block diagram of the proposed automatic flight control system for the helicopter

Description of LQR Controller Tuning

LQR controller tuning refers to the process of selecting the elements of the **Q** and **R** matrices appearing in the cost function defined in equation 12. For the linear model of the helicopter dynamics 9, 10, the number of considered state variables n is 12, while the number of control signals m is 4. Assuming that the matrices $\mathbf{Q} \in R^{n \times n}$ and $\mathbf{R} \in R^{m \times m}$ are symmetric and positive definite, the tuning task consists of selecting a total of 16 weighting coefficients on the diagonals of the matrices **Q** and **R**. Example matrices **Q** and **R** are shown in equation 17.

$$\mathbf{Q} = \begin{bmatrix} q_1 & 0 & \cdots & 0 \\ 0 & q_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & q_{12} \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} r_1 & 0 & 0 & 0 \\ 0 & r_2 & 0 & 0 \\ 0 & 0 & r_3 & 0 \\ 0 & 0 & 0 & r_4 \end{bmatrix} \tag{17}$$

The control law determined by the LQR method guarantees minimization of the cost function 12 for the system dynamics described by the linear model. However, it should be noted that real control systems cannot implement control signals of unlimited magnitude, which is not accounted for in the classical LQR theory. Minimizing the cost function corresponding solely to the time-domain control error integral would result in applying control signals with infinite amplitude, which is not feasible in practice. Therefore, correct reasoning requires incorporating actuator saturation into the optimization process.

To simplify further reasoning, we introduce a vector of sought weights Θ, defined by equation 18:

$$\Theta = \begin{bmatrix} \text{diag}(\mathbf{Q}) \\ \text{diag}(\mathbf{R}) \end{bmatrix} \tag{18}$$

The tuning objective is to find a vector Θ^* such that for a given reference trajectory, the cost function $F(\Theta^*)$, defined by equation 20, is minimized. This optimization task is summarized in equation 19:

$$\Theta^* = \arg\min_{\Theta} F(\Theta) \quad \text{s.t.} \quad \Theta_i \in [0.01,\ 100000] \tag{19}$$

$$F(\Theta) = F_1(\Theta) + F_2(\Theta) \tag{20}$$

where:

$$F_1(\Theta) = \int_0^T \left[(X - X_{\text{ref}})^2 + (Y - Y_{\text{ref}})^2 + (Z - Z_{\text{ref}})^2\right] dt \tag{21}$$

$$F_2(\Theta) = \frac{180}{\pi} \int_0^T \left[(\Phi - \Phi_{\text{ref}})^2 + (\Theta - \Theta_{\text{ref}})^2 + (\Psi - \Psi_{\text{ref}})^2\right] dt \tag{22}$$

The above problem corresponds to controller performance optimization in the time domain, where the weights are selected to reduce the cumulative integral of the position and orientation errors. The scalar $\frac{180}{\pi}$ in $F_2(\Theta)$ converts radians to degrees, minimizing the impact of unit scaling on the cost function.

Due to the structure of the LQR cost function 12, its scaling by an arbitrary constant does not affect the location of the global minimum. As a result, the final form of the $\mathbf{K}$ matrix in the LQR control law depends more on the relative proportions of Θ elements than on their absolute values. To account for this property, it is necessary to fix one element of the vector Θ. We assume $\Theta_1 = 1000$, which reduces the optimization problem's dimension from 16 to 15.

To further reduce dimensionality, the $\mathbf{R}$ matrix may be simplified as in equation 23:

$$\mathbf{R} = \lambda_R \cdot \mathbf{I} \tag{23}$$

where $\mathbf{I} \in R^{m \times m}$ is the identity matrix, and λ_R is a scalar expressing the influence of control effort on the cost function 12, without differentiating among components. Searching for a single λ_R instead of four r_i values reduces the problem to 12 decision variables.

Given that the LQR control law is static and of the form $u = -\mathbf{K}(\mathbf{x} - \mathbf{x}_{\text{ref}})$, one might consider optimizing $\mathbf{K}$ directly instead of designing weights. However, for a 12-state, 4-input system, direct optimization would involve 48 variables, while tuning the weights involves only 12.

