UDC 004.852 **doi: 10.32620/reks.2025.1.05**

Oleksandr BEZSONOV[1], Serhii LIASHENKO[2], Oleg RUDENKO[1], Sofiia RUTSKA[1], Kateryna VASHCHENKO[1]

[1] ***Kharkiv National University of Radio Electronics, Kharkiv, Ukraine***
[2] ***State Biotechnological University, Kharkiv, Ukraine***

USING A DEEP LEARNING NEURAL NETWORK TO PREDICT FLIGHT PATH

The subject *of this paper is a new approach using a deep learning neural network designed for predicting the flight path of an unmanned aerial vehicle (UAV).* ***The purpose*** *of this study was to improve the accuracy of drone flight path prediction by developing a deep learning-based trajectory forecasting model.* ***The task*** *was to collect and prepare a dataset of video and photo materials for training the neural network, develop and implement a deep learning model for trajectory prediction, and enhance UAV flight trajectory forecasting through model optimization and validation.* ***Methods*** *used included the creation of a synthetic dataset using the 3D modeling tool Blender, which enabled the generation of animations representing various drone flight scenarios. These scenarios include different environmental conditions and urban landscapes, providing a robust training ground for the neural network. To further improve and test the model's predictive capabilities, real-world data, including eyewitness videos, were used. The architecture of the neural network includes long short-term memory (LSTM) units that can process sequential data, making them ideal for predicting dynamic UAV trajectories. The training process involved several stages, starting with pre-training on general visual features and then fine-tuning to UAV-specific motion patterns.* ***The results*** *of this study show that the neural network achieved high accuracy in trajectory prediction, with the model showing better performance in real-world scenarios compared to traditional trajectory prediction methods. The integration of LSTM enabled efficient learning and generalization of temporal data, capturing complex motion patterns and interactions with the environment. This research not only demonstrates the feasibility of using deep learning to predict UAV trajectories but also offers potential applications in civilian security or delivery logistics, where real-time trajectory prediction can significantly improve the efficiency and speed of decision-making.* ***Conclusions****. The scientific novelty of the obtained results lies in the development and training of deep learning models specifically designed for predicting drone flight paths. This study demonstrated the effectiveness of the proposed approach by demonstrating its ability to enhance the accuracy of UAV trajectory forecasting*.

Keywords: *neural network; deep learning; trajectory prediction; LSTM; 3D models; synthetic dataset; UAV trajectory.*

1. Introduction

1.1. Motivation

In today's world, where technology is developing at an extraordinary pace, its use to protect information and improve the quality of life is becoming increasingly important. Improving cargo delivery logistics, preserving the environment, and surveillance to improve safety emphasize the urgent need to develop and implement innovative technological solutions to develop technological capabilities. One such solution is the development of new efficient methods to detect and predict the trajectories of drones, which are used for various purposes, such as entertainment during leisure time, photography of unusually beautiful landscapes, and in developed countries to catch intruders faster. Most methods for predicting the behavior of dynamical systems are based on the use of information about the parameters of mathematical models and environmental properties. However, the lack of such a priori information leads to inefficiency in the use of traditional parametric methods, and often to their inoperability. In addition, the presence of nonstationary, nonlinearity, and various types of uncertainties in the models of real systems leads to inefficiency when using traditional nonparametric methods. Under these conditions, the most effective approach is the use of modern methods of computational intelligence, in particular neural network methods. The neural network-based approach to drone detection remains relevant. It provides an opportunity to prevent non-ideal human factors and use modern technologies for noble purposes. Intelligent recognition by neural network of such devices as drones can enrich the delivery of products and food, for example, in a non-excepted period as a pandemic, to perform unpleasant routine activities such as refueling a car autonomous process without the

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

presence of a person but using a drone. It can also help protect the integrity of citizens and prevent the spread of illegal activities using drones, such as the delivery of dangerous substances for human health and life and other non-standard and unexpected situations. Using a neural network to plot the trajectory of a drone can improve public protection and safety throughout the country. Protect against illegal invasion of privacy with drones that can take photographs and videos. Intelligent advancements in technology can and should help to keep everyone's intimate lives safe.

The purpose of this study was to improve the accuracy of drone flight path prediction by developing a deep learning-based trajectory forecasting model. Two datasets were used to obtain the expected results. The first part is an artificially created dataset, including animations including models of drones, and different types of backgrounds and residential complexes to diversify the dataset. These animations were created using the 3D modeling program Blender. This approach allows us to train the model in conditions as close to reality as possible without the need to use real video footage, which can be time-consuming and expensive and may be legally restricted in some sites. The second type includes real videos filmed by ordinary people and presented on the Internet and news channels on which the trained neural network was tested.

The task was to collect and preprocess video and photo data, develop a deep learning model for flight trajectory prediction and improve UAV trajectory prediction through optimization and validation.

1.2. State of the art

Currently, there are different methods for trajectory prediction depending on the type of trajectory, such as trajectories of living beings [1, 2] and inanimate objects [3]. There are also two main problems with the currently existing trajectory prediction models. The first is the prediction accuracy: simple models do not work well if there are many objects around or a complex scene. The second is the lack of explanation: neural networks predict a change in trajectory, but it is not known why it was changed. The authors of [1] presented a new method for predicting motion trajectory in crowds. The authors have created a model that combines physics and neural networks. Each person, according to their method, is influenced by three main forces: moving toward a destination, trying not to run into other people, and avoiding obstacles such as walls or cars. The model also includes a part of the conditional-variance autoencoder (CVAE), which adds a randomness element. The model learns from the data when people speed up, slow down, or find themselves in unpleasant situations where they go in the wrong direction and have to change their destination. In such cases, the model adds to the predictions. This makes the movements appear more realistic and natural, approximating the behavior of people. The authors tested the proposed method on real trajectories and compared its performance to that of existing deep learning models. Their model was better at predicting movements, performed better in complicated scenes, and had fewer collisions. Detecting human motion, which can often be relatively unpredictable, shows that combining physics and neural networks gives better results for humans and potentially for UAVs. The Social-NCE method allows the model to be trained to recognize safe trajectories from potentially dangerous trajectories that could cause collisions by generating negative samples based on the closeness of other objects [2]. Recently, model-free methods based on deep learning that exhibit surprisingly high prediction accuracy have been leading the way in predicting human motion trajectories.

The trajectory of inanimate objects usually obeys the laws of physics and can often be derived using formulas and equations; thus, detections are sometimes easier to predict even manually. As an example, in a previous study [3], the authors proposed a model that combines an improved LSTM, which predicts the next point of the trajectory based on the movement history and captures the dependencies between the time steps. Using the Kalman filter, the authors demonstrated that the LSTM-KF algorithm yields a good effect. However, this approach does not consider dynamic and environmental factors such as weather conditions, which reduces the accuracy, and it limits the application in real time.

