UAV Localization Using CNN-based Ground Landmark Detection Under GNSS-denied Situation

Young Seo[1], Chunggil Ra[2], Yeji Kim[1], Taegyun Kim[1], Seungkeun Kim[1], and Jinyoung Suk[1]

[1]Department of Aerospace Engineering, Chungnam National University, Daejeon 34134, Republic of Korea,
[2]UAV RnD Center, Dbrain, Daejeon 34104, Republic of Korea,
* Corresponding author : jsuk@cnu.ac.kr
http://uasg.cnu.ac.kr

Abstract. Modern unmanned aerial vehicle (UAV) technology has been widely used in unmanned transportation and monitoring. However, UAV flight largely depends on Global Navigation Satellite Systems (GNSS), such as the Global Positioning System (GPS), which are vulnerable to jamming and spoofing attacks. These vulnerabilities and potential system failures can significantly compromise the accuracy and safety of UAV operations. This study proposes a method to reduce dependency on GNSS by enhancing position estimation through landmark recognition techniques. The proposed system uses a deep learning-based object detection algorithm to compare and analyze key features from a video map with landmark images. We estimate the position of the UAV by calculating the relative distance to these landmarks. The effectiveness of the proposed algorithm is further validated using Unreal Engine.

Keywords: Video-Based Navigation, Landmark Detection, Unreal Engine, GNSS-denied Situation

1 Introduction

Global Navigation Satellite Systems (GNSS) is a technology that provides critical positioning and timing information across various fields, including UAV. Despite its widespread use, civilian GNSS signals are inherently vulnerable to jamming and spoofing attacks due to their low transmission power, approximately -160 dBW, and open-access design[1]. These vulnerabilities pose significant threats to the accuracy and safety of UAV systems, underscoring the need for alternative positioning methods to reduce dependence on GNSS.

Previous studies[2] have proposed various methods to mitigate the vulnerabilities associated with GNSS signals. These methods are generally divided into encryption-based and non-encryption-based approaches. Encryption-based methods involve integrating encrypted authentication signatures into civilian GNSS signals to verify the integrity of the received data. Non-encryption-based

methods include techniques such as using jamming-to-noise (J/N) sensor to detect unexpected changes in GNSS signal strength or employing multi-system and multi-frequency defense strategies to identify inconsistencies among multiple GNSS signals. However, these methods have several limitations, such as high costs, complex implementation processes, and the requirement for expensive hardware, which restrict their widespread adoption in civilian systems. Additionally, some methods remain vulnerable to low-power spoofing signals[3]. These issues highlight the need for more practical and effective alternatives to enhance UAV positioning accuracy and reliability without relying solely on GNSS. To address these issues, this paper proposes a method for recognizing landmarks using a convolutional neural network (CNN)-based object detection model trained through pre-flight simulations in a virtual environment created with Unreal Engine. The proposed method estimates the relative positions of landmarks by utilizing the pan and tilt angles of the Gimbal camera. By leveraging the bounding box information from the CNN-based model and combining it with the camera's rotation data, this method enables the estimation of landmark positions in 3D space, even in GPS-denied environments, using only visual information from the camera.

The rest of this paper are organized as follows: Section 2 provides a preliminary overview, outlining the basic assumptions and data collection methods used in this study. Section 3 presents the relative position estimation method developed to enhance UAV positioning using landmark recognition. Section 4 discusses the simulation results, demonstrating the effectiveness of the proposed method. Finally, the paper concludes with a summary of the findings and suggestions for future research directions.

2 Preliminary

The assumptions for data acquisition are as follows: The camera mounted on the UAV is assumed to follow the pinhole camera model with a fixed focal length f, which is used to calculate the scale of objects in the image and determine their relative distances from the camera. Additionally, the UAV is expected to perform an orbital flight around the landmarks while maintaining a constant radius, and its altitude is kept constant during this flight.

2.1 Data Collection and Training

A real-world environment was created using the high-performance rendering simulator Unreal Engine and the 3D geospatial application Cesium. Since Cesium references the WGS84 coordinate system, it provides accurate geographic data in the simulation environment, enabling realistic simulations. By adding 3D objects to the map layers, a virtual environment resembling the real world was constructed.