The PSO-based tuning process follows the procedure shown in Figure 4. It starts with an initial guess of weights and performs iterative simulation and cost evaluation:

- Analyze the helicopter's linear model:
  - Simulate LQR-controlled flight along a predefined reference trajectory over $t \in [0, T]$ for each particle's $\Theta^{(i)}$
  - Compute the cost function 20 for each scenario
- Check termination criteria:
  - If not met, update particle positions
  - If met, terminate optimization

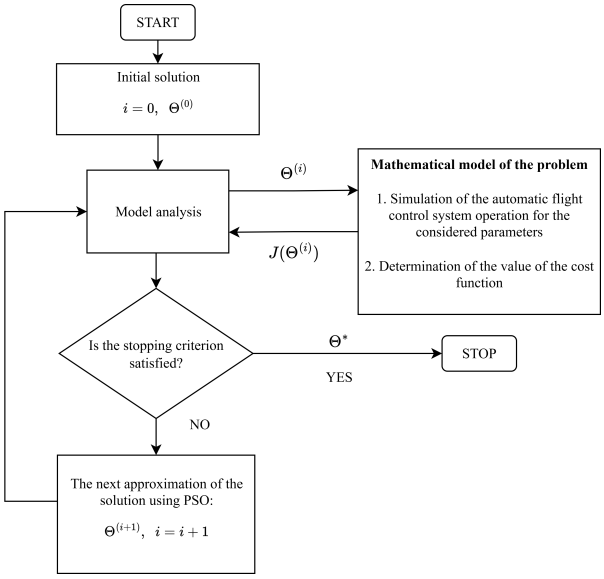

Figure 4. Diagram of LQR tuning using the PSO algorithm

The following parameters were selected based on PSO validation tests on the Rastrigin function:

- Number of particles: $N = 100$
- Maximum velocity: $V_{\max} = 1000$
- PSO parameters: $c_1 = 1.0$, $c_2 = 2.0$

The reference trajectory involves a step increase of 50 feet in X, Y, and Z and a 90° yaw rotation. Simulation duration is $T = 10\,\text{s}$ with time step $\Delta t = 0.01\,\text{s}$. This trajectory was selected to test near-hover behavior.

The maximum number of iterations was 2000. No absolute cost threshold was imposed due to lack of prior knowledge about minimum cost value.

The best global solution found during optimization is shown in Figure 5, with the x-axis presented on a logarithmic scale due to the large iteration count.

It can be observed that random particle initialization successfully led to a stable LQR controller. The most significant cost reduction occurred after the first 120 iterations. After reaching

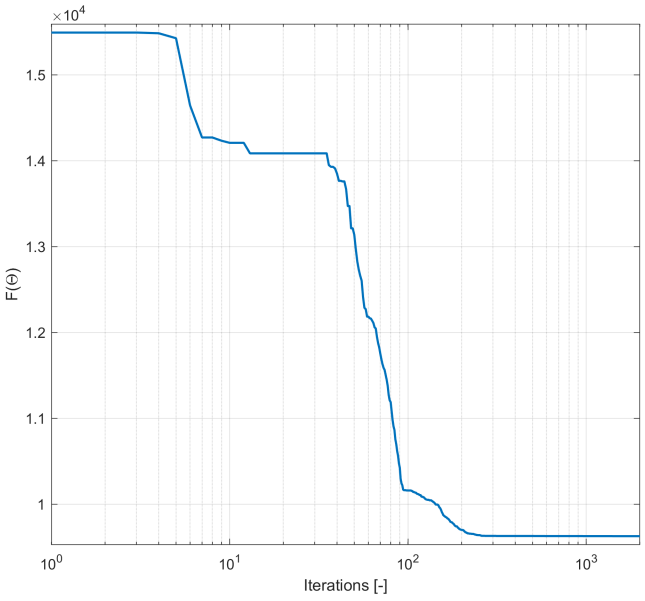

Figure 5. Global best cost value found by particles vs. number of iterations

Table 2. LQR controller weighting coefficients

Weight	Value
q_1	1000
q_2	1060.37
q_3	3918.16
q_4	51114.93
q_5	41596.53
q_6	73414.65
q_7	0.01
q_8	0.01
q_9	474.60
q_{10}	0.01
q_{11}	23882.16
q_{12}	38602.25
λ_R	35.99

approximately 210 iterations, the algorithm slightly improved the optimization solution.

The final optimized weights are presented in Table 2.

As seen in Table 2, the position-related weights along the X and Y axes are similar. The highest weights correspond to orientation angles, ensuring strong stabilization. The lowest weights are assigned to linear velocities, except vertical speed, and to angular velocity P.

SYSTEM VERIFICATION

Test cases

For the configuration of the automatic flight control system proposed in the article and the determined controller weights, simulations of the designed autopilot were carried out. To verify the system performance, several test scenarios were prepared as described below.

