2023 The 23rd International Conference on Control, Automation and Systems (ICCAS 2023)
Yeosu Venezia Hotel & Utop Marina Hotel, Yeosu, Korea, Oct. 17∼20, 2023

Indoor Pedestrian-Following System by a Drone with Edge Computing and Neural Networks: Part 2 - Development of Tracking System and Monocular Depth Estimation

Jung-Il Ham[1], In-Chan Ryu[2], Jun-Oh Park[3], Jae-Woo Joeng[4], Sung-Chang Kim[5], Hyo-Sung Ahn[6,†]

[1,2,3,4,6]School of Mechanical Engineering, Gwangju Institute of Science and Technology,
Gwangju, 61005, Korea
([1]jungilham@gm.gist.ac.kr, [2]inchanryu@gm.gist.ac.kr, [3]junoingist@gm.gist.ac.kr, [4]ju.jeong@gm.gist.ac.kr)
([6,†]hyosung@gist.ac.kr)
[5]Edge Computing Application Service Research Section, Electronics and Telecommunications Research Institute
Gwangju, 61011, Korea
([5]sungchang@etri.re.kr)

Abstract: This paper is the second installment in a series on indoor drone pedestrian tracking utilizing edge computing and neural networks. Building upon the SLAM and EKF technologies introduced in Part 1, this paper introduces Monocular Depth Estimation to reduce camera costs and overall weight. The system leverages AI-driven depth information for indoor positioning and real-time human tracking. Experiments demonstrate the drone's ability to autonomously track a specific individual indoors using vision and IMU sensors. Key contributions encompass an AI-based tracking system employing YOLO v3 and a novel depth estimation approach that supersedes traditional depth cameras.

Keywords: Indoor Drone, Pedestrian Tracking, Edge Computing, Neural Networks, Monocular Depth Estimation, YOLO v3, AI-based Tracking.

1. INTRODUCTION

This paper is the second part of a two-part series investigating the development of indoor drone pedestrian following systems with edge computing and neural networks. The first part of this series introduced the flight control system, emphasizing the role of SLAM as a visual measurement, which is crucial for indoor drone navigation [1].

In this second part of the series, we utilize the system developed in Part 1 to address the issues of human detection and tracking, and furthermore, we have developed AI-based Monocular Depth Estimation technology.

In order to track a specific person for indoor positioning and mission performance through 3D vision, research on 3D mapping technology for autonomous flight of indoor unmanned vehicles, research on stable navigation of VPS-based unmanned vehicles, and research on AI-based user recognition and tracking technology are required. [2]

In this paper, we introduce a drone system that uses vision information in an indoor environment to position itself, detect, and track people in real-time based on AI. The ground control center converts the received information into bounding box information of a person trained with YOLO v3, localization information through SLAM, and location information of a person in a monocular camera through depth estimation. To verify that the system works well, an experiment was conducted to autonomously fly a drone to track a person using vision information and an IMU sensor without using GPS indoors. The results show that a drone that recognizes a specific person tracks it while maintaining a stable distance from the person. The contribution of this paper can be summarized as follows:

1. Developed a complex AI-based human following system and algorithm combining real-time onboard YOLO v3 and depth estimation.

2. In order to replace the depth camera, which is heavy to load on a drone, AI-based depth estimation, was used to replace the depth value of SLAM information with a mono camera.

This article is structured as follows. Section 2 describes YOLO and monocular depth estimation along with a simple tracking controller based on measured data. Section 3 presents the experiment, and Section 4 provides a comprehensive conclusion.

† Corresponding Author

2. PEDESTRIAN TRACKING

2.1 Pedestrian Detection

To develop autonomous drone tracking of pedestrians, we employed the YOLO v3 algorithm [3]. This algorithm is tailored to recognize specific pedestrians by training on their data. YOLO uses bounding boxes to probabilistically predict object type and location within trained images.

In our experiment, we considered only bounding boxes with a recognition rate of 95% or higher as humans. Using a dataset of 1500 images, we extracted 300 mannequin models from five locations. The model was configured with one class, 18 filters, a learning rate of 0.001, and 30,000 training iterations. We saved the weight file just before loss rate escalation. A ROS node enabled weight usage in our drone's software-in-the-loop (SITL) simulation.

