

SESSION D: AUTOMATIC CONTROL, IT AND ARTIFICIAL INTELLIGENCE**Intelligent control of a UAV using a sensor glove**Simona Tosheska Radovic^{1*}, Dushko Stavrov¹, Gorjan Nadzinski¹, Vesna Ojleska Latkoska¹¹ Faculty of Electrical Engineering and Information Technologies, "Ss. Cyril and Methodius" University in Skopje, Skopje, Republic of North Macedonia**ARTICLE INFO*** Correspondence: simona.tosheska@gmail.comDOI: <https://doi.org/10.46793/ET26.D02TR>

PAGES: D11 – D22

ABSTRACT

This paper presents the development and simulation of an intelligent system for controlling a drone through hand gestures, evaluated and tested in the MATLAB/Simulink environment. First, a detailed mathematical model of the drone dynamics is built, including aerodynamic forces, motor dynamics and battery model. For precise estimation of the system states the model also incorporates the sensor noise, processed with an Unscented Kalman filter. Flight stabilization is achieved through a PID control system. The central contribution of the paper is an implementation of an intelligent control of a drone using a smart glove, equipped with flex sensors and an inertial measurement unit. The smart glove raw analog signals are processed and classified by a neural network with an 8–10–4 architecture, trained using the Levenberg-Marquardt algorithm. The network classifies the fused sensor signals into four discrete control commands, which are fed directly into the drone's flight control system, as structured reference inputs. The integrated system is verified through a simulation, where the drone successfully executes a three-dimensional trajectory, commanded exclusively by the hand gestures. The results show stable tracking of the reference signals, without oscillatory instability, confirming the practical applicability of the proposed approach.

KEYWORDS

Drone, Gesture control, Smart glove, Neural network, Kalman filter, PID controller, MATLAB/Simulink

1. INTRODUCTION

Drones, or quadcopters, have revolutionized various industries and have taken the public stage in many areas over the past few decades. The need for unmanned aerial vehicles (UAVs) with greater maneuverability and hovering capabilities compared to fixed-wing aircraft has led to a surge in research into drones. These drones have applications in diverse fields such as aerial photography, search and rescue operations, agricultural monitoring, and package delivery [1], [2]. Their popularity stems from their versatility, maneuverability, vertical takeoff and landing, and ability to access hard-to-reach areas in a safe and efficient manner [1]. However, the seemingly carefree flight of drones hides the complex engineering and control systems that enable their smooth operation.

One of the biggest challenges in drone operation is the efficient control, due to the inherent instability [3], [4]. The drone has six degrees of freedom, but only four controlled inputs, making it an underactuated system. This complexity requires highly robust and efficient control algorithms to maintain stability and achieve precise maneuvering. The process of developing an effective drone control system begins with a thorough understanding of its dynamics. In this paper, the drone dynamics are described using a mathematical model based on the physical and mathematical laws that govern drone operation.

Recent research in UAV control has increasingly moved beyond classical control methods toward intelligent, multimodal approaches, with machine learning and artificial intelligence playing a central role in advancing autonomous capabilities. [6], [7], [11], [13], [14], [15], [16] Gesture-based human-drone interaction has emerged as a particularly active research area, where control commands can be issued either through wearable sensor devices or through vision-based systems relying on cameras and computer vision algorithms, both removing the dependency on traditional remote controllers [13], [16], [22]. More advanced implementations further combine gesture recognition with obstacle avoidance and autonomous navigation, broadening the scope of intuitive human-machine interaction [25]. These developments collectively reflect a

clear trend toward intelligent, multimodal UAV control systems, which serves as the direct motivation for the research presented in this work.

The focus of this paper is the modeling of the drone dynamics and the development of a control system based on hand gesture recognition - an approach that uses the user's body as the control interface instead of a conventional remote control. For this purpose, a smart glove equipped with flex sensors on the fingers and an inertial measurement module on the hand will be realized, which in combination allow precise recording of the bending angle of each finger, the orientation of the hand, and its linear movements in real time. The collected sensor data is first denoised and normalized, and then fed into a neural network that classifies the gestures and converts them into appropriate navigation commands for the drone, such as pitch, roll, yaw, and throttle.

The entire system is implemented and verified in several interconnected stages. The first stage encompasses building a mathematical model based on the reconstructed states provided by the Kalman filter and designing a PID control system [28], [27]. The second stage is the physical realization of the smart glove containing five flex sensors [19] and IMU module [20] all integrated by Arduino Nano microcontroller. The third stage of collecting and labeling of training data for series of defined gestures is followed by training a neural network for classification, trained using the Levenberg-Marquardt algorithm [26]. The last stage is the integration of the individual stages in one fully operational system defined in MATLAB/Simulink. The end result is a fully integrated intelligent system in which each hand gesture is converted in real time into a specific control action of the aircraft, with verified stability and precision in a simulation environment.

2. SYSTEM MODELING

This section presents the system model through three interconnected parts: the dynamic model of the drone (Figure 1), the sensor noise model with estimation (Figure 3), and the controller design for stable reference tracking.

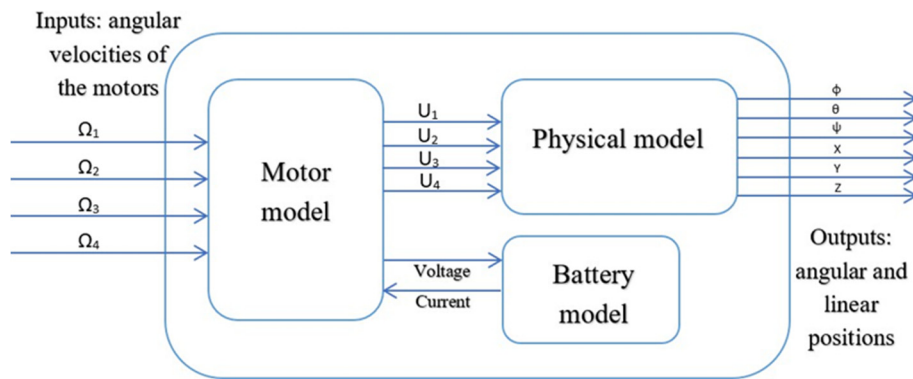


Figure 1: Block diagram of the drone dynamics model

2.1. Dynamic Drone Model

The drone has six degrees of freedom, translations along the X, Y and Z axes, as well as rotations described by the angles of roll (ϕ), pitch (θ) and yaw (ψ). The local NED (North-East-Down) frame is adopted as the reference coordinate system, in which the X axis points north, Y east, and Z down, normal to the XY plane (Figure 2). In this framework, a positive roll angle moves the drone to the east, a positive pitch angle to the south, and a positive yaw angle corresponds to a clockwise rotation.