These methods are effective when applied to the environment in which they were developed. However, these methods are not implemented in aerial environments, and there is not enough data to analyze and make conclusions using a specific method. In UAV tasks, drones work in 3D space, where altitude, speed, and viewing angles are important, and simply following a trajectory is not enough for such cases.

Another recent vehicle trajectory prediction approach focuses on modeling potential future interactions between objects. Social LSTM applies Long Short-Term Memory (LSTM) networks to model social interactions and learn from temporal data. LSTM is also commonly used to predict the motion trajectories of inanimate objects. Other studies have investigated vehicle motion and observed that prediction uncertainty arises from interactions with surrounding objects when vehicles change their route. Based on this, the authors [4] proposed a model that incorporates a Future Relation Module (FRM), which estimates the probability of vehicles occupying adjacent lanes and interacting. The module computes lane-level probability distributions and potential interaction zones by leveraging Graph

Convolutional Networks (GCN) and Gaussian Mixture (GM) distributions to simulate interactions, such as handling current road conditions, and demonstrate various behaviors. This approach allows the model to capture long-range interactions and improves prediction accuracy in complex traffic scenarios. The proposed method demonstrated strong performance on the nuScenes and Argoverse benchmarks. The study investigated vehicular motion; thus, predictions in this paper stem from uncertainty. Trajectories can change due to interactions with surrounding objects.

1.3. Objectives and Approach

This section describes the design and implementation of a deep learning model for flight trajectory prediction using deep learning RCNNs and predictive LSTMs. The basic concept behind these steps can be described as follows:

- First, the research was divided into working with RCNNs recognition neural network, creating a dataset for it, searching for rare drone images under different conditions, and using software to create an artificial dataset to expand and diversify the amount of data on which the neural network is trained. In this way, the artificial intelligence will be better prepared for unexpected data and can easily handle the processing of such data.
- The LSTM neural network was used to design the estimated drone trajectory. For this purpose, materials were found, including videos of the desired object flying at test bases. To add variety to the data, plausible videos were created in the software to train the neural network to better predict the drone's flight path. The training can be categorized as follows:
- Creating a set of videos and photos for training and validation;
- Augmenting and annotating the datasets;
- Developing a neural network model capable of recognizing drone;
- Developing an LSTM model to predict drone flight trajectories;
- Using a set of videos and photos for training and validation to recognize and create trajectories;

In this study, a synthetic dataset was created to represent diverse drone flight scenarios under varying environmental conditions and landscapes. Existing trajectory prediction methods were analyzed, and a deep learning-based approach was designed and implemented.

To achieve the objectives of the study, several tasks were completed, as reflected in the corresponding sections of the article. The creation of the dataset for training the neural network is described, where both real recordings and artificially generated animations in Blender were used to ensure data diversity (section 2). Values of high resolution cameras to get an accurate image (section 2). Value of high resolution cameras to get an accurate image (section 2.1). Data segmentation for more precise object recognition and classification is discussed (section 2.2). Selection of RCNN neural network architecture (section 3). The process of training the network is outlined (section 4), then the results are presented (section 5). Section 6 provides a detailed explanation of the proposed method for predicting flight trajectories using the LSTM neural network. The possibilities for extending the model, including the use of Google Earth data to analyze object trajectories (section 7).

2. Methodology

The training of the neural network was a multistage process that used both artificially generated animations and real-world footage to provide the model with a diverse dataset. The artificial animations were created using Blender, a sophisticated 3D modeling and animation software. This tool afforded us the precision to control environmental variables such as lighting conditions, background, and flight trajectories of drone models with pinpoint accuracy. The meticulous control of these variables is crucial in training neural networks because it ensures that the model can identify and learn from a wide variety of scenarios in a controlled and repeatable manner.

For example, using Blender, we could simulate different weather conditions, ranging from bright sunlight to overcast skies, or foggy mornings to rainy evenings. The textures and colors of the background environments can be altered to represent various landscapes, such as urban settings, rugged terrains, and open fields. This level of detail in the animations means that the neural network can be exposed to almost every conceivable situation that a drone might encounter in the real world [5, 6]. A drone model was used, examples of which are shown in Figure 1.

For example, scaling transformations can be mathematically represented using the following formula, where S is the scaling matrix applied to the image coordinates (x,y):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = S * \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix}, \quad (1)$$

where s_x and s_y – there scaling factors in the horizontal and vertical directions, respectively;

(x', y') – the new coordinates after scaling.

The rotation transformation involves rotating the image by an angle, which is described by the rotation matrix R as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = R(\theta) * \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix}, \quad (2)$$

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Fig. 1. 3D models of drone

Adjustments in lighting conditions are simulated by altering the intensity values of the pixels, which can be represented by a function L that maps the original pixel values p to new values p' based on the lighting parameter l:

$$p' = L(p, l). \quad (3)$$

Additionally, Blender particle systems can be used to simulate environmental factors such as rain, snow, or dust, that can interfere with the drone's visibility. The motion of these particles can be described by Newton's laws of motion as follows:

$$F = ma, \quad (4)$$
$$v_{t+1} = v_t + a * \Delta t, \quad (5)$$
$$x_{t+1} = x_t + v_t * \Delta t, \quad (6)$$

where F – the force applied to a particle;
m – the mass of the particle;
a – the acceleration;
v – the velocity;
x – the position;
Δt – the change in time.

In terms of lighting, Blender allows for dynamic lighting adjustments, which can significantly affect the appearance of the drone in the animation. By changing the intensity, color, and position of light sources, different times of day, weather conditions, and shadowing effects can be replicated [7]. These lighting conditions can be described by parameters such as the light's intensity I, the angle of incidence θ, and the distance from the light source d, which affect the illumination of the object according to the inverse square law:

$$E = \frac{I}{d^2}\cos(\theta), \quad (7)$$

where E is the illumination experienced by the object.

The lighting and environmental effects, such as shadows and reflections on the drone's body, were also meticulously adjusted to simulate different times of day and weather conditions. The rendering equation, which Blender uses to simulate light interaction with surfaces, is as follows (8):

$$L_o(x,\omega,\lambda,t) = L_e(x,\omega,\lambda,t) + \int_\Omega \begin{matrix} f_r(x,\omega',\omega,\lambda,t)L_i * \\ (x,\omega',\lambda,t)(-\omega'*n)d\omega' \end{matrix}, \quad (8)$$

where λ – wave length of light, t – time;

$L_o(x,\omega,\lambda,t)$ – the radiance of the outgoing light of wavelength λ at point x and time t, in direction ω;

$L_e(x,\omega,\lambda,t)$ – the radiance of the emitted light, $\int_\Omega \ldots d\omega'$ – integral over the hemisphere of incoming directions;

$f_r(x,\omega',\omega,\lambda,t)$ – bidirectional reflectance distribution function (BRDF), defining the proportion of light of wavelength λ at time t from direction ω′ that is reflected into direction ω at point x;

$L_i(x,\omega',\lambda,t)$ – the radiance of the incoming light of wavelength λ from direction ω′ at time t, $-\omega' * n$ – absorption of the incoming value by the set angle.