Figure 1 represents the process flow, which consists of several steps. Assuming the search area is surveyed before the flight, images of the landmarks are

processed and labeled using OpenCV. This prepares the dataset for the model's training. During training, the feature extraction portion of the dataset remains unchanged, and only the final classifier is custom-trained, with images classified into predefined labels in a one-dimensional array format[4]. The classes are composed of the coordinates of the landmarks (targets), and the training was conducted in a Google Colab GPU T4 environment using a dataset of 723 images as the baseline. As the number of training epochs increases, errors in the training data decrease; however, this also leads to overfitting, which reduces the model's performance. The training process is monitored to assess the error rate and address overfitting, with the algorithm modified to include early stopping.

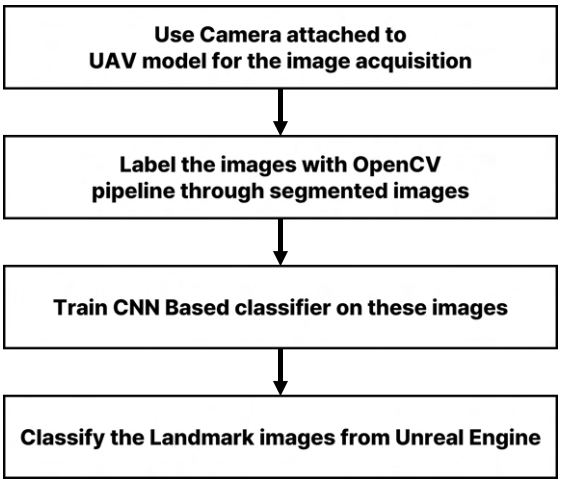


Fig. 1. Workflow for Landmark Image Classification

2.2 CNN-based Object Detection Model

CNNs are deep learning models that extract local features from input images through multiple layers of convolutional operations, enabling object recognition based on these features. The model utilized in this study is the You Only Look Once (YOLO) model, which operates on a CNN framework and performs object detection in a single network pass (Forward Pass). This approach demonstrates superior speed and efficiency compared to traditional object detection algorithms.

Figure 2 shows the structure of the CNN-based object detection model, based on the CNN architecture and consisting of three parts: Backbone, Neck, and Head. The Backbone utilizes Cross Stage Partial Network (CSPNet)-based CSP-Darknet53 to extract various levels of features from input images. This structure minimizes redundant calculations within the network, enabling deeper network

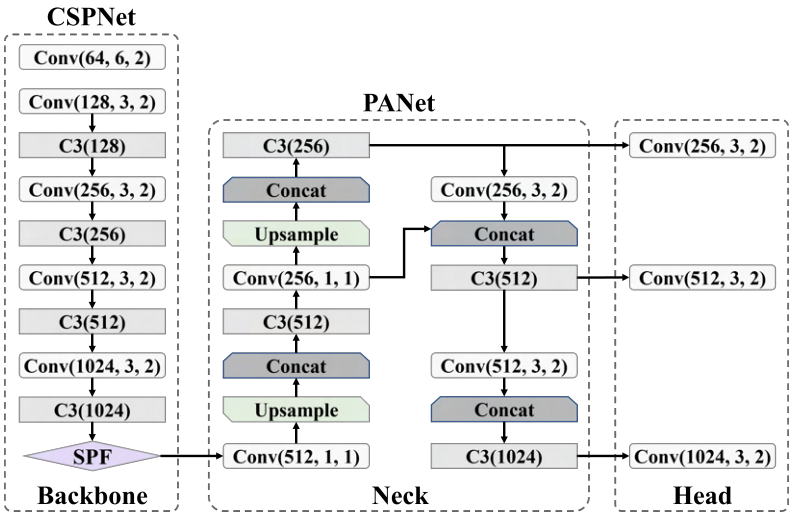


Fig. 2. The structure of a CNN-based object detection model

learning and deriving efficient feature representations, which makes real-time detection possible. The Neck is responsible for effectively integrating feature maps and combining information at various resolutions. It uses Path Aggregation Network (PANet) to merge multi-resolution features extracted from the Backbone, enhancing the detection performance for objects of different sizes, from small to large. The Head is the final part that predicts objects' location, size, and class probability. It employs anchor boxes to generate bounding boxes of various sizes and aspect ratios, allowing more accurate object location and size prediction. The bounding box coordinates (x, y), width and height (w, h), and confidence scores are output as a multidimensional array[5]. The model processes input images in one pass to extract features, integrates multi-resolution information, and predicts object location, size, and class probability through this structure, enabling real-time object detection.