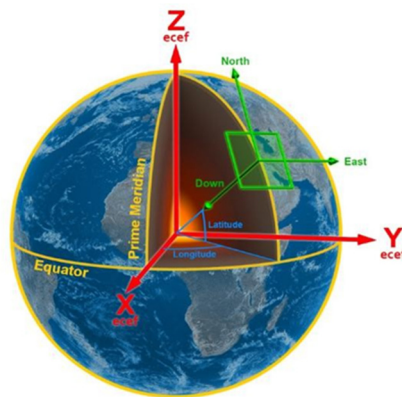


Figure 2: NED (North-East-Down) coordinate system

The model assumes that the drone is a rigid body with a symmetrical construction, that the centers of gravity and geometry coincide, and that the thrust force and drag force are proportional to the square of the angular velocity of the propeller. Aerodynamic air resistance is modeled as a force opposite to the direction of motion, proportional to speed, with a coefficient of 0.1 kg/s in each axis, a value suitable for simulation purposes [24].

2.2. Dynamics Equations

For an "X" frame configuration, the four control quantities, vertical thrust U_1 , roll U_2 , pitch U_3 and yaw U_4 , are defined as:

$$U_1 = b \cdot (\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \quad (1)$$

$$U_2 = b \cdot \sin\left(\frac{\pi}{4}\right) \cdot (\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \quad (2)$$

$$U_3 = b \cdot \sin\left(\frac{\pi}{4}\right) \cdot (\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \quad (3)$$

$$U_4 = d \cdot (\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2) \quad (4)$$

where Ω_i is the angular speed of the i -th engine, b is the thrust coefficient, and d is the resistance moment coefficient [17], [24]. The resulting angular velocity of the rotor is:

$$\Omega_r = \Omega_1 - \Omega_2 + \Omega_3 + \Omega_4 \quad (5)$$

The angular accelerations $\ddot{\phi}$, $\ddot{\theta}$ and $\ddot{\psi}$ along all three axes are obtained from the force-momentum equations, where each moment includes the gyroscopic effect of the body, the gyroscopic effect of the propeller, and the action of the actuator:

$$\ddot{\phi} = \left(\dot{\theta}\dot{\psi}(I_{yy} - I_{zz}) + J_r\dot{\Omega}_r + lU_2 \right) / I_{xx} \quad (6)$$

$$\ddot{\theta} = \left(\dot{\phi}\dot{\psi}(I_{zz} - I_{xx}) + J_r\dot{\Omega}_r + lU_3 \right) / I_{yy} \quad (7)$$

$$\ddot{\psi} = \left(\dot{\theta}\dot{\phi}(I_{xx} - I_{yy}) + U_4 \right) / I_{zz} \quad (8)$$

where $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ are angular velocities, I_{xx} , I_{yy} and I_{zz} are inertia moments, J_r is rotor inertia, l is rotor axis to copter center distance.

Linear accelerations $\ddot{X}$, $\ddot{Y}$ and $\ddot{Z}$ in the NED system are defined as:

$$\ddot{X} = \left((\sin\psi \sin\phi - \cos\psi \sin\theta \cos\phi) \cdot U_1 - A_x \dot{X} \right) / m \quad (9)$$

$$\ddot{Y} = \left((\cos\psi \sin\phi + \sin\psi \sin\theta \cos\phi) \cdot U_1 - A_y \dot{Y} \right) / m \quad (10)$$

$$\ddot{Z} = \left(mg - \cos\theta \cos\phi \cdot U_1 - A_z \dot{Z} \right) / m \quad (11)$$

where A_x , A_y and A_z is air resistance in each axis.

It is evident from equations (9) – (11) that the linear accelerations depend on the current orientation angles. By double integration of the acceleration, the velocity and position of the drone can be obtained at all times - which forms the basis for closing the control loop.

2.3. Sensor Modeling and State Estimation

Based on the derived equations of motion (9) – (11) [17], [24], a corresponding model was realized in the MATLAB/Simulink environment. The subsystem that models the dynamics of the drone has four inputs (Ω_1 – Ω_4) and six outputs (X , Y , Z , ϕ , θ , ψ), and internally consists of four interconnected subsystems that implement the thrust, angular and linear acceleration equations, with built-in motor and battery dynamics. To improve the interpretability of the model, the following constraints have been introduced: the Z coordinate is limited to negative values, the angular velocities of the rotor to the interval $[0, \infty)$, and the Euler orientation angles to the range $[-\pi, \pi]$.

In real conditions, the dynamic model of the drone does not accurately describe the actual behavior of the real system, due to the deliberate and necessary approximations introduced during modeling and the atmospheric conditions (wind, temperature etc.). The control approach is designed based on the feedback data provided by the Inertial Measurement Unit (IMU), which combines a gyroscope and an accelerometer. The gyroscope measures the angular velocities $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$, while the accelerometer measures the linear accelerations $\ddot{X}$, $\ddot{Y}$, $\ddot{Z}$ along all three axes (Figure 3). The sensor model in its general form can be represented as:

$$y = K_{bias} + LPF_{signal} + noise \quad (12)$$

where K_{bias} represents the bias factor, the LPF low-pass filter, and the last term corresponds to the measurement noise.