Blender's physics engine also allows for the simulation of collisions and interactions between the drone and other objects, which can be modeled using the principles of rigid body dynamics. These interactions can alter a drone's flight path in ways that are important for a neural network to learn if it is to understand the full scope of potential real-world behaviors.

These transformations significantly bolster the model's capacity to recognize objects under various perspectives and lighting conditions, which closely simulates the variability that occurs in natural settings. By incorporating a range of augmented data, the neural network can be exposed to a broader spectrum of experiences, thereby enhancing its ability to generalize from training data to real-world applications.

Incorporating these techniques into the training pipeline involves randomly applying a series of transformations to each image in the dataset before it is fed into the neural network. This ensures that the network rarely sees the same image twice, which mitigates overfitting and encourages the development of a more general understanding of the features associated with the objects of interest. Ultimately, data augmentation contributes to the creation of a more versatile and adaptable model that can deliver reliable performance across several environmental conditions and variations in object appearance. This enriched learning process is fundamental for developing advanced neural networks that are expected to function effectively in dynamic and unpredictable real-world scenarios [8].

Blender created many synthetic scenarios with precise control over every aspect of the animation, allowing for the simulation of real-life complexities within a virtual environment. In pursuit of advancing neural network capabilities for drone recognition and trajectory prediction, a detailed animation of the drone was created. Highly realistic and diverse flight patterns served as training grounds for the neural network. Within Blender, the drone's motion is animated using keyframe interpolation, which defines the start and end points of a motion sequence, with Blender computing the intermediate frames. For complex maneuvers, the animation curve can be mathematically described using a series of control points that form a spline, typically a Bezier spline in 3D space (9).

$$B(t)=\sum_{i=0}^{n}\binom{n}{i}(1-t)^{n-i}t^{i}P_i\,,\ t\in[0,1], \tag{9}$$

where B(t) is the position of the drone on the Bezier curve at time t;

P_i – the control points;

$\binom{n}{i}$ – the binomial coefficients.

This formula ensures smooth transitions between keyframes and realistic motion paths.

For instance, consider the creation of an animation sequence in Blender in which a drone model maneuvers through a virtual urban landscape. The drone's trajectory is not just a simple linear path but includes various maneuvers, such as ascent, descent, and sharp turns, to mimic real-world flight patterns. This can be expressed mathematically in the animation keyframes as a Bezier curve, which is defined by control points $P_0, P_1, P_2, \ldots P_n$ to formula (2) where, B(t) represents the Bezier curve, $\binom{n}{i}$ are the binomial coefficients, and t is the parameter along the curve.

The real-life footage, on the other hand, included authentic video recordings of drones captured under several operational conditions. This footage was instrumental in enhancing the robustness of the model. By including real-life data, the neural network was trained on not only the idealized conditions presented in animations but also the unpredictability and variance found in real-life scenarios. This included factors such as unpredictable drone behavior, varying speeds, abrupt changes in direction, and the presence of obstacles like birds, aircraft, and man-made structures.

The combination of these two data sources created a comprehensive learning environment for the neural network. The model was thus not only trained to recognize and predict drone behavior in a theoretical sense but was also well-equipped to handle the complexities and nuances of real-world operations. The goal was to create a neural network that could function with high reliability and accuracy, regardless of the operating environment.

2.1. High-resolution Technology

The successful implementation and application of neural networks in trajectory prediction are highly contingent upon the use of advanced high-resolution cameras with superior zoom capabilities. Such sophisticated imaging technology is imperative for capturing the intricate details and nuances necessary for the neural network to accurately learn and predict object trajectories. High-resolution cameras are capable of capturing images and videos with several pixels, which directly translate to a higher level of detail in each frame. This granularity is not just a matter of visual quality; it provides the neural network with the subtle visual cues required to differentiate between objects in complex scenes and to recognize patterns with greater precision. The importance of high-resolution input data can be likened to providing a painter with a finer brush, thus providing the painter with the opportunity to create a more nuanced and detailed work of art. Furthermore, cameras with powerful zoom capabilities extend the range and versatility of the neural network's predictive ability. By bringing distant objects into clear view, these cameras allow the network to effectively learn and predict the movements of objects that are far away from the lens. This is particularly crucial when subjects of interest are often at a significant distance from the camera. The deployment of cutting-edge cameras should be accompanied by equally advanced processing hardware capable of handling the large volumes of data generated. High-resolution imagery requires significant storage capacity and powerful computational resources to process detailed images in real-time, especially when feeding these data into a neural network for immediate trajectory prediction [9]. Additionally, cameras must be equipped with high dynamic range (HDR) capabilities to handle a wide spectrum of lighting conditions, from

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

darkest shadows to brightest highlights, ensuring consistent performance regardless of environmental lighting challenges to fully realize its potential across various conditions. This approach empowers the neural network to operate at its highest capability, increasing accuracy and reliability in trajectory prediction camera examples (Figure 2).

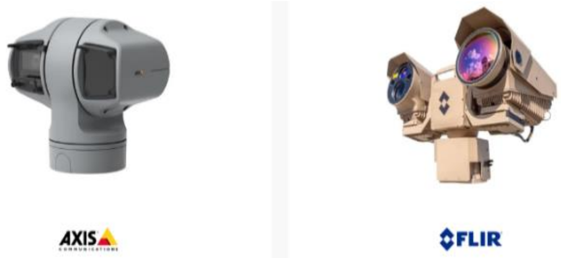

Fig. 2. Examples of high-resolution camera

2.2. UAV dataset

Data segmentation is the process of dividing an image into smaller parts or segments for more detailed analysis. The primary purpose of segmentation in computer vision is to extract and classify objects in an image. It is an important step in various tasks, such as object detection, pattern recognition, and automatic image processing and analysis.

To illustrate data segmentation, an example using the drone model is shown in Figure 3.

To begin with, it is worth noting that the process of training a model for trajectory prediction is a rather variable topic and can be interpreted differently depending on the requirements and settings of the model for real-world applications. Models can be adapted to recognize objects that inherently move differently than other objects; thus, each case should be investigated separately. As a result, a test system was developed for trajectory finding, which involves recognizing an object in a video and calculating the possible direction of the target.