Upon drone camera capture of a stationary or moving mannequin, a stable recognition signal was sent to

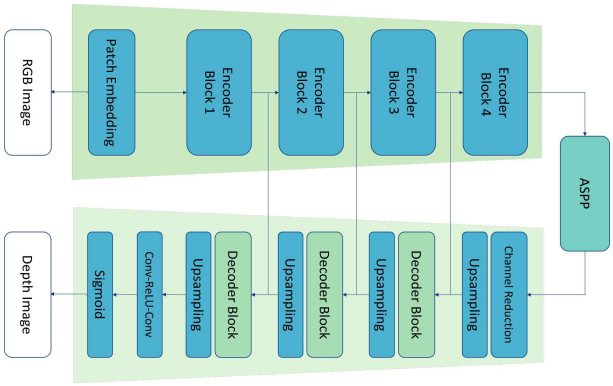

Fig. 1.: Overview of Our Network Structure

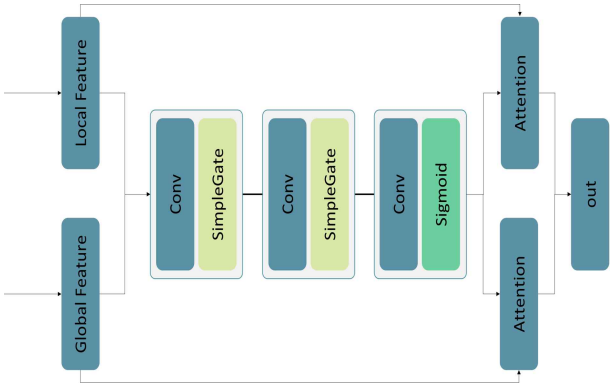

Fig. 2.: Structure of the DecoderBlock

the ground control PC. Bounding box size exhibited less than 5% error from actual size, demonstrating reliability. Applying findings to indoor drone flight, we utilized the darknet ROS package. Rqt graph analysis confirmed successful transmission of YOLO v3 frames from the drone to the ground control PC, achieving frame rates of 75 to 100 fps.

2.2 Monocualr Depth Estimation

Usually, the stereo camera is mainly used for automated drone navigation. However, if one of the stereo cameras is broken or is covered by dust or an obstacle, the whole system is not going to work.

Also, in the case of extremely small drones, the stereo camera is not suitable because of its size and weight. To solve this problem, we used the monocular camera for depth estimation with the edge computing method.

Monocular depth estimation developed rapidly with the growth of deep learning methods. In vision-based control, depth is an important issue, and it can be used in various problems such as SLAM(Simultaneous Localization And Mapping), Visual Odometry, and obstacle avoidance.

However, the network size of monocular depth estimation is too large to be used in embedded systems. Therefore, we chose to use the edge computing method. The image information receives the real-time image from the drone and computes the network to generate the depth image at the ground PC. Then, only the control information is sent to the drone.

As the Fig1 The network has 4 Encoder blocks and 3 skip connections, ASPP(Atrous Spatial Pyramid Pooling) [4] module right after the encoder and 3 decoder blocks merges multi-scale features.

Recently, many models using Transformer have been used in the image segmentation field. In the monocular depth estimation, there are also attempts to use this. We implemented the encoder using SegFormer [5], which was used in GLPdepth [6]. SegFormer is a model that achieved SOTA in the image segmentation field by using Transformer and is able to encode images of any size using overlapping. Since the encoder can generate various size features, it has the advantage of being able to extract good performance with a simple decoder. We used the MiT B4 pre-trained with Image-net network for our encoder, introduced in SegFormer [5].

The decoder consists of channel reduction, DecoderBlock, ConvBlock, and Sigmoid. The back born structure of Decoder is using GLPdepth [6], we used the module by replacing the BatchNorm-ReLU block with the SimpleGate block. The SimpleGate [7] is based on element-wise multiplication. So batch normalization is unnecessary for the decoder when using SimpleGate as an activation function.