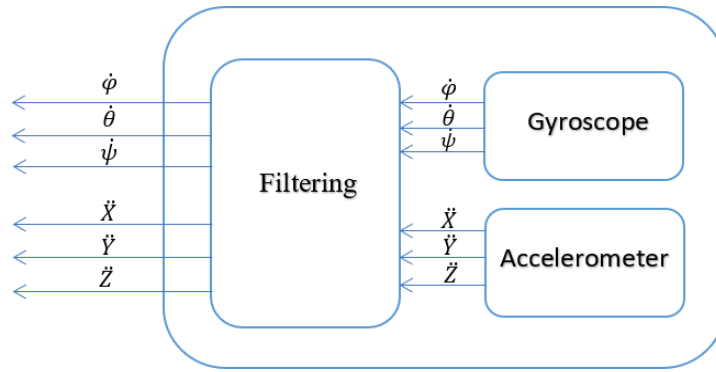


Figure 3: Block diagram of the sensor system

2.3.1. Unscented Kalman Filter

Due to the highly non-linear nature of drone dynamics, the Unscented Kalman filter (UKF) [28], [30] was chosen for state estimation. Unlike the extended Kalman filter (EKF) [29], [30], which relies on linearization via the Jacobian, the UKF uses the unscented transform (UT), a deterministic set of carefully selected sigma points to approximate the distribution of system states.

Algorithm Unscented_Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

- 1: $\mathcal{X}_{t-1} = (\mu_{t-1}, \mu_{t-1} + \gamma\sqrt{\Sigma_{t-1}}, \mu_{t-1} - \gamma\sqrt{\Sigma_{t-1}})$
- 2: $\bar{\mathcal{X}}_t^* = g(u_t, \mathcal{X}_{t-1})$
- 3: $\bar{\mu}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{X}}_t^{*[i]}$
- 4: $\bar{\Sigma}_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)(\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)^T + R_t$
- 5: $\bar{\mathcal{X}}_t = (\bar{\mu}_t, \bar{\mu}_t + \gamma\sqrt{\bar{\Sigma}_t}, \bar{\mu}_t - \gamma\sqrt{\bar{\Sigma}_t})$
- 6: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$
- 7: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$
- 8: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$
- 9: $\Sigma_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$
- 10: $K_t = \Sigma_t^{x,z} S_t^{-1}$
- 11: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$
- 12: $\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^T$
- 13: **return** μ_t, Σ_t

Figure 4: Unscented Kalman Filter algorithm [28], [30]

For a system of dimension n , $2n + 1$ sigma points are used. Each sigma point is independently propagated through the nonlinear system model, and the new predicted mean and covariance matrix are obtained as a weighted average of the transformed points. The distribution of sigma points is controlled by parameters α , β and κ : the parameter α determines the compactness of the distribution around the mean value, β takes into account the assumed error distribution ($\beta = 2$ for Gaussian distribution), while κ affects the range of sigma points in relation to the system dimension.

The key covariance matrices of the filter are: the process noise matrix Q , which reflects the uncertainty of the model; the measurement noise matrix R , which describes the reliability of the sensor; and the initial covariance matrix P_0 , which defines the initial uncertainty in knowing the state of the system. Balancing these matrices achieves an optimal compromise between filter response speed and noise immunity.

2.3.2. UKF Implementation in a Simulation Environment

In the Simulink model, the UKF block receives as inputs the discretized angular velocities and linear accelerations from the sensors, along with the relevant constants of the dynamic model. The discretization step is 0.001s. Corrected values of angular and linear velocities are obtained at the output. Filtered angular velocities are directly fed into the controller, while linear velocities undergo an additional transformation of the coordinate system before further use in controller.

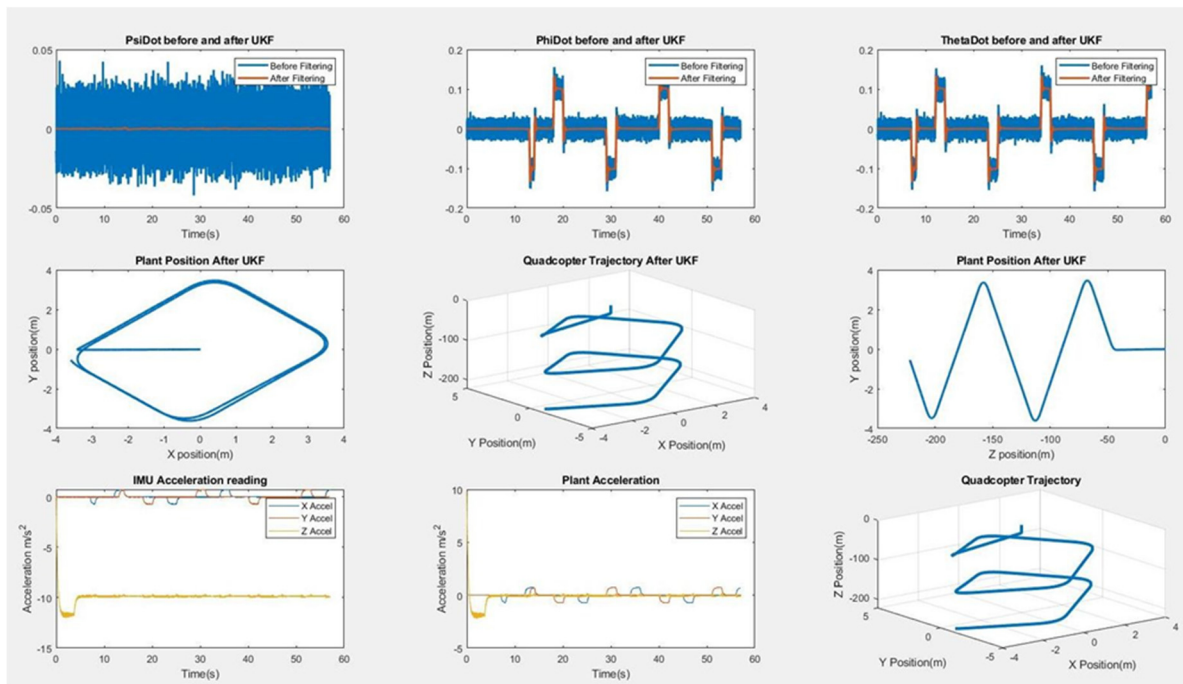


Figure 5: Simulation of a spiral movement results with using of UKF

The simulation results confirm the effectiveness of the applied approach (Figure 5). Unfiltered angular velocity signals contain significant noise that could potentially lead to instability of the control system, while UKF successfully eliminates noise and generates smooth, precise signals that closely follow the expected dynamics of the drone. The filtered orientation angles almost ideally match the reference values, which indicates a high estimation accuracy. The simulated three-dimensional trajectory, obtained at an aerodynamic drag coefficient of 0.1 kg/s, follows a circular path without significant deviations, confirming that the filter not only corrects measurement deviations but also ensures stability, predictability and robustness of the entire control system.