3 Relative Position Estimation Method

Figure 3 represents a virtual environment based on real-world coordinates. In Unreal Engine, after receiving input parameters for roll, pitch, yaw, and speed, the system enables flight relative to the target object using the aircraft-gimbal camera actor. The angles of the UAV (φ, θ, ψ) and the aircraft's position in 3D space (x, y, z) are computed. Eqs. (1) and (2) use the UAV's speed (v), time change (Δt), and yaw angle (ψ) to calculate the distances traveled along the X and Y axes, and Eq. (3) is used to update the aircraft's position[6],[7].

Fig. 3. WGS84 coordinate-based virtual environment

$$X = v \cdot \Delta t \cdot \sin\left(\frac{\psi\pi}{180}\right) \tag{1}$$

$$Y = v \cdot \Delta t \cdot \cos\left(\frac{\psi\pi}{180}\right) \tag{2}$$

$$\text{New Position} = (\text{current}_x + X, \text{current}_y + Y, \text{current}_z) \tag{3}$$

3.1 Relative Distance Estimation:

After the UAV detects the landmark, the relative distance to the landmark is estimated using the tilt and pan angles of the camera.

The relative distance is estimated through three steps as follows:

– **Initial Distance Calculation**

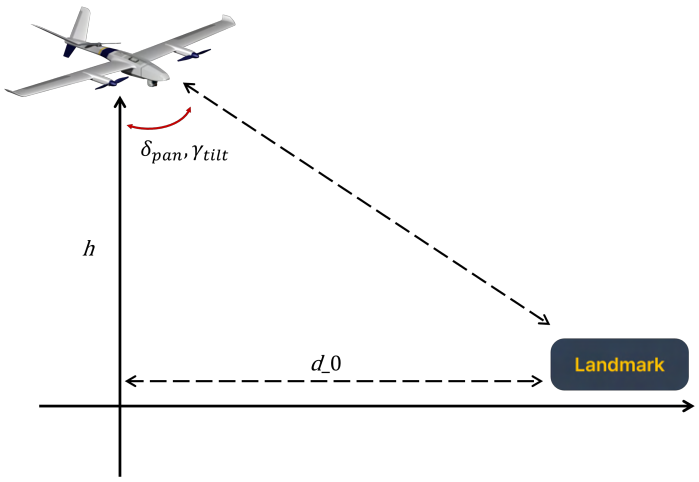


Fig. 4. Initial Distance Calculation of the UAV Toward the Landmark

The UAV estimates the initial distance to the landmark using the camera's tilt angle δ_{tilt} and the altitude h. This initial distance calculation is visually represented in Figure. 4, and Eq. (4) calculates the initial distance d_0 based on these parameters.

$$\mathrm{d}_0 = \frac{h}{\tan(\delta_{\text{tilt}})} \tag{4}$$

– **Calculation of Horizontal and Vertical Distance Components**
The initial distance d_0 is calculated according to Eq. (4). Using the camera's pan angle γ_{pan}, the horizontal distance d_1 and vertical distance d_2 to the landmark are determined with Eqs. (5) and (6):

$$\mathrm{d}_1 = \mathrm{d}_0 \times \cos(\gamma_{\text{pan}}) \tag{5}$$

$$\mathrm{d}_2 = \mathrm{d}_0 \times \sin(\gamma_{\text{pan}}) \tag{6}$$

Here, γ_{pan} represents the horizontal rotation angle of the camera, which allows for the estimation of the relative distances to the landmark.