We used the scale-invariant loss [8] as the training loss. The scale-invariant loss is as follows.

$$L = \frac{1}{n}\sum_{i} d_i^2 - \frac{1}{2n^2}(\sum_{i} d_i^2) \tag{1}$$

$$d_i = logy_i - logy_i^* \tag{2}$$

The y_i means the predicted depth map, and the y_i^* means the ground truth with n pixels indexed by i.

To train and evaluate our algorithms with the NYU Depth V2 dataset [9], which contains 640×480 indoor environment images and depth maps of indoor environments. We used pre-defined centre cropping by Eigen [8], which includes around 24K pictures and its depth image for evaluating our result.

We used the AdamW as an optimizer. The model was trained for 50 epochs with a batch size of 10. The training GPU was Nvidia RTX 3090. We used the PyTorch [10] framework for our implementation.

The RMSE(Root Mean Square Error) has improved by almost 1.1%. Other values were also slightly improved in performance.

The table 1 is the accuracy compared with other methods. The result shows that the root means square error(RMSE) is 1.1% better than the GLPdepth method. Also, absolute relative error(abs rel), log10 error, $\delta < 0.25$ got better performance and $\delta^2 < 0.25$, $\delta^3 < 0.25$ was same.

2.3 Monocular Depth Estimation Refinement

In this section, we will talk about the estimation quality of the monocular depth estimation in a new environ-

Table 1.: Monocular depth estimation performance in NYU Depth V2 dataset.

Models	Parms(M)	RMSE	abs rel	log10	$\delta < 0.25$	$\delta^2 < 0.25$	$\delta^3 < 0.25$
Adabins [11]	78	0.364	0.103	0.044	0.903	0.984	0.997
BTS [12]	47	0.392	0.110	0.047	0.885	0.978	0.995
GLPdepth	62	0.344	0.098	0.042	0.915	**0.988**	**0.997**
Ours	67	**0.340**	**0.095**	**0.040**	**0.917**	**0.988**	**0.997**

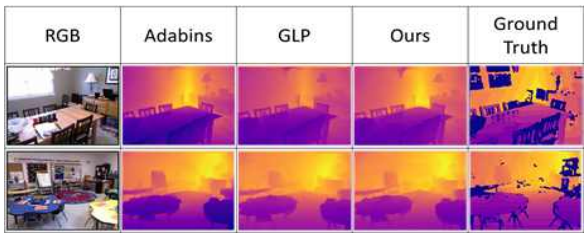

Fig. 3.: Images Depicting Monocular Depth Estimation Results

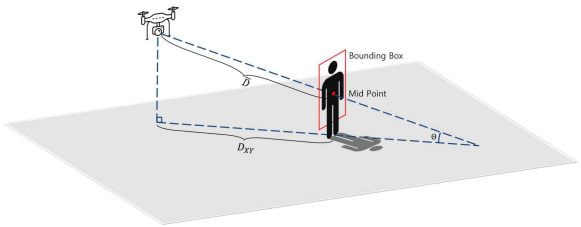

Fig. 4.: Distance Measurement Method

ment.

Scale is a big trouble in monocular depth estimation because there is no way to get accurate depth information in a single image. So, supervised monocular depth estimation in the new environment cannot guarantee accuracy compared with ground truth. There are several reasons why the supervised depth estimation cannot guarantee its accuracy, such as brightness, distortion, and angle of View.

The monocular depth estimation results from an indoor experiment show the orderly output in each environment, which means that if we offer the new environment, it shows the ordered information on which object is far or close.

With the assumption that information is ordered, linear correction can be applied. The modification consists of mainly two man-tuned parameters, which c_o correct offset and c_s are suggested.

$$D_c = c_o + c_s * \hat{D} \tag{3}$$

With raw information of monocular depth estimation $\hat{D}$ we get the D_c. By employing linear regression on the aforementioned model output and distance that is physically measured within the experimental environment, the values of c_o and c_s are found to be 1.29 and -0.89, respectively.