1. The test case includes a simulation of helicopter motion along a trajectory composed of step changes in position in the reference coordinate frame $O_n x_n y_n z_n$.
   - At time $T = 2$, a step change is applied to the $O_n x_n$ axis, corresponding to a climb of 40 feet (12,19 m).
   - At time $T = 10$, a step change is applied to the $O_n y_n$ axis, corresponding to a 40-foot (12,19 m) sideward motion.
   - At time $T = 20$, a step change is applied to the $O_n x_n$ axis, corresponding to a 40-foot (12,19 m) forward motion.
   - At time $T = 30$, a step change in the yaw angle Ψ of 90 degrees is introduced.
   - At time $T = 40$, another step change s applied to the $O_n x_n$ axis, corresponding to a 40-foot (12,19 m) forward motion.
2. The test case includes a simulation of step changes in position along the $O_n z_n$ axis while maintaining constant forward velocity along the $O_n x_n$ direction.
   - At time $t = 5\,\mathrm{s}$, the helicopter accelerates to a forward velocity of $V_x = 6\,\mathrm{m/s}$.
   - From $t = 5\,\mathrm{s}$ to $t = 25\,\mathrm{s}$, the helicopter performs alternating step changes in position along the $O_n z_n$ axis with an amplitude of 5 feet upwards and downwards.
3. The test case involves flight along a helical path, which in the $O_n x_n y_n z_n$ coordinate system, for $t \in [10, 50]$ $[s]$, can be parametrized by the set of equations listed below.

$$\begin{cases} x(t) = 10 \cdot sin\,(2\pi \cdot 0.075 \cdot (t-10))\ [ft] \\ y(t) = 10 \cdot cos\,(2\pi \cdot 0.075 \cdot (t-10))\ [ft] \\ z(t) = 5 \cdot t\ [ft] \end{cases} \tag{24}$$

The initial conditions for all three simulations are presented below.

$$\mathbf{x}_0 = [X_0, Y_0, Z_0, \Phi_0, \Theta_0, \Psi_0, U_0, V_0, W_0, P_0, Q_0, R_0]^T \tag{25}$$

where:

$$\begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -3.43 \end{bmatrix} [\mathrm{m}], \quad \begin{bmatrix} \Phi_0 \\ \Theta_0 \\ \Psi_0 \end{bmatrix} = \begin{bmatrix} 0.0698 \\ -0.0087 \\ 0 \end{bmatrix} [\mathrm{rad}] \tag{26}$$

$$\begin{bmatrix} U_0 \\ V_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [\mathrm{m/s}], \quad \begin{bmatrix} P_0 \\ Q_0 \\ R_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} [\mathrm{rad/s}] \tag{27}$$

$$\mathbf{u}_0 = [x_{a_0}, x_{b_0}, x_{c_0}, x_{r_0}]^T \tag{28}$$

where:

$$\begin{bmatrix} x_{a_0} \\ x_{b_0} \\ x_{c_0} \\ x_{r_0} \end{bmatrix} = \begin{bmatrix} 50.3 \\ 50.4 \\ 64.5 \\ 59.4 \end{bmatrix} \; [-] \tag{29}$$

Results

Case One In the first case, step responses of the automatic flight control system can be observed in the position control channels in the $O_n x_n y_n z_n$ coordinate system and in the yaw angle Ψ channel. The evolution of the controlled variables for Case One is presented in Figure 6.

As exemplified in Fig. 6, step changes in position and yaw angle in Case 1 result in marginal overshoot, and the position control exhibits a settling time of approximately 4 seconds. When altitude increases due to a strong increase in the main rotor collective pitch, a significant change in the yaw angle (up to $\Psi = 55°$) can be observed. This is caused by a sudden increase in the rotor torque. The tail rotor compensates for the yaw deviation within approximately 2 seconds.