2.4. PID Controller

The controller is the central element of the closed-loop system whose task is to reduce the error between the reference and actual state values to zero. To stabilize the drone, a PID controller was used that simultaneously regulates four parameters: height (Z), roll angle (ϕ), pitch angle (θ) and yaw speed (ψ'). The control signal is formed as the sum of the proportional, integral and derivative term, where the proportional term provides general gain, the integral term eliminates the stationary error by accumulation, and the derivative term reacts to the rate of error change [27].

In order to avoid the impulse reaction of the derivative term at sudden changes of the reference signal, a PI-D structure is adopted, in which the derivative action is applied to the feedback state quantity and not to the error signal. The controller receives ten input signals, three linear accelerations, three angular velocities, and corresponding reference values and generates four motor angular velocities (Ω_1 – Ω_4) as outputs.

3. INTELLIGENT CONTROL OF THE DRONE

The principles of assigning commands to the drone can be divided into automatic and manual control. Automatic control relies on built-in algorithms and sensor systems that enable stabilization and trajectory tracking without direct operator intervention. Manual control is traditionally realized through an RC controller, where the operator directly controls the height, orientation and movement of the aircraft [5], [6], [7], [8].

Recently, approaches based on the recognition of human limb movements as a natural interface for drone control have attracted increasing interest. Hand gesture recognition stands out as one of the most current areas within the human-machine interface [13], [14], [15], [16]. Hand gesture recognition systems can be divided into two categories: visual systems based on computer vision and systems based on sensor measurements. Visual systems, while not requiring the wearing of additional devices, have high computational complexity, are sensitive to lighting conditions, and can introduce latency critical for real-time applications.

As a more efficient alternative, sensor approaches based on a smart glove with resistive bending sensors and an inertial measurement module (IMU) are being proposed, which directly measure the biomechanical parameters of the hand independently of external visual conditions, with lower latency and less computing resource requirements [13], [14], [19]. The integration of gesture control with machine learning algorithms opens up the possibility of creating an intuitive and adaptive interface in which hand movements are directly mapped into control commands, thereby reducing the operator's cognitive effort and improving human-aircraft interaction [5], [10], [11], [23].

3.1. Architecture and Hardware Realization of the Smart Glove

The smart glove is a wearable interface for gestural control of the aircraft in which hand movements are recorded through a multisensory architecture and transformed into control signals. The design integrates five resistive bending sensors to monitor the position of the fingers and an IMU module to detect the orientation and angular velocity of the hand (Figure 6). The goal is to ensure stable acquisition of biomechanical parameters with sufficient resolution and sampling frequency for application in a real-time system [14], [15], [16], [19], [20].

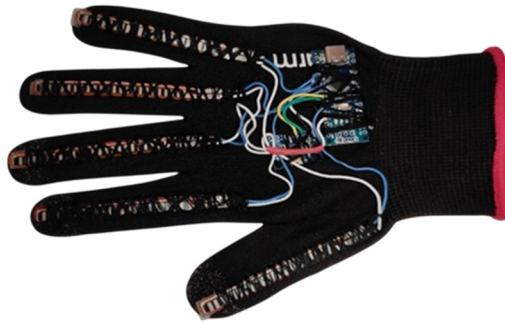


Figure 6: Smart glove

An Arduino Nano microcontroller based on an ATmega328P processor with a clock speed of 16 MHz is used for signal processing. The microcontroller reads analog values from the flex sensors via voltage dividers and communicates with the IMU module MPU6050 via the I²C protocol. The processed data is forwarded to a Simulink simulation, which serves as a virtual representation of the gesture classification processor.

Resistive flex sensors are placed on each finger of the glove. Each sensor changes resistance in proportion to the bend angle, and a voltage divider translates that change into a voltage signal that the microcontroller can process. For each sensor, two calibration values are defined, resistance at fully extended and fully flexed finger, which serve as reference points for normalizing the measured bending angle [14], [19].

The MPU6050 IMU module combines a three-axis accelerometer and a three-axis gyroscope in a single MEMS device with 16-bit analog-to-digital conversion per channel. The module provides continuous monitoring of the orientation and angular velocities of the hand in three-dimensional space, which, in combination with the bending sensor signals, enables precise detection of gestures in real time [3], [20], [22].

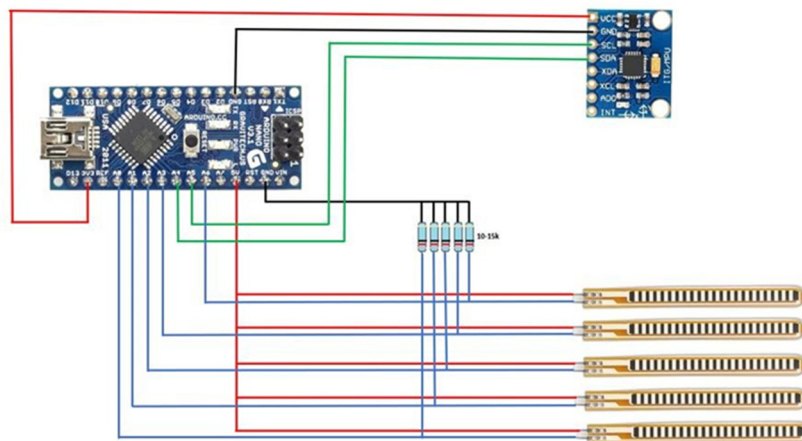


Figure 7: Electrical schematic of the sensor glove

3.2. Gesture Recognition Algorithm

The transformation of glove sensor signals into drone control commands requires an algorithm capable of modeling non-linear dependencies between the eight-dimensional measurement input into the four-dimensional control command space. The input vector consists of five normalized finger flexion values and three accelerometer components, while the control outputs correspond to the commands: roll, pitch, yaw and throttle. To realize this transformation, a multilayer feedforward neural network with an 8–10–4 architecture (8 inputs, 10 hidden and 4 output neurons), implemented in the MATLAB Neural Network Toolbox environment, was used [5], [9], [10], [11], [13].