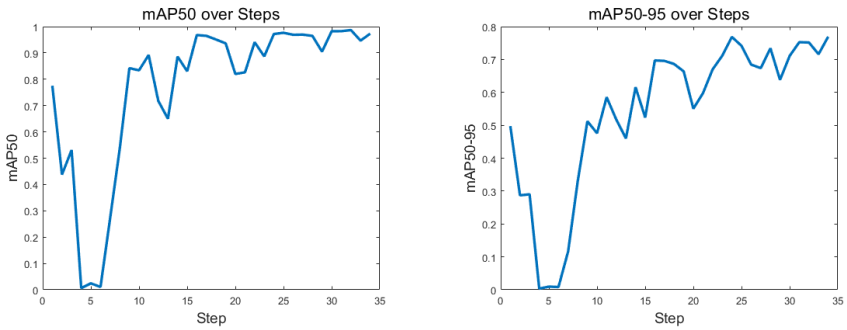
– **Calculation and Update of the Position Vector**
The UAV's position can be expressed as a vector in a three-dimensional space defined relative to the landmarks. This allows for real-time estimation of the UAV's position

$$\begin{bmatrix} \mathrm{d}_1 \\ \mathrm{d}_2 \\ h \end{bmatrix} = \begin{bmatrix} \mathrm{d}_0 \times \cos(\gamma_{\text{pan}}) \\ \mathrm{d}_0 \times \sin(\gamma_{\text{pan}}) \\ h \end{bmatrix} \tag{7}$$

Equation (7) represents the UAV's relative position and is used to update the three-dimensional coordinates in real-time from the landmark reference point.

4 Model Training Performance

Figure 5 is the Mean Average Precision (mAP) metrics, critical indicators of the model's object detection performance. The mAP is evaluated at two levels: mAP50 and mAP50-95. In Figure. 5(a), mAP50 denotes the average precision at an Intersection over the Union (IoU) threshold of 0.5, where a predicted bounding box is considered correct if it overlaps with the ground truth by at least 50%. Initially, mAP50 exhibits significant fluctuations and lower values. However, as training progresses, it increases rapidly and stabilizes near 1.0, indicating enhanced detection accuracy. Initially, the mAP50 shows significant fluctuations and lower values, but as the training progresses, it increases rapidly and stabilizes close to 1.0, indicating that the model's detection accuracy has significantly improved. In Figure. 5(b), mAP50-95 represents the mean average

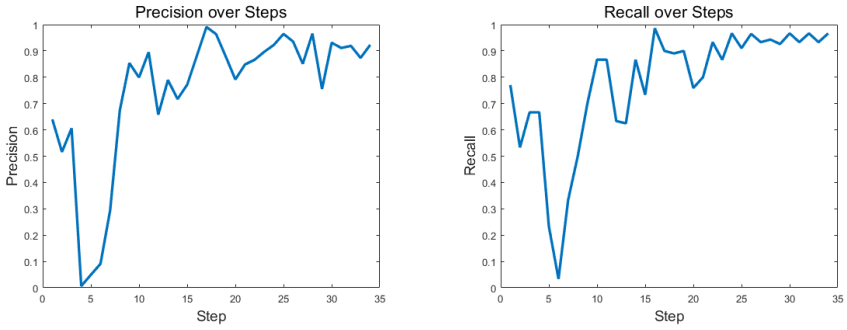


(a) mAP at IoU 0.5 (b) mAP at IoU 0.5 to 0.95

Fig. 5. Mean Average Precision (mAP) metrics

precision across multiple IoU thresholds ranging from 0.5 to 0.95, which comprehensively evaluates the model's generalization ability across different object sizes and locations.

Like mAP50, mAP50-95 begins with lower and unstable values in the early stages of training but gradually rises and stabilizes as training continues. The increasing trends in both mAP metrics demonstrate that the model effectively learns to detect objects under various conditions [8]. Figure 6 is a depiction of the precision and recall metrics, which are fundamental indicators of the model's detection performance.



(a) precision (b) recall

Fig. 6. Precision and Recall metrics

In Figure. 6(a), precision represents the proportion of correctly identified objects among all detected objects. Precision values fluctuate during the initial stages of training but gradually stabilize at a high level as training progresses. This trend indicates a reduction in false positives, reflecting improved accuracy

in detecting actual objects. In Figure. 6(b), recall represents the proportion of actual objects correctly identified by the model. The recall graph initially shows low and unstable values, indicating the model's early difficulties in capturing all true objects. As training continues, recall improves and stabilizes at a high level, demonstrating the model's increasing effectiveness in identifying most actual objects without omissions. The combination of these precision and recall metrics indicates that the model enhances its ability to detect true objects while minimizing false negatives and positives [8].