To successfully investigate a high-speed flying object, we faced limitations in the available materials. In this situation, computer graphics, especially 3D modeling, was a useful tool to overcome data scarcity problems. After collecting information about the objects and creating virtual models for different scenarios, we developed and trained a machine learning model capable of recognizing the objects we needed among the data obtained from the simulation [10].

This approach allowed us to work efficiently with limited resources and produce meaningful results despite the constraints of physical access to the object and limited amount of data. As a result, we developed a tool that can automatically identify and analyze objects of interest in environments where direct observation is limited or impossible.

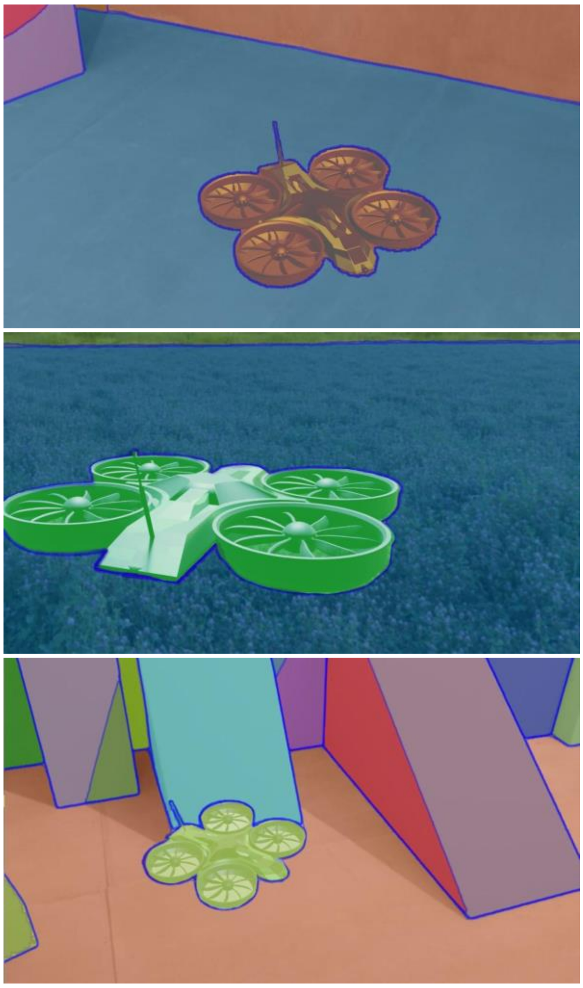

Fig. 3. Segmentation results

3. Neural network architectures

For this study, an architecture was selected that could ensure the detection of objects in video footage, thereby necessitating high-precision recognition capabilities. Among the various options considered, particular attention was given to the convolutional neural network architecture known as RCNN (Region-based Convolutional Neural Networks). This architecture has distinct advantages and disadvantages. It was selected for its ability to precisely delineate and classify objects in images and videos. RCNNs are effective in detecting visual patterns and structures in images due to their capacity for local perception and hierarchical information processing. They are perfectly suited to processing static images and video content.

RCNNs are applied to video by processing each frame as a separate image because they do not consider the temporal dependencies between successive frames. As the research did not require the speed of the objects to be taken into account, the video could be dis-segmented into individual frames. Ultimately, a hybrid model was chosen to achieve the objectives, combining convolutional layers for effective perception of visual patterns within frames and layers based on attention mechanisms for processing temporal dependencies within the frame sequence. This approach considers both the spatial and temporal aspects of the video stream, ensuring high object detection accuracy and trajectory prediction [11].

The hybrid model's design strategically leverages the strengths of RCNNs while compensating for their limitations. By integrating attention mechanisms, the model gains the ability to track objects across frames, which is critical for capturing motion and predicting future locations. This dual focus on both the immediate visual details and the broader temporal context creates a robust framework for understanding and interpreting dynamic scenes. The hybrid model's versatility makes it a powerful tool for a wide range of applications, from automated surveillance systems to advanced driver-assistance systems (ADAS) in vehicles, where the precise tracking of objects around the vehicle can be vital for safety and navigation.

In summary, this sophisticated video analysis approach harnesses the latest advances in neural network technology, thereby setting a new standard for accurate and efficient object detection and trajectory forecasting. The potential of this technology to revolutionize how we interact with and analyze visual data is immense, opening up new possibilities for innovation across numerous industries and fields of research.

LSTM stands for Long Short-Term Memory. A recurrent neural network (RNN) architecture is well-suited for processing and predicting sequences of data.

Figure 4 illustrates the internal structure of LSTM cell. The main function of LSTM is to capture long-range dependencies by controlling the flow of information through three gates: the forget gate, input gate, and output gate. These gates are regulated by sigmoid σ and "tanh" functions. The forget gate determines which part of the previous state h_{t-1} to discard, the input gate decides what new information X_t should be added to the current cell state, and the output gate regulates which part of the cell state should contribute to the output h_t. The cell state is updated and passed to the next time step, which allows the model to retain information over long sequences. The flow of operations, shown through multiplications, additions, and non-linear activations, demonstrates that the proposed LSTM effectively maintains long-term memory while selectively updating it with new information.

The LSTM updates its internal state by considering both the current input (the spatial coordinates of the drone at the current time step) and the information retained from previous time steps. This allows the LSTM to capture temporal dependencies and patterns in trajectory data. After processing the input, the LSTM unit generates an output, which represents the predicted spatial coordinates of the drone at the next time step. This predictive capability is facilitated by the LSTM's ability to learn and remember relevant information from past observations, enabling it to make informed predictions about future trajectory movements.

Once the LSTM model generated predictions for the next time step, these predictions were compared with the ground truth spatial coordinates of the drone. Discrepancies between the predicted and actual positions provide feedback to the model, which is used to update its parameters during the training process. By iteratively adjusting the parameters based on this feedback, the LSTM gradually improved its predictive accuracy over time. The training and refinement process is crucial for ensuring that the LSTM effectively learns the underlying dynamics of the drone movements and can make accurate predictions even in complex and unpredictable scenarios. Additionally, the LSTM's ability to handle sequential data makes it well-suited for modeling dynamic systems like the drone trajectory, where past observations significantly influence future behavior.