The input layer receives eight numeric values from the glove sensor. The ten-neuron hidden layer uses a hyperbolic tangent activation function (tansig), while the four-neuron output layer uses a linear activation function (purelin), which is a standard configuration for regression tasks. The number of neurons in the hidden layer is determined empirically as a compromise between learning capacity and the risk of overfitting. Since the system performs instantaneous input-output

transformation without time dependencies, recurrent architectures are not required, and a network with one hidden layer is theoretically justified for approximating this class of nonlinear functions.

The Levenberg-Marquardt algorithm (trainlm) [26] was used for network training, which combines the advantages of gradient descent and Newton's method and is the standard and fastest algorithm for training networks with back propagation in the MATLAB environment. The error function to be minimized is the mean squared error (MSE). The data set was split randomly in the ratio 70/15/15% for training, validation and testing. The maximum number of epochs is set to 1000, and the early stopping criterion is activated after six consecutive epochs without improving the validation error, thus preventing overfitting (Figure 9 and 10).

The training dataset was formed by systematically recording sensor measurements for nine defined gestures: up, down, left, right, forward, back, left rotation, right rotation, and stop (Figure 8). Each gesture was performed multiple times with different intensities and speeds to capture natural variations in real use. Each sample represents an eight-dimensional feature vector assigned a corresponding control label, an annotation process that is a key step of supervised learning.



Figure 8: Defined hand commands (down, up, left, right, backward, forward, right rotation, left rotation, stop)

Training was completed after 29 epochs, with the best validation performance achieved at epoch 23 with a value of $MSE = 0.010873$. The error gradient at the end of training is 0.00003023, which indicates a good convergence towards a local minimum. The scaling parameter of the Levenberg-Marquardt algorithm (μ) reached the value of 1×10^{-15} in the last epoch, which confirms that the algorithm has entered the phase of rapid Newton descent characteristic of the vicinity of the minimum.

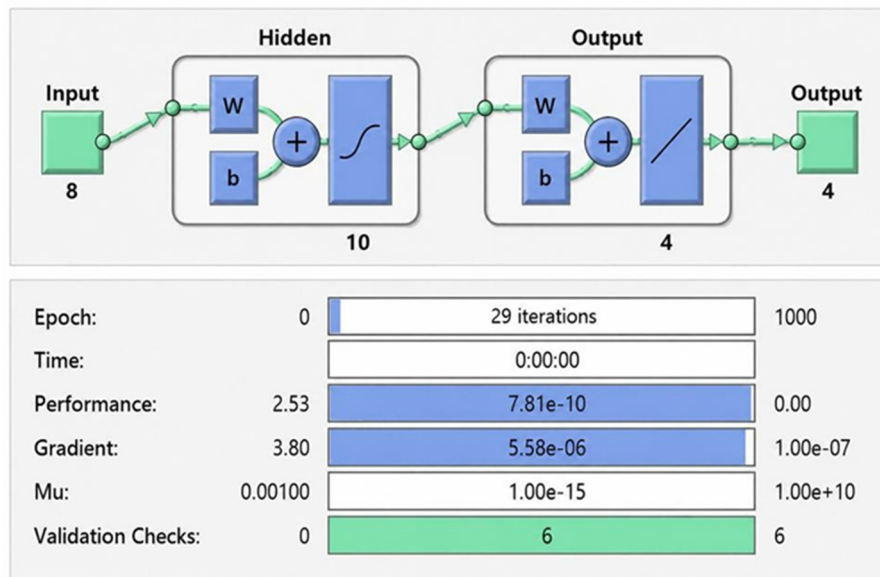


Figure 9: Neural network architecture and training parameters

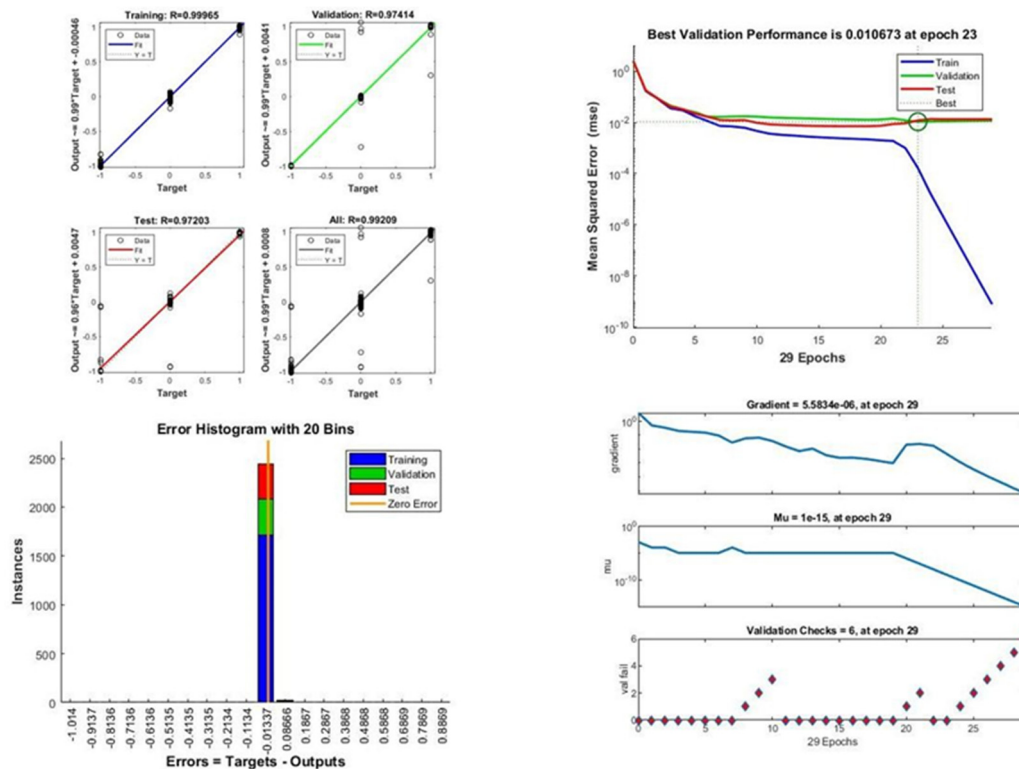


Figure 10: Neural network training in MATLAB

High correlation coefficient values ($R > 0.97$) on all data sets confirm that the network successfully learns the mapping between sensor data and control commands and has a satisfactory ability to generalize to new, previously unseen samples. A small difference between the training and validation coefficients indicates a slight overfitting, which is acceptable given the size of the data set (hundreds of samples for each of 9 commands). The error histogram confirms the concentration of errors around zero with a distribution approaching normality, which is a desirable property of a well-trained regression model.