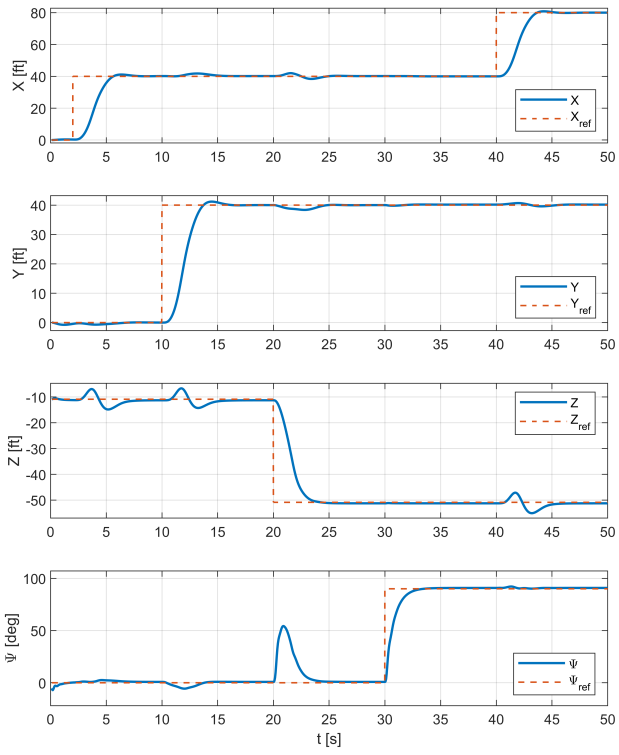

Figure 6. Time evolution of controlled state variables for Case One

Figure 7 shows the control signals over time. The sudden increase in altitude for time $t > 20\ s$ causes saturation of the actuator controlling the collective pitch.

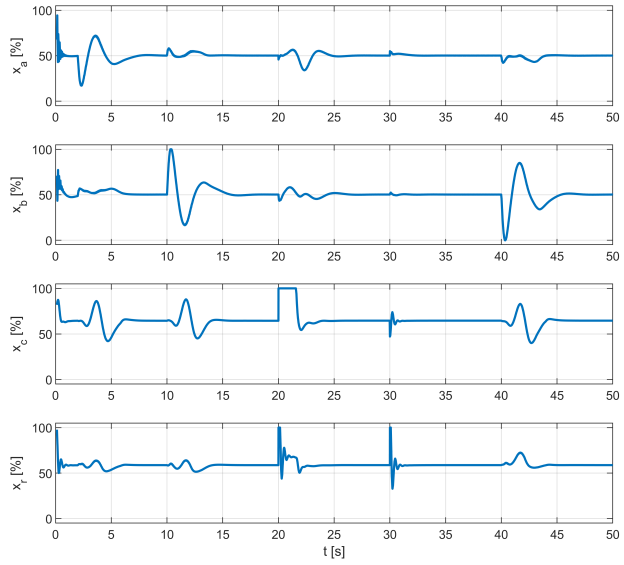

Figure 7. Time evolution of control signals for Case One

Figure 8 shows the helicopter's flight trajectory in three dimensions. Step changes along the $O_n x_n$ and $O_n y_n$ axes cause a temporary drop in altitude, which is compensated over time by adjusting the main rotor collective pitch. The change in altitude also contributes to a shift of the helicopter within the $O_n x_n y_n$ plane. Due to the helicopter's rotation for $t > 30, \mathrm{s}$ and the change in its position along the $O_n x_n$ axis after $t > 40, \mathrm{s}$, the rotorcraft flies sideways while still correctly performing its task.

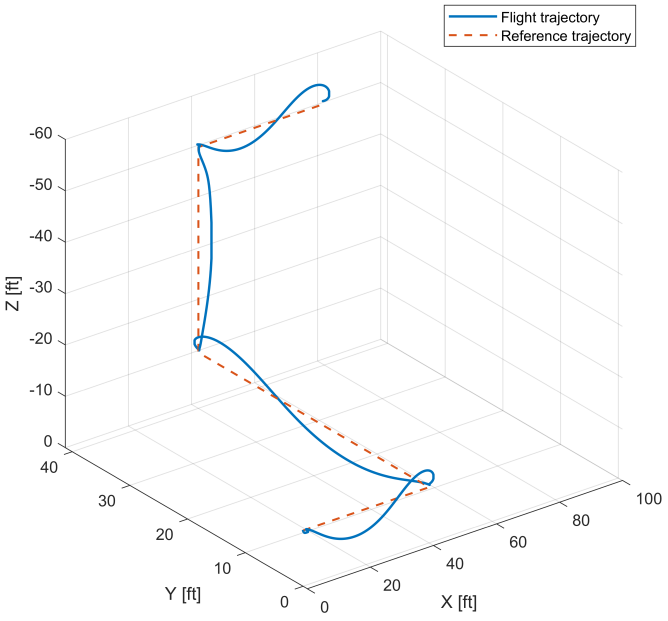

Figure 8. Three-dimensional helicopter flight trajectory for Case One

Case Two Figure 9 shows the helicopter's flight trajectory in response to a step input increasing forward flight speed to $Vx = 20\ ft/s$. In the next step, after accelerating the aircraft, the helicopter responded to a step change in position along the $O_n z_n$ axis with an amplitude of 5 ft.. The helicopter successfully reached the desired speed, however, the acceleration maneuver introduced an unbalanced $O_n y_n$ position steady-state error of approximately 1 ft and a yaw angle error of about 5 degrees.