4. Training process

The training process of a neural network can be described by the following formula, which characterizes the updating of the network's weights at each training step:

$$W_{t+1} = W_t - \alpha * \nabla L(W_t), \tag{10}$$

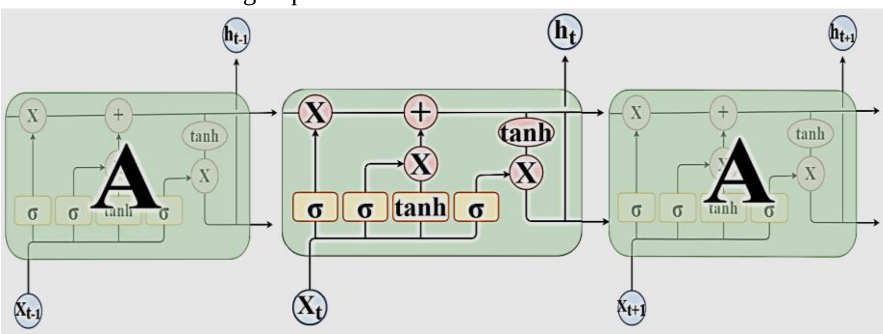

Fig. 4. Long Short-Term Memory

 ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

where, W_t represents the model weights at step t;

α – denotes the learning rate;

$\nabla L(W_t)$ – the gradient of the loss function relative to the weights at step t.

Training was conducted using the error backpropagation technique, which facilitated the efficient tuning of the network weights to minimize recognition and prediction errors. Backpropagation is a cornerstone neural network training algorithm that is particularly adept at dealing with complex tasks that involve high levels of computational intricacy, such as the prediction of object trajectories in video streams.

During backpropagation, after a forward pass through the network, where the inputs are processed layer by layer to produce an output, the error is calculated. This error is the difference between the predicted and actual desired output. The backpropagation algorithm then proceeds to calculate the error gradient with respect to each weight by the chain rule, propagating the error backward from the output layer to the input layer. This systematic approach allows for the identification and correction of each weight's contribution to the overall error.

This equation is a fundamental component of the backpropagation algorithm, where W_{t+1} are the updated weights after taking a step guided by the gradient of the loss function, and α is a scalar that controls the size of the step, which is known as the learning rate. The learning rate is a critical hyperparameter in the training of neural networks because it determines how much the weights are adjusted during each update. If the learning rate is too large, the network may overshoot the minimum of the loss function, which results in divergent behavior. Conversely, if the learning rate is too small, the training process may become excessively slow and become stuck in local minima.

The gradient $\nabla L(W_t)$ provides the direction in which the loss function increases most rapidly. Therefore, to minimize loss, we move in the opposite direction, i.e., we subtract the gradient from the current weights. This process is repeated iteratively over many epochs or iterations, with each update aiming to reduce the loss function until the algorithm converges to a minimum, ideally the global minimum [12].

Further, by extending this concept, other variations of the gradient descent algorithm can be introduced to enhance the training process. For instance, momentum-based methods, such as SGD with momentum, can help accelerate gradient vectors in the right direction, thereby leading to faster convergence [13]:

$$V_{t+1} = \beta * V_t + \nabla L(W_t), \quad (11)$$

$$W_{t+1} = W_t - \alpha * V_{t+1}. \quad (12)$$

where V_t represents the velocity (i.e., the accumulated gradient);

β – the momentum coefficient, typically set between 0.9 and 0.99.

The velocity term helps to smooth out the updates and can also help to navigate the rough landscapes of the loss functions more effectively.

In addition to momentum, algorithms such as RMSprop and Adam introduce adaptive learning rates for each parameter as follows:

$$S_{t+1} = \delta * S_t + (1 - \delta) * (\nabla L(W_t))^2, \quad (13)$$

$$W_{t+1} = W_t - \frac{\alpha}{\sqrt{S_{t+1} + \epsilon}} * \nabla L(W_t). \quad (14)$$

In the RMSprop update rule, S_t is the running average of the squared gradients, δ is the decay rate, and ϵ is a small scalar added to the denominator to avoid division by zero. This adaptive mechanism helps reduce the gradient of weights that receive large updates, thereby leading to more stable and efficient training.

Carefully crafted animation data were created in a blender and used to teach the model to recognize and predict the drone’s flight behavior. Convolutional layers are adept at extracting features from images and are essential for interpreting the visual information of a drone against various backgrounds and lighting conditions [14]. The convolution operation within these layers can be represented as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau, \quad (15)$$

where f is the image function, g is the kernel function, and $*$ denotes the convolution operation.

The neural network was trained using backpropagation, adjusting its weights to minimize the error between the predicted and actual drone positions. The loss function used could have been the mean squared error (MSE), which is standard for regression problems (16).

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \quad (16)$$

where y_i is the true value (the actual frame position of the drone) and $\hat{y}_i$ is the value predicted by the network, where n is the number of samples.

By iterating over numerous epochs, the network optimizes its weights to reduce this loss, effectively learning to predict the drone's trajectory based on the animated data. This method not only provided a safe and controlled way to generate training data but also allowed for the inclusion of scenarios that might be rare or difficult to capture in real-life footage, thereby enhancing the neural network's exposure and experience.

By utilizing these advanced optimization techniques, the training of neural networks becomes more reliable and efficient, allowing for the development of sophisticated models that can perform a wide array of tasks, from simple classification to complex trajectory predictions in dynamic environments.

In addition, this approach fosters continuous improvement and adaptation, ensuring that the neural network remains robust and adaptable to evolving challenges and environments. This iterative process not only enhances the network's accuracy and generalizability, enabling it to excel in real-world scenarios beyond the scope of its training data.

5. Results

The RCNN model was trained for recognition through a multi-step process. Initially, a large dataset containing images with labeled instances of drones was compiled, including both actual video footage of witnesses and images taken in Blender. These images were then used to pretrain the model on generic visual features through techniques like supervised learning with convolutional neural networks (CNNs). During training, the model learned to distinguish between different objects, including the drone, by iteratively adjusting its parameters to minimize a predefined loss function. The performance of the trained RCNN model was evaluated using separate validation datasets and measuring metrics such as precision, recall, and mean average precision (mAP) across different images. Through this iterative training process, the RCNN model achieved high accuracy in recognizing drones within images, demonstrating its effectiveness in real-world applications.

Figure 5 shows some examples of well-recognized objects using the proposed model. The model's effectiveness in analyzing objects was commendable for medium- to large scale analyses. However, it exhibited inaccuracies and errors when tasked with too small resolutions or objects that were diminutive. Its performance was reduced when confronted with minute details or miniature entities. Despite its proficiency at larger scales, the model struggled to maintain precision when evaluating smaller elements.

Fine-tuning may be necessary to enhance the accuracy of the model for objects of various scales and sizes. More precise parameters of the results are presented in Table 1.

To calculate the mean accuracy (mAP) and mean completeness (mAR), we averaged the AP and AR values over all classes and areas of the image.