3.3. Simulation of the Drone System with Intelligent Control

After the integration of the trained neural network into the Simulink model of the drone, a series of simulations lasting 20 seconds were carried out. The simulation environment is composed of mathematical model of drone dynamics that accounts for physical characteristics of the system, aerodynamic forces and moments, the gravitational field and regulatory loops for stabilizing position and orientation, and neural network that generates the input commands - roll, pitch, yaw, and throttle, which are applied directly to the drone model, simulating a realistic gesture control scenario without a physical remote control.

To start the simulation, the glove is physically connected to the computer via a serial connection (COM4, 9600 baud), after which a script is run in the MATLAB environment that establishes serial communication, loads the previously trained neural network model and initializes the simulation in the Simulink environment (Figure 11). From that moment, the user controls the drone in real time with hand movements and finger flexions, and the system in each iteration receives sensor data, processes it through a neural network and forwards the generated commands directly to the drone model.

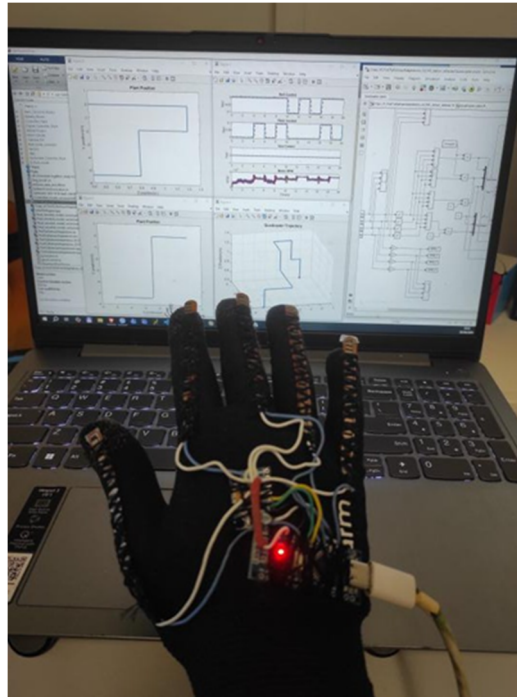


Figure 11: Simulation environment

The simulation results confirm that the proposed system enables stable and precise control of the drone exclusively through hand gestures (Figure 12). The time diagrams of the angular quantities show that the system successfully follows the given reference signals: the transitions between values are fast, and the tracking error in the stationary phases is negligibly small. The control signals have discrete changes corresponding to the detected commands of the neural network, while the system response indicates that the command recognition and transformation algorithm does not introduce significant delay into the control.

In order to quantitatively validate the simulation results, key performance metrics were extracted from the recorded signals. The steady-state tracking error for the roll and pitch angles was measured to be below 0.003 rad, while the yaw angle maintained a steady-state error of less than 0.005 rad throughout the entire simulation. The rise time for step transitions in the roll and pitch reference signals averaged approximately 0.4 seconds, with a settling time of under 0.8 seconds, confirming rapid and stable response to gesture-generated commands. Overshoot across all angles remained below 5%. The total distance traveled by the drone during the simulation was approximately 10.5 meters across all three axes, with positional transitions executed without oscillation or drift between commanded segments. These quantitative results collectively confirm the stability, accuracy, and efficiency of the proposed gesture-based control system under simulated flight conditions.

The three-dimensional representation of the drone's trajectory gives the most complete picture of the spatial movement during the simulation. The characteristic stepped form of the trajectory clearly describes the individual phases of the flight: horizontal movements in different directions alternate with changes in height, forming a three-dimensional trajectory that directly reflects the sequence of gestures performed during the simulation. Vertical jumps between heights are performed precisely and with well-defined levels, which confirms the efficient management of the throttle channel. The projection of the trajectory in the X–Y plane shows smooth and stable segments of movement without sudden jumps or instability, which testifies to the good quality of regulation and the absence of oscillations in the control loop.