The helicopter followed the reference position trajectory along the $O_n z_n$ axis; however, due to flight conditions differing from a perfect hover, the altitude response appears to be consistently shifted upward by approximately 1 ft.

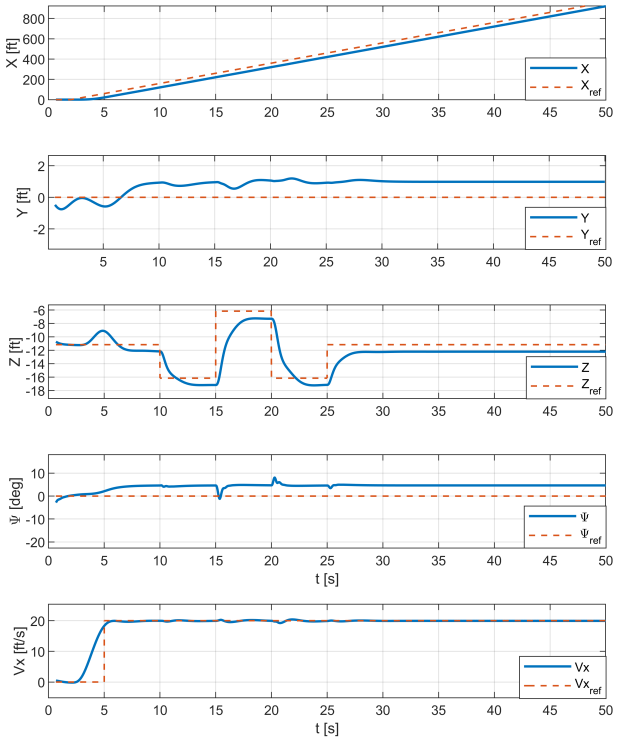

Figure 9. Time evolution of controlled state variables for Case Two

Figure 10 presents the control signals for Case Two. The executed maneuvers required aggressive adjustments of the collective pitch. The characteristic spikes reflect sudden changes in collective pitch applied to increase the thrust of the main rotor and rapidly change the rotorcraft's altitude.

Figure 11 shows the three-dimensional trajectory of the simulated motion in Case Two. A steady-state error in altitude is observed during acceleration along the $O_n x_n$ axis.