Fig. 5. Drone detection

First, let's average the AP values over all classes and image areas as follows:

$$\mathrm{mAP} = (77 + 65 + 82 + 45 + 71 + 73) / 6 \approx 68.5$$

Now let's average the AR values over all classes and areas of the image:

$$\mathrm{mAR} = (65 + 69 + 78 + 40 + 71 + 78) / 6 \approx 66.83$$

Thus, the average accuracy (mAP) is approximately 68.5 and the average completeness (mAR) is approximately 66.83.

6. Trajectory prediction

The motion trajectory is predicted based on the trajectory data provided by all drones. It is assumed that each scene undergoes preprocessing, yielding spatial coordinates for each i-th subject at time t.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Table 1

Metrics for assessing model quality

Metrics	IoU	Max detect	Area	Value, %
Average Precision (AP)	0.50:0.95	100	all	77
Average Precision (AP)	0.50	100	all	65
Average Precision (AP)	0.75	100	all	82
Average Precision (AP)	0.5:0.95	100	small	45
Average Precision (AP)	0.5:0.95	100	medi-um	71
Average Precision (AP)	0.5:0.95	100	large	73
Average Recall (AR)	0.5:0.95	1	all	65
Average Recall (AR)	0.5:0.95	10	all	69
Average Recall (AR)	0.5:0.95	100	all	78
Average Recall (AR)	0.5:0.95	100	small	40
Average Recall (AR)	0.5:0.95	100	medi-um	71
Average Recall (AR)	0.5:0.95	100	large	78

The preprocessed information is derived from images, which are presented in bounding boxes encompassing the respective subjects of interest. The input comprises a vector of coordinates $\{x_t, y_t\}$ representing the minimum and maximum spatial extents of the subjects at time t.

We can compute the prediction error, which is the distance between the predicted location and the actual location at t-s in the future [15], as shown in equation (17)

$$\sqrt{\left(x_{t_0+t}-\tilde{x}_{t_0+t}\right)^2+\left(y_{t_0+t}-\tilde{y}_{t_0+t}\right)^2}. \tag{17}$$

This preprocessing step is crucial because it allows us to isolate and extract the relevant subjects from the scene, thereby enabling further analysis and prediction of their motion trajectories. By obtaining the spatial coordinates of each subject at different time points, we can observe their movements over time and infer potential patterns or trends in their behavior.

The changes in trajectory are shown in Figure 6. It becomes obvious that the model diligently attempts to predict future trajectories with sufficient accuracy. Nevertheless, performance errors were observed because the model lacked fine-tuning. Notably, straight routes exhibited higher accuracy. Conversely, deviations or turns in the trajectories can sometimes create problems, leading to increased detection errors in the model.

According to Table 2, for trajectory durations varying from 1 to 5 s, we obtained diverse data for different trajectory types. This allows us to evaluate the accuracy of motion prediction at different time intervals. The lowest average error was observed for slow motion in both short and long trajectories. At the same time, turns are characterized by a higher average error, especially for longer trajectories. These findings suggest that the difficulty of trajectory prediction is directly related to the type and duration of the motion [16].

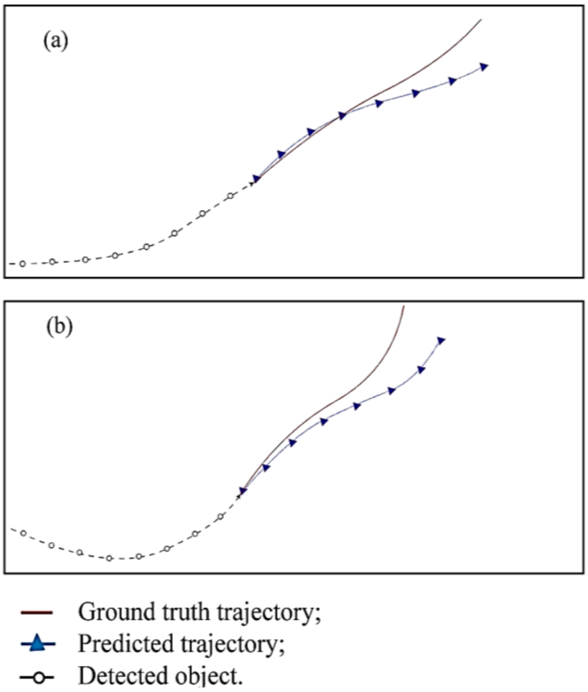

Fig. 6. Two subplots (a) and (b) illustrate the difference between ground truth (actual) and predicted trajectories for an object over a short-term prediction window of 1 s. In subplot (a), the object's trajectory appears to follow a more curved path, where the predicted trajectory is close to the actual trajectory but diverges slightly toward the end. In subplot (b), the trajectory follows a sharper curve, and the difference between the predicted and actual paths is more noticeable, with the predicted path diverging more significantly from the actual ground truth, especially further along the trajectory. The figure highlights the challenge of predicting trajectories, particularly when the object’s movement involves complex curves or changes in direction

Table 2

Average error of various route types

Route type	Average error, %	Duration, s
Straight path	12.5	1
Straight path	9.7	5
Turn	36.2	1
Turn	28.9	5
Slow motion	10.6	1
Slow motion	6.4	5

7. Discussion

The results presented in this study demonstrate the high performance of using a hybrid RCNN+LSTM architecture for detecting and predicting the trajectories of unmanned aerial vehicles (UAVs). The proposed approach is based on applying a convolutional region-based neural network (RCNN) to accurately recognize objects in video frames and a long-term short-term memory (LSTM) network to capture temporal dependencies and generate predictions of future drone positions. This combined method provides reliable detection accuracy, as shown in Fig.5 and reasonably accurate short-term trajectory prediction for a few seconds, as shown in Fig. 6. Difficulty consists of the irregular and curved motion of the object under study.

The proposed method includes the integration of a CNN-based detector with an LSTM time-series predictor and the use of a synthetic dataset created in Blender to cover a wide range of flight scenarios, lighting, and weather conditions. The real video data, although limited, provided further validation by confirming that the RCNN part maintains good detection performance (table 1) and that the LSTM module correctly estimates the drone's motion, including changes in direction (table 2).

Analysis of (table 1) indicates that the model possesses balanced recall and precision mAP ≈ 68.5% and mAR ≈ 66.83%, despite the fact that detection accuracy decreases when the drone takes up very few pixels in the image, which means small-scale objects. Table 2 shows that the smallest average errors (about 6.4–10.6%) were obtained on slow-motion flight trajectories, whereas turns lead to higher prediction errors of up to 28.9–36.2%. This suggests a general issue for recurrent approaches: steady motion is predicted to be sufficiently accurate, whereas sudden turning leads to higher discrepancy between predicted and real trajectories. Figure 6 also illustrates this issue, where one trajectory example shows little discrepancy between the predicted trajectory and ground truth, whereas the hard turn greatly amplifies the discrepancy. There are some limitations to this study. The results obtained show that the detection accuracy of tiny or distant objects decreases, indicating the need for improved optics or additional fine tuning. It is also important to consider that when working with synthetic data, although the synthetic flight trajectories are diverse, they do not cover all possible maneuvers. Therefore, it may be more difficult for the model to predict trajectory in the presence of unexpected directional changes.