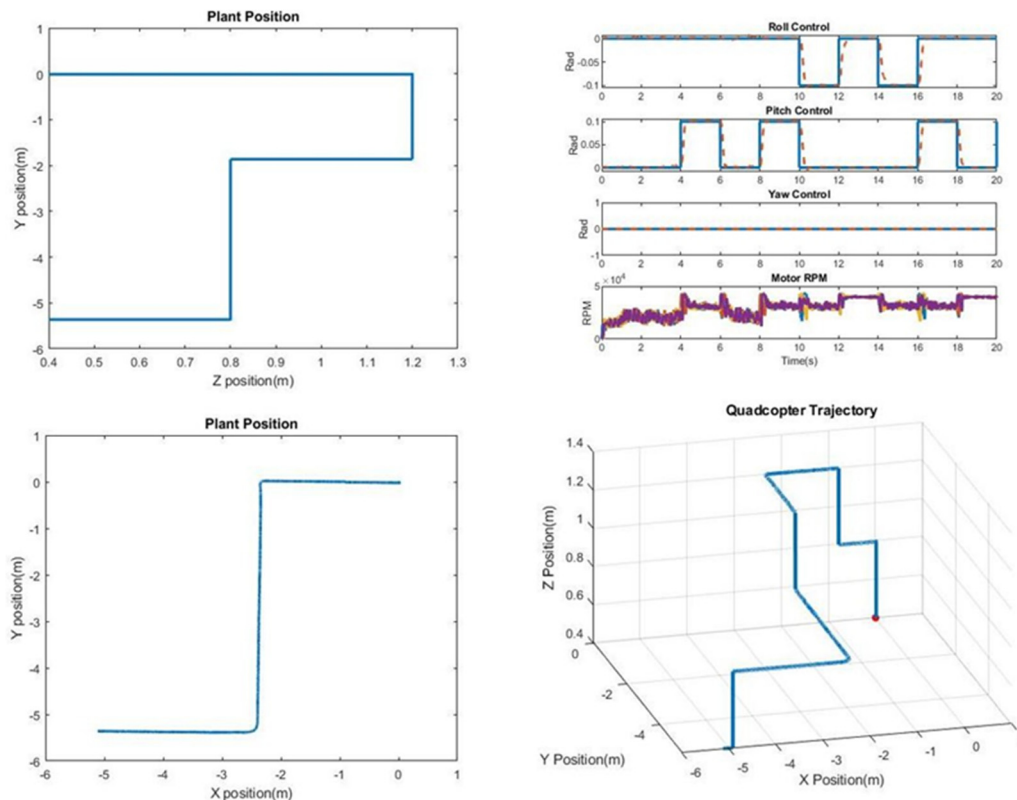


Figure 12: Simulation results when using smart glove as input for drone commands

4. CONCLUSION

In this paper, a complete system for modeling, simulation and intelligent control of a drone has been implemented. A detailed mathematical model of the drone dynamics has been built, which includes aerodynamic forces, engine and battery dynamics, as well as sensor systems and control algorithms. The mathematical model of the drone is implemented in the MATLAB/Simulink environment, which allows for analysis of the aircraft's behavior in controlled simulation conditions. Special attention is paid to the processing of sensor data, where a Kalman filter is applied in order to reduce noise and obtain a more accurate assessment of the system's state, and flight stabilization is achieved through a PID control system. The core idea of the project is the method of interaction between the user and the drone - instead of a classic remote control, control is realized through a smart glove equipped with sensors. The data from the glove is processed by a neural network that recognizes the user's gestures and transforms them into control commands that are used as input to the drone's simulation model.

The results of the conducted simulations show that the proposed approach works stably and predictably. The system successfully tracks the reference values for pitch, yaw, and rotation, with low tracking error and fast and stable transitions between different states. The control signals generated by the neural network directly affect the dynamics of the motors and result in expected changes in the drone's movement. The analysis of the three-dimensional flight trajectory further confirms that the system successfully interprets the user's gestures and converts them into realistic maneuvers in space. The drone's movement is continuous and stable, without sudden or uncontrolled changes, indicating that the regulatory loops and the gesture recognition algorithm work in sync and efficiently.

The practical significance of such a system is broad and has the potential to be applied in systems where intuitive and natural interaction between man and machine is required. Gesture control can be used in rescue missions, inspection operations, education, as well as in research and demonstration systems where it is important that the control is simple and easy to understand for the user. Such an approach represents a step towards more intuitive interfaces for controlling unmanned aerial vehicles and other autonomous systems. Today, research in this area is moving towards fully wireless, embedded systems with negligible delay and high accuracy.

Although the simulation results are encouraging, there are several possibilities for further improving the system. In the current implementation, control commands are generated as discrete values, which means that each gesture corresponds to a fixed command. In the future, a system could be developed that recognizes continuous movements, i.e. not only the direction of movement but also a slight tilt to the left to generate a small tilt value, and a pronounced tilt — a larger value, which would provide the user with proportional control. Further improvement can be achieved by using more precise sensors, a larger number of sensor points, or by introducing intermediate steps and combined gestures, which would allow for a larger number of control commands. In parallel, replacing serial with wireless communication is a technically trivial step that eliminates the only physical limitation of the prototype, and testing on a real drone remains a logical and inevi-

table next step for final validation of the system outside the simulation environment. Modern research in this area often uses deeper neural networks, advanced sensor data fusion methods, and adaptive control algorithms, which could further increase the precision and stability