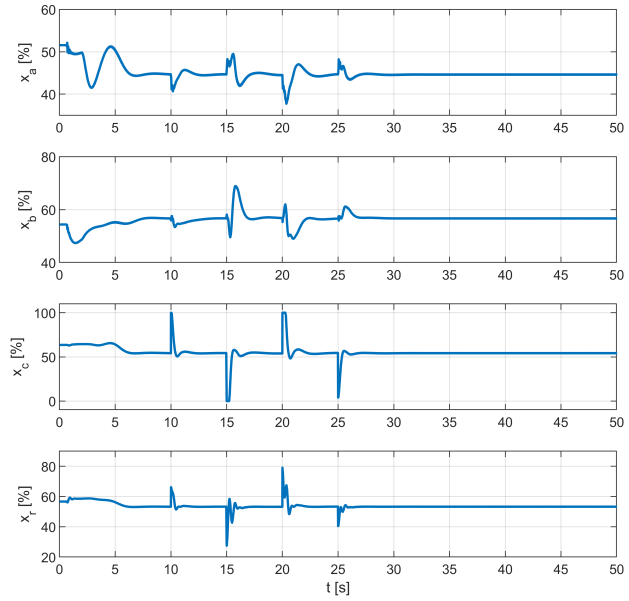

Figure 10. Time evolution of control signals for Case Two

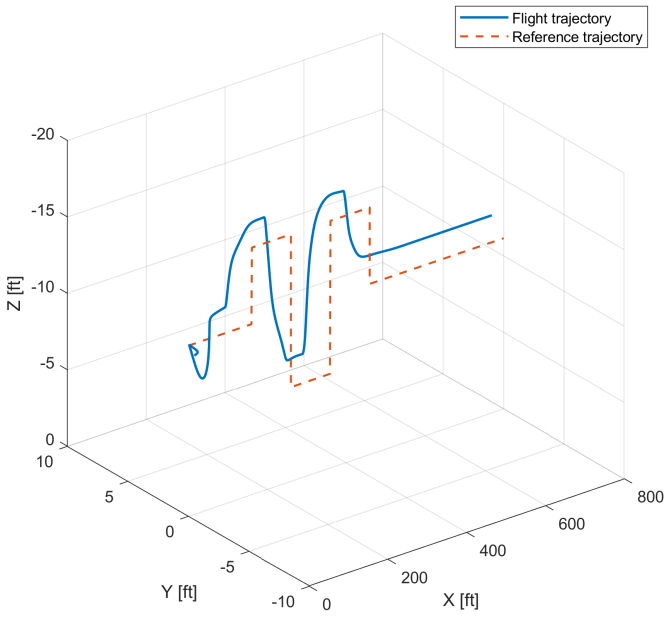

Figure 11. Three-dimensional helicopter flight trajectory for Case Two

Case Three Figure 12 shows the evolution of the directly controlled state signals in Case Three. During the execution of the helical flight path, a phase lag in the control of the position along the $O_n x_n$ and $O_n y_n$ axes was observed. The control system maintained a fixed yaw angle Ψ and increased the altitude in accordance with the reference signal.

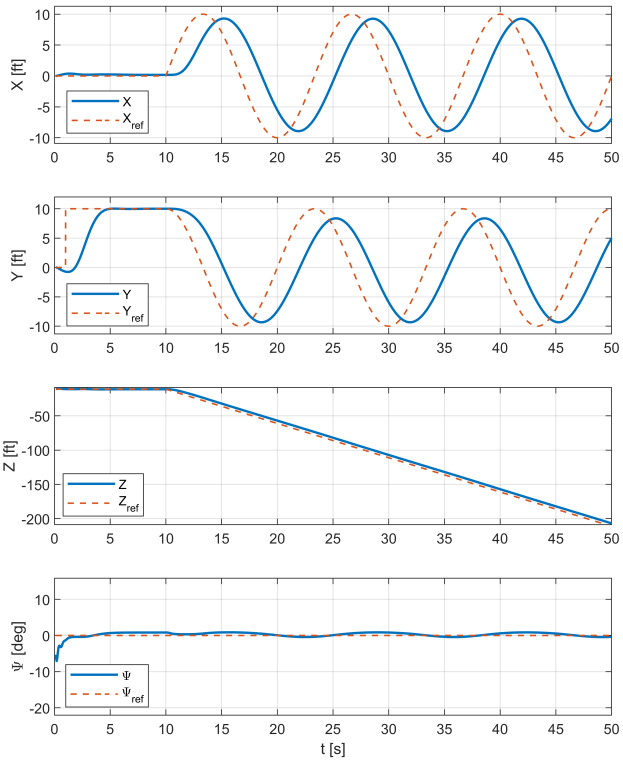

Figure 12. Time evolution of controlled state variables for Case Three

Figure 13 shows the control signals for Case Three. Increasing the flight velocity along the $O_n z_n$ axis required an increase in collective pitch, which is observed at $t = 10, \mathrm{s}$.

Figure 14 shows the helicopter's three-dimensional flight trajectory for Case Three. The helicopter successfully executed the predefined flight path. As shown in the figure, the rotorcraft's trajectory exhibited a slightly reduced radius, which may be attributed to the introduced phase lag and control limitations.