In practice, these results can be applied to automated UAV tracking, the creation of safe air corridors in urban areas, and real-time security systems above residential buildings and other infrastructures.

The goal of this study is to enhance our model's functionality by expanding it from object recognition in videos and images to analyzing Google Earth data. The proposed experiment involves projecting the flight and motion trajectories of objects onto the Earth's surface and tracking their likely trajectories, as shown in Figure 7. Using this experimental design, we expect to gain valuable insights into the behavior of objects in different environments. Using Google Earth data promises to offer a new perspective that will enable more accurate analysis and enrich research.

Through this project, our purpose is to improve the accuracy of drone flight path prediction by developing a deep learning-based trajectory forecasting model and for this we need to advance our understanding of drone flight dynamics and enhance trajectory prediction by integrating Google Earth data. By projecting drone flight paths and motion onto the Earth's surface, we can track their likely trajectories with greater accuracy. This approach not only helps us refine prediction models but also opens up potential applications for drone navigation and monitoring in diverse environments. The use of Google Earth data will provide a new perspective, enabling more precise analysis and enriching research on drone behavior across different terrains.

Fig. 7. Example of using Google Earth

Using Google Earth can provide an additional perspective and the ability to more accurately analyze the behavior of an object in different environments and conditions, which in turn can advance our research work and increase its application potential.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

8. Conclusions

This paper demonstrated the effectiveness of a deep learning-based UAV trajectory prediction. A set of video and photo materials was prepared to train the neural network model, develop a model based on deep learning for trajectory prediction, and improve the prediction of UAV drone flight paths.

This study used a hybrid model combining region-based convolutional neural networks (RCNNs) and long short-term memory (LSTM) networks. RCNNs were used to accurately detect objects in individual frames, whereas LSTM networks captured temporal dependencies and predicted object trajectories across frames. The proposed approach allows the model to accurately track objects and predict their motion in dynamic scenes. By iteratively adjusting parameters based on feedback from predicted and actual positions, the model improved its prediction accuracy over time. In summary, the study demonstrated the effectiveness of combining spatial perception and temporal understanding in video analysis, which may be applicable to video surveillance systems and driver assistance technologies.

The experiment also contributes to our understanding of the dynamics of objects and their potential and allows us to more accurately predict the movements of objects on the Earth's surface.

The scientific novelty of the study is to demonstrate the effectiveness of the proposed approach for flight path prediction.

Future research development. The current dataset contains a large amount of synthetic data. The next step is to collect as many examples of real data as possible and supplement them, possibly with other objects, to diversify flight trajectories and track changes in trajectory dependence on the drone type.

Increasing the number of real-world data samples can be a challenging task because collecting such data requires time and resources. In addition, data diversity must be ensured to accurately reflect different flight scenarios and drone types. This consideration may include accounting for different meteorological conditions, terrain types, obstacles, and other factors that affect the flight path. Another challenge is the need to train deep learning models to accurately predict trajectories. This requires not only a large amount of data but also careful tuning of the model parameters and possibly the development of specialized algorithms to account for the peculiarities of different types of objects. In addition, possible changes in drone or object behavior over time must be considered, which may require constant updating and adaptation of prediction models.

Since the existing architecture is relatively simple, future research will focus on improving tracking and integrating the context into the model.

Contributions of authors: Conceptualization, methodology – **Oleksandr Bezsonov**; formulation of tasks, analysis – **Serhii Liashenko**; development of model, software, verification – **Sofiia Rutska, Kateryna Vashchenko**; analysis of results, visualization – **Oleg Rudenko**; writing – original draft preparation – **Sofiia Rutska, Kateryna Vashchenko**; writing – review and editing – **Oleksandr Bezsonov**.

Conflicts of Interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

This study was conducted without financial support.

Data availability

Data will be made available upon reasonable request.

Use of Artificial Intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current study.

Acknowledgments

The authors would like to thank the scientific and teaching staff of Kharkiv National University of Radio Electronics for their support.

All the authors have read and agreed to the published version of this manuscript.

References

1. Yue, J., Manocha, D., & Wang, H. Human trajectory prediction via neural social physics. *Computer Vision – ECCV 2022. ECCV 2022*. Lecture Notes in Computer Science, 2022, vol. 13694. Springer, Cham. DOI: 10.1007/978-3-031-19830-4_22.
2. Liu, Y., Yan, Q., & Alahi, A. Social NCE: Contrastive learning of socially-aware motion representations. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 15118-15129. DOI: 10.48550/arXiv.2012.11717.
3. Qin, W., Tang, J., Lu, C., & Lao, S. Trajectory prediction based on long short-term memory network and Kalman filter using hurricanes as an example. *Computational Geosciences*, 2021, vol. 25, pp. 1005-1023. DOI: 10.1007/s10596-021-10037-2.

4. Park, D., Ryu, H., Yang, Y., Cho, J., Kim, J., & Yoon, K. Leveraging future relationship reasoning for vehicle trajectory prediction., article no. 6558, DOI: 10.48550/arXiv.2305.14715.

5. Blain, J. M. The complete guide to Blender graphics: computer modeling & animation. AK Peters/CRC Press, 2019, ISBN: 9781032510552.

6. Hosen, M. S., Ahmmed, S., & Dekkati, S. Mastering 3D modeling in Blender: From novice to pro. ABC Research Alert, 2019, vol. 7, no. 3, pp. 169-180. DOI: 10.18034/ra.v7i3.654.

7. Zheng, Q., Yang, M., Tian, X., Jiang, N., & Wang, D. A full stage data augmentation method in deep convolutional neural network for natural image classification. Discrete Dynamics in Nature and Society, article no. 4706576, 2020. DOI: 10.1155/2020/4706576.

8. Strecha, C., Von Hansen, W., Van Gool, L., Fua, P., & Thoennessen, U. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1-8. DOI: 10.1109/CVPR.2008.4587706.

9. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W-Y., Dollar, P., & Girshick, R. Segment anything. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015-4026. DOI: 10.48550/arXiv.2304.02643.

10. Turan, M., Almalioglu, Y., Araujo, H., Konukoglu, E., & Sitti, M. Deep EndoVO: A recurrent convolutional neural network (RCNN) based visual odometry approach for endoscopic capsule robots. Neurocomputing, 2018, vol. 275, pp. 1861-1870. DOI: 10.48550/arXiv.1708.06822.