REFERENCES

- [1] M. Idrissi, M. Salami, and F. Annaz, “A Review of Quadrotor Unmanned Aerial Vehicles: Applications, Architectural Design and Control Algorithms”, *Journal of Intelligent & Robotic Systems*, Vol. 101(22), <https://doi.org/10.1007/s10846-021-01527-7>, (2021)
- [2] I. H. Beloev, “A Review on Current and Emerging Application Possibilities for Unmanned Aerial Vehicles”, *Acta Technologica Agriculturae*, Vol. 19(3), pp. 70-76, <https://doi.org/10.1515/ata-2016-0015>, (2016)
- [3] N. Koksai, M. Jalalmaab, and B. Fidan, “Adaptive Linear Quadratic Attitude Tracking Control of a Quadrotor UAV Based on IMU Sensor Data Fusion”, *Sensors* 2019, Vol. 19(1), pp. 46, <https://doi.org/10.3390/s19010046>, (2019)
- [4] S. Bouabdallah, A. Noth, and R. Siegwart, “PID vs. LQ control techniques applied to an indoor micro quadrotor”, *Proceedings of the 2004 IEEE International Conference on Robotics and Systems (IROS)*, Sendai (Japan), 28 September - 02 October 2004, (2004)
- [5] B. Jiang, B. Li, W. Zhou, L.-Y. Lo, C.-K. Chen, and C.-Y. Wen, “Neural Network Based Model Predictive Control for a Quadrotor UAV”, *Aerospace*, Vol. 9(8), p. 460, <https://doi.org/10.3390/aerospace9080460>, (2022)
- [6] T. Wang, S. S. Zhang, and N. M. Freris, “An Overview of Quadrotor UAV Control Methodologies”, <https://doi.org/10.48550/arXiv.2209.02954>, (2022)
- [7] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. de la Puente, and P. Campoy, “A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Moving Platform”, *Journal of Intelligent & Robotic Systems*, Vol. 93(1), pp. 351-368, <https://doi.org/10.1007/s10846-018-0891-8>, (2019)
- [8] A. S. Brandao, F. N. Martins, and H. B. Soneguetti, “A vision-based line following strategy for an autonomous UAV”, *Proceedings of the 2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, Colmar (France), 21-23 July 2015, pp. 214-219, (2015)
- [9] S. L. Waslander, G. M. Hoffmann, J. S. Jang, and C. J. Tomlin, “Multi-Agent Quadrotor Testbed Control Design: Integral Sliding Mode Control Design”, <https://doi.org/10.48550/arXiv.1905.01657>, (2019)
- [10] J. Cifuentes, P. Boulanger, M. T. Pham, F. Prieto, and R. Moreau, “Gesture Classification Using LSTM Recurrent Neural Networks”, *Proceedings of the 2019 41st Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, Berlin (Germany), 23 July 2019, pp. 6864-6867, (2019)
- [11] J.-W. Lee and K.-H. Yu, “Wearable Drone Controller: Machine Learning-Based Hand Gesture Recognition and Vibrotactile Feedback”, *Sensors*, Vol. 23(5), p. 2666, <https://doi.org/10.3390/s23052666>, (2023)
- [12] C. Yu, S. Fan, Y. Liu, and Y. Shu, “End-Side Gesture Recognition Method for UAV Control”, *IEEE Sensors Journal*, (2022)
- [13] K. Haratiannejadi and R. R. Selmic, “Smart Glove and Hand Gesture-Based Control Interface for Multi-Rotor Aerial Vehicles in a Multi-Subject Environment”, *IEEE Access*, Vol. 8, pp. 227667-227677, <https://api.semanticscholar.org/CorpusID:230512516>, (2020)
- [14] W.-C. Chuang, W.-J. Hwang, T.-M. Tai, D.-R. Huang, and Y.-J. Jhang, “Continuous Finger Gesture Recognition Based on Flex Sensors”, *Sensors*, Vol. 19(18), p. 3986, <https://doi.org/10.3390/s19183986>, (2019)
- [15] G. Zhu, L. Zhang, P. Shen, and J. Song, “Multimodal Gesture Recognition Using 3-D Convolution and Convolutional LSTM”, *IEEE Access*, Vol. 5, pp. 4517 - 4524, <https://doi.org/10.1109/ACCESS.2017.2684186>, (2017)
- [16] H. Cheng, L. Yang, and Z. Liu, “Survey on 3D Hand Gesture Recognition”, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 26(9), pp. 1659-1673, <https://doi.org/10.1109/TCSVT.2015.2469551>, (2016)
- [17] A. Sulficar, H. Suresh, A. Varma, and A. Radhakrishnan, “Modeling, Simulation and Complete Control of a Quad-copter”, *Major Project Semester Report*, National Institute of Technology Karnataka (India), (2017)
- [18] R. Mahony, V. Kumar, and P. Corke, “Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor”, *IEEE Robotics & Automation Magazine*, Vol.19(3), pp. 20-32, <https://doi.org/10.1109/MRA.2012.2206474>, (2012)
- [19] A. Sreejan and Y. S. Narayan, “A Review on Applications of Flex Sensors”, *International Journal of Emerging Technology and Advanced Engineering*, Vol. 7(7), pp. 97-101, (2017)
- [20] J. Huang, “Design of Angle Detection System Based on MPU6050”, *Proceedings of the 7th International Conference on Education, Management, Information and Computer Science (ICEMC 2017)*, pp. 6-8, <https://doi.org/10.2991/icemc-17.2017.2>, (2017)
- [21] H. P. Gupta, H. S. Chudgar, S. Mukherjee, T. Dutta, and K. Sharma, “A Continuous Hand Gestures Recognition Technique for Human-Machine Interaction Using Accelerometer and Gyroscope Sensors”, *IEEE Sensors*, Vol. 16(16), pp. 6425-6432, <https://doi.org/10.1109/JSEN.2016.2581023>, (2016))

- [22] C. Liu and T. Szirányi, “Real-Time Human Detection and Gesture Recognition for On-Board UAV Rescue”, *Sensors*, Vol. 21(6), p. 2180, <https://doi.org/10.3390/s21062180>, (2021)
- [23] B. Hu and J. Wang, “Deep Learning Based Hand Gesture Recognition and UAV Flight Controls”, *International Journal of Automation and Computing*, Vol. 17(1), pp. 17-29, <https://doi.org/10.1007/s11633-019-1194-7>, (2020)
- [24] N. Ferry, “Quadcopter Plant Model and Control System Development With MATLAB/Simulink Implementation”, Master's Thesis, Rochester Institute of Technology (USA), (2017)
- [25] B. Taylor, M. Allen, P. Henson, X. Gao, H. Malik, and P. Zhu, “Enhancing Drone Navigation and Control: Gesture-Based Piloting, Obstacle Avoidance, and 3D Trajectory Mapping”, *Applied Sciences*, Vol. 15(13), p. 7340, <https://doi.org/10.3390/app15137340>, (2025)
- [26] J. J. Moré, “The Levenberg-Marquardt algorithm: Implementation and theory”, in *Numerical Analysis*, G. A. Watson, Ed., Berlin, Germany: Springer, 1978, pp. 105-116., <https://doi.org/10.1007/BFb0067700>, (2006)
- [27] K. J. Åström and T. Hägglund, “PID Controllers: Theory, Design, and Tuning”, Instrument Society of America, Research Triangle Park, North Carolina (USA), (1995)
- [28] S. J. Julier and J. K. Uhlmann, “New extension of the Kalman filter to nonlinear systems”, *Proceedings of SPIE*, Vol. 3068, Signal Processing, Sensor Fusion, and Target Recognition VI, pp. 182-193, <https://doi.org/10.1117/12.280797>, (1997)
- [29] M. S. Grewal and A. P. Andrews, “Kalman Filtering: Theory and Practice Using MATLAB”, Wiley, New York, New York (USA), (2002)
- [30] F. Auger, M. Hilaret, J. M. Guerrero, E. Monmasson, T. Orłowska-Kowalska, and S. Katsura, “Industrial Applications of the Kalman Filter: A Review”, *IEEE Transactions on Industrial Electronics*, Vol. 60(12), pp. 5458-5471, <https://doi.org/10.1109/tie.2012.2236994>, (2013)