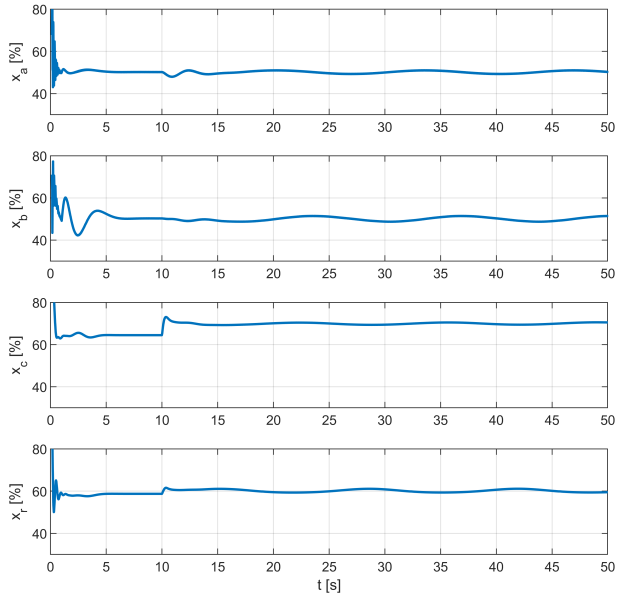

Figure 13. Time evolution of control signals for Case Three

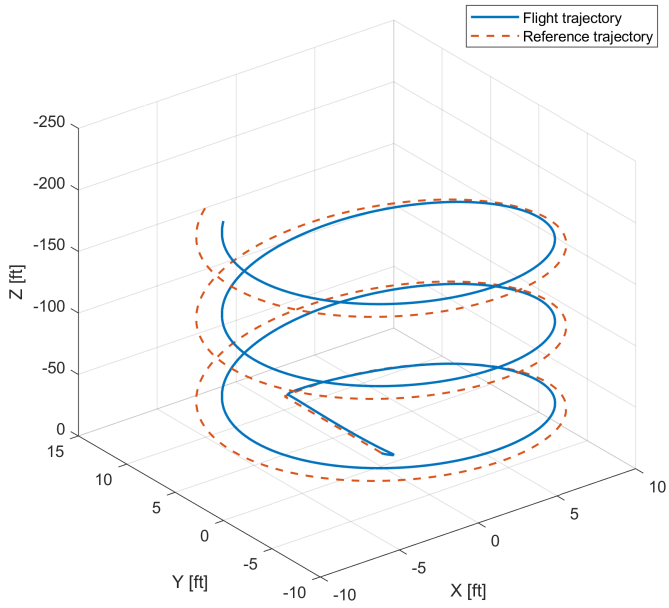

Figure 14. Three-dimensional helicopter flight trajectory for Case Three

CONCLUSIONS

This paper presents a framework for the automatic flight control system of an unmanned helicopter based on the operation of a Linear Quadratic Regulator (LQR). The weighting coefficients in the definition of the linear-quadratic problem were selected automatically using a Particle Swarm Optimization (PSO) algorithm to minimize a cost function describing system performance in the time domain.

To evaluate the performance of the designed automatic control system, simulation tests were carried out as described in the previous chapter. The conclusions drawn from the analysis are presented below.

- The use of the PSO algorithm allowed for effective tuning of the LQR controller for unmanned helicopter control. The determined numerical coefficients constitute an approximate solution to the optimization problem defined in Section 6.2.
- The quality of the response of the helicopter automatic flight control system strongly depends on the constraints of the control signal. The actuators that control the blade pitch angles of the main and tail rotors operate within limited ranges, which directly affects the helicopter's flight behavior. Effective LQR controller tuning via the inverse design approach requires these limitations to be considered in the cost function being minimized.
- The system's step responses, reflecting helicopter behavior in near-hover conditions, showed no overshoots and exhibited satisfactory settling times. However, during level flight at a constant speed, the system's performance was weaker. This is due to two main reasons:
    1. The LQR controller, in its basic formulation, is a static control law derived for a linear model of helicopter dynamics. In the presented case, the approximate mathematical model was obtained for hover conditions, which means that it inaccurately represents helicopter behavior during level flight.
    2. The LQR controller was tuned using the PSO algorithm by minimizing a cost function corresponding to the sum of tracking errors for step changes in position and yaw angle. The test scenario in the cost function did not include level flight, which reduces the autopilot's effectiveness under such conditions.
- The LQR controller does not include integral action in the feedback loop, which explains the occurrence of steady-state errors in altitude and yaw angle in the second test case. It is possible to modify the classic LQR controller by adding integral action to the tracking error, as described in detail in Ref. 21.

The control law proposed in this paper enables effective state regulation of the helicopter in near-hover conditions, which limits the applicability of the system. To improve the operation of the automatic control system during level flight, the concept of Parallel Distributed Compensation (PDC) can be employed. This method involves designing multiple controllers for different operating conditions and intelligently combining multiple control laws to manage the helicopter's flight at various speeds. To merge the outputs of multiple parallel controllers, Takagi-Sugeno fuzzy inference can be applied (like in Ref. 22 and Ref. 23), which offers an effective solution to the stated problem.

In this paper, the PSO algorithm was used to tune the weights of the LQR controller. Further research may focus on applying the proposed inverse design concept to the tuning of other types of controllers, such as PID structures or Model Predictive Controllers (MPC). Apart from the minimizing cost function dependent on time domain performance, PSO can be used to minimize H_∞ norm, which is discussed in the Ref. 24.