2.4 Relative Distance and Angle

To maintain a certain distance from the pedestrian, we made a ROS topic that measures the distance between the pedestrian and the drone using monocular depth estimation. The detection of the pedestrian by YOLO V3 and measuring the position of the bounding box of the pedestrian. Then, we estimate the average depth information of the 17 points around the location in the depth map using monocular depth estimation to measure the distance. The distance information estimated by monocular depth estimation is the distance in the direction of the camera's line of sight. Therefore, to measure the distance between the pedestrian and the drone D_{XY} from the estimated distance $\tilde{D}$, we need to multiply the simple trigonometric transformation shown in Fig. 4.

$$D_{XY} = \tilde{D} \times \cos(\theta) \tag{4}$$

The angle θ is the angle between the line of sight of the camera and the horizontal line. The rise θ is the pitch and offset angles summation.

The relative angle between the drone and the target can be calculated from the bounding box pixel and camera parameters.

$$\psi_r = atan2(W/2 - x_{obj}, l_f)$$

The midpoint of the camera frame on the x-coordinate side is denoted as $W/2$. Similarly, the midpoint of the bounding box on the x-coordinate side is represented as x_{obj}, while the camera's focal length is designated as l_f. Considering the intrinsic parameters of the camera and the camera frame size, the values of l_f and $W/2$ are calculated to be 700.819 and 320, respectively.

2.5 Tracking Control

This section proposes a controller that tracks detected pedestrians using both distance and pixel measurements obtained from a deep learning technique. If the controller's goal is only to maintain a specific distance between the drone and the detected pedestrian, an equilibrium set is established in the form of a circle centered around the pedestrian.

In scenarios where external forces are exerted on the drone, particularly phenomena like the ground effect—where propeller-generated wind is reflected off the floor and affects the drone -the controller and the drone moves around the circle which is the equilibrium set of controller, which is undesirable. Thus, having a controller with a unique equilibrium point within the map frame is highly desirable.

Fig. 5 illustrates the map frame denoted as X_m, Y_m, showing the drone's position $p_d = [x_d, y_d]^T$, its heading

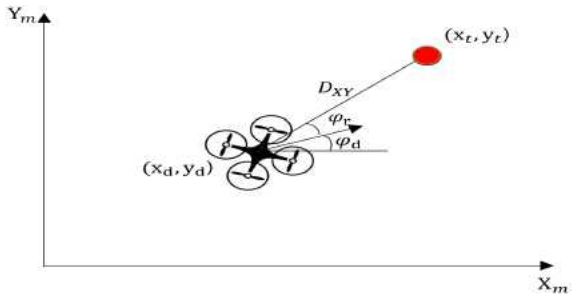

Fig. 5.: Map Coordinates and Relative Geometry of Drone and Pedestrian

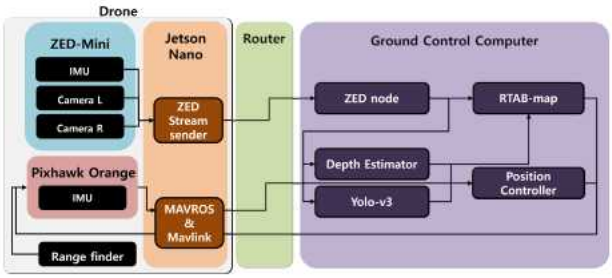

Fig. 6.: Diagram Illustrating the Overall System

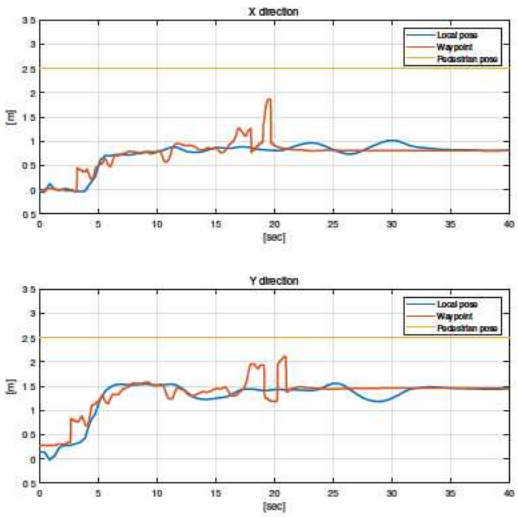