5 Simulation Results

Fig. 7. Landmark Detection Results

The simulation is conducted in the Unreal Engine environment to validate the proposed UAV position estimation method. With initial flight parameters (speed, roll, pitch, yaw, pan angles, and tilt angles) provided, the UAV flew along a predetermined path while detecting the landmark (a bridge) in real time.

Using the initial input values, including the UAV's speed and various angles (roll, pitch, yaw, pan, and tilt), the UAV calculated its relative position to the target landmark within the simulation. As shown in Figure. 7, the UAV detected the landmark at coordinates X=125.595 and Y=38.689, and the detected landmark's location information (latitude 38.75256°, longitude 125.67654°) along with a confidence score of 0.79 was displayed. The error rates relative to the true values were 0.06% and 0.16% for the X and Y coordinates, respectively, as summarized in Table 1, indicating a high level of accuracy in the proposed method.

The UAV used the given parameters from the initial inputs to calculate the distance traveled based on the angles and estimated the relative distance and position to the landmark in real time. For example, the log messages indicate that the current yaw angle (ψ) of the UAV is approximately 6.208802°, which is

Table 1. error rate relative to the true value

Coordinate	True Value	Detected Value	Difference (%)
X Coordinate	125.595	125.67654	0.06%
Y Coordinate	38.689	38.75256	0.16%

combined with the camera's pan and tilt angles to assist in object detection and position estimation.

This simulation demonstrates that the proposed position estimation method can estimate the UAV's relative position to the landmark based on the initial input parameters, even without GNSS signals.

6 Conclusion

In this study, we proposed a new UAV position estimation method that reduces dependence on GNSS signals by utilizing a CNN-based object detection model and initial flight parameters. The proposed method allows the UAV to calculate its relative position to landmarks in real-time, even in environments where GNSS signals are disrupted or unavailable due to jamming or spoofing. The simulation conducted in the Unreal Engine environment demonstrated that the UAV could perform relative position estimation using predefined flight parameters and real-time landmark detection. The UAV effectively detected the target landmark and calculated its relative position based on the initial input parameters, with a lower error rate than the true values. Based on this study, future research will focus on enhancing the robustness of the proposed method in more complex environments and testing the system's performance with different types of landmarks.

Acknowledgements

This research was supported by Unmanned Vehicles Core Technology Research and Development Program through the National Research Foundation of Korea (NRF) and Unmanned Vehicle Advanced Research Center (UVARC) funded by the Ministry of Science and ICT, the Republic of Korea (2020M3C1C1A01083162).

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT)(No. 2021R1A5A1031868).

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No. NRF-2021R1A2C2013363).

References

1. Sathyamoorthy, D., et al.: Evaluation of the vulnerabilities of unmanned aerial vehicles (UAVs) to global positioning system (GPS) jamming and spoofing. Defence S and T Technical Bulletin 13, 333-343 (2020).
2. Humphreys, T. E.: Statement on the vulnerability of civil unmanned aerial vehicles and other systems to civil GPS. Proceedings (2012).
3. Jung, S., Kim, T., Shin, C., Lee, S.: Technical Trends of Smart Jamming for GPS Signal. Electronics and Telecommunications Research Institute 27 (2012)
4. Culjak, I., Abram, D., Pribanic, T., Dzapo, H., Cifrek, M.: A brief introduction to OpenCV. 2012 Proceedings of the 35th International Convention MIPRO, 1725-1730 (2012).
5. Yan, B., Fan, P., Lei, X., Liu, Z., Yang, F.: A Real-Time Apple Targets Detection Method for Picking Robot Based on Improved YOLOv5. Remote Sens. 13, 1619 (2021).
6. Liu, W., Zheng, Z., Cai, K.-Y.: Bi-level programming based real-time path planning for unmanned aerial vehicles. Knowledge-Based Systems 44, 34-47 (2013).
7. Cai, J., Luo, L., Hu, S.: Bi-direction Direct RGB-D Visual Odometry. Applied Artificial Intelligence 34, 1137-1158 (2020).
8. Jiang, P., Ergu, D., Liu, F., Cai, Y., Ma, B.: A Review of Yolo Algorithm Developments. Procedia Computer Science 199, 1066-1073 (2022).