Author contact:
Łukasz Kiciński: lukasz.kicinski2.stud@pw.edu.pl
Sebastian Topczewski: Sebastian.Topczewski@pw.edu.pl

REFERENCES

1. Shim, D., Kim, H.J., and Sastry, S., "Control system design for rotorcraft-based unmanned aerial vehicles using time-domain system identification," *Proceedings of the American Control Conference*, Chicago, IL, June 2000, pp. 808–813.
2. Gadewadikar, J., Chen, B., and Subbarao, K., "Structured H∞ Command and Control-Loop Design for Unmanned Helicopters," *Journal of Guidance, Control, and Dynamics*, Vol.31, July 2008, pp. 1093–1102.
3. Chen, Y., Wang, X., Lu, G., and Zhong, Y., "Modeling and LQR control of small unmanned helicopter," *Proceedings of the 32nd Chinese Control Conference*, 2013, pp. 4301–4305.
4. Jan, S., and Lin, Y., "Integrated Flight Path Planning System and Flight Control System for Unmanned Helicopters," *Sensors*, Vol. 11, No.8, 2011, pp. 7502–7529.
5. Kadmiry, B., and Driankov, D., "Fuzzy Control of an Autonomous Helicopter," *IFSA World Congress and 20th NAFIPS International Conference*, Vancouver, BC, Canada, July 2001.
6. Gong, X., "Model Predictive Control for an Unmanned Helicopter," *2019 IEEE International Conference on Unmanned Systems (ICUS)*, 2019, pp. 361–366.
7. Bedyńska, A., Curyło, P., Czarnota, N., Ceniuk, M., Grabowski, Ł., Łukasiewicz, M., and Bibik, P., "ARCHER - Autonomous Reconfigurable Compound Helicopter for Education and Research," *45th European Rotorcraft Forum*, Warsaw, Poland, Sep. 17-20, 2019.

8. Żugaj, M., Edawdi, M., Iwański, G., Topczewski, S., and Bibik, P., "An Unmanned Helicopter Energy Consumption Analysis," *Energies*, Vol.15, No.16, 2022.

9. Padfield, G. D., *Helicopter Flight Dynamics*, John Wiley Sons, Ltd, 2007, Chapter 3.

10. Peters, D.A., and He, C., "Correlation of Measured Induced Velocities with a Finite-State Wake Model," *Journal of The American Helicopter Society*, Vol.36, 1991, pp. 59–70.

11. Kleinman, D., "On an iterative technique for Riccati equation computations," *IEEE Transactions on Automatic Control*, Vol.13, No.1, February 1968, pp. 114–115.

12. McLean, D., *Automatic Flight Control Systems*, Prentice Hall International, Cambridge, 1990.

13. Yang, Y., "Analytic LQR Design for Spacecraft Control System Based on Quaternion Model," *Journal of Aerospace Engineering*, Vol.25, July 2012, pp. 448–453.

14. Dul, F., Lichota, P., and Rusowicz, A., "Generalized Linear Quadratic Control for a Full Tracking Problem in Aviation," *Sensors*, Vol.20, 2020.

15. Maghfiroh, H., Hizam, M., Anwar, M., and Ma'Arif, A., "Improved LQR Control Using PSO Optimization and Kalman Filter Estimator," *IEEE Access*, Vol.10, 2022, pp. 18330–18337.

16. Topczewski, S., and Bibik, P., "LQR and LQG control of the helicopter during landing on the ship deck," *Aircraft Engineering and Aerospace Technology*, Vol.95, 2023, pp. 1–9.

17. Lichota, P., Dul, F., and Karbowski, A., "System Identification and LQR Controller Design with Incomplete State Observation for Aircraft Trajectory Tracking," *Energies*, Vol. 13, 2020, pp. 1–28.

18. Kennedy, J., and Eberhart, R., "Particle swarm optimization," *Proceedings of ICNN'95 - International Conference on Neural Networks*, Vol.4, 1995, pp. 1942–1948.

19. Shi, Y., and Eberhart, R., "A modified particle swarm optimizer," *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 1998, pp. 69–73.

20. Junqiang, W., Aijia, O., and Libin, L., "A Self-Adaptive Hybrid Algorithm of PSO and BFGS Method," *2012 International Conference on Industrial Control and Electronics Engineering*, 2012, pp. 1690–1693.

21. Young, P.C., and Willems, J.C., "An approach to the linear multivariable servomechanism problem," *International Journal of Control*, 1972.

22. Takagi, T., and Sugeno, M., "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-15, No. 1, 1985, pp. 116–132.

23. Wang, H.O., Tanaka, K., and Griffin, M., "Parallel distributed compensation of nonlinear systems by Takagi-Sugeno fuzzy model," *Proceedings of 1995 IEEE International Conference on Fuzzy Systems*, Vol.2, 1995, pp. 531–538.

24. Bouallègue, S., Haggege, J., Benrejeb, M. - "Particle Swarm Optimization-Based Fixed-Structure H-infinity Control Design", *International Journal of Control, Automation and Systems*, Vol. 9, 2011, pp. 258-266,