11. Li, J., Cheng, J. H., Shi, J. Y., & Huang, F. Brief introduction of back propagation (BP) neural network algorithm and its improvement. *Advances in Computer Science and Information Engineering*, 2012, vol. 2. Springer Berlin Heidelberg, pp. 553-558. DOI: 10.1007/978-3-642-30223-7_87.

12. Kingma, D. P., & Ba, J. Adam: A method for stochastic optimization. *3rd International Conference for Learning Representations*, 2015. DOI: 10.48550/arXiv.1412.6980.

13. Bharilya, V., & Kumar, N. Machine learning for autonomous vehicle’s trajectory prediction: A comprehensive survey, challenges, and future research directions. *Vehicular Communications*, 2024, vol. 46, article no. 100733. DOI: 10.48550/arXiv.2307.07527.

14. Payeur, P., Le-Huy, H., & Gosselin, C. Trajectory prediction for moving objects using artificial neural networks. *IEEE Transactions on Industrial Electronics*, 1995, vol. 42, no. 2, pp. 147-158. DOI: 10.1109/41.370380.

15. Giuliari, F., Hasan, I., Cristani, M., & Galasso, F. Transformer networks for trajectory forecasting. *25th International Conference on Pattern Recognition (ICPR),* 2021, pp. 10335-10342. DOI: 10.48550/arXiv.2003.08111.

16. An, Y., Liu, A., Liu, H., & Geng, L. Multidimensional trajectory prediction of UAV swarms based on dynamic graph neural network. *IEEE access*, 2024, vol. 12, pp. 57033-57042. DOI: 10.1109/ACCESS.2024.3391374.

Received 08.05.2024, Accepted 17.02.2025

ВИКОРИСТАННЯ НЕЙРОМЕРЕЖІ ГЛИБОКОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ ТРАЕКТОРІЇ БПЛА

О. О. Безсонов, С. О. Ляшенко, О. Г. Руденко,
С. П. Руцька, К. С. Ващенко

Предметом вивчення в статті є новий підхід з використанням нейронної мережі глибокого навчання, спеціально розробленої для прогнозування траєкторії польоту безпілотного літального апарату (БПЛА). **Метою** цієї статті є підвищення точності прогнозування траєкторії польоту дрона шляхом розробки моделі прогнозування траєкторії на основі глибокого навчання. **Завдання** полягало в тому, щоб зібрати та підготувати набір відео- та фото матеріалів для навчання нейронної мережі, розробити та впровадити модель глибокого навчання для прогнозування траєкторії та покращити прогнозування траєкторії польоту БПЛА шляхом оптимізації та валідації моделі. Використовувалися такі **методи** як створення синтетичного набору даних було створено за допомогою інструменту 3D-моделювання Blender, який полегшив генерацію анімації, що відображає різні сценарії польоту дронів. Ці сценарії включають різні умови навколишнього середовища та міські ландшафти, забезпечуючи надійний тренувальний майданчик для нейронної мережі. Для подальшого вдосконалення та тестування прогностичних можливостей моделі були використані реальні дані, зокрема відеоматеріали, зняті очевидцями. Архітектура нейронної мережі, що використовується, включає блоки довготривалої короткочасної пам'яті (LSTM), які вміють обробляти послідовні дані, що робить їх ідеальними для прогнозування динамічних траєкторій руху БПЛА. Процес навчання включав кілька етапів, починаючи з попереднього навчання на загальних візуальних ознаках, а потім точне налаштування на специфічні для БПЛА патерни руху. **Результати** дослідження показують, що нейронна мережа досягла високої точності в прогнозуванні траєкторії, причому модель показала кращу продуктивність в реальних

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

сценаріях порівняно з традиційними методами прогнозування траєкторії. Інтеграція LSTM дозволила ефективно навчатися та узагальнювати часові дані, фіксуючи складні патерни руху та взаємодію з навколишнім середовищем. Це дослідження не тільки демонструє можливість використання глибокого навчання для прогнозування траєкторії БПЛА, але й пропонує потенційні застосування у сфері цивільної безпеки чи логістики доставки, де прогнозування траєкторії в реальному часі може значно підвищити ефективність і швидкість прийняття рішень. **Висновки**. Наукова новизна отриманих результатів полягає в розробці та навчанні моделей глибокого навчання, спеціально призначених для прогнозування траєкторії польоту дронів. Це дослідження демонструє ефективність запропонованого підходу, демонструючи його здатність підвищувати точність прогнозування траєкторії БПЛА.

Ключові слова: нейронна мережа; глибоке навчання; прогнозування траєкторії; LSTM; 3D-моделі; синтетичний набір даних; траєкторія БПЛА.

Безсонов Олександр Олександрович – д-р техн. наук, проф., проф. каф. комп'ютерних інтелектуальних технологій та систем, Харківський національний університет радіоелектроніки, Харків, Україна.

Ляшенко Сергій Олексійович – д-р техн. наук, проф., проф. кафедри мехатроніки, безпеки життєдіяльності та управління якістю, Державний біотехнологічний університет, Харків, Україна.

Руденко Олег Григорійович – д-р техн. наук, проф., зав. каф. комп'ютерних інтелектуальних технологій та систем, Харківський національний університет радіоелектроніки, Харків, Україна.

Ващенко Катерина Сергіївна – студентка, Харківський національний університет радіоелектроніки, Харків, Україна.

Руцька Софія Петрівна – студентка, Харківський національний університет радіоелектроніки, Харків, Україна.

Oleksandr Bezsonov – Doctor of Technical Sciences, Professor, Professor at the Department of Computer Intelligence Technologies and Systems, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: oleksandr.bezsonov@nure.ua, ORCID: 0000-0001-6104-4275, Scopus Author ID: 57221227175.

Serhii Liashenko – Doctor of Technical Sciences, Professor, Professor at the Department of Mechatronics, Life Safety and Quality Management, State Biotechnological University, Kharkiv, Ukraine,
e-mail: lyashenkosa05@ukr.net, ORCID: 0000-0001-8304-9309, Scopus ID: 57203146990.

Oleg Rudenko – Doctor of Technical Sciences, Professor, Head of the Department of Computer Intelligent Technologies and Systems, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: oleh.rudenko@nure.ua, ORCID: 0000-0003-0859-2015, Scopus ID: 7005837246.

Kateryna Vashchenko – Student of the Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: kateryna.vashchenko@nure.ua, ORCID: 0009-0005-5449-2455, Scopus Author ID: 58266293600.

Sofiia Rutska – Student of the Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: sofiia.rutska@nure.ua, ORCID: 0009-0009-0575-8172, Scopus Author ID: 58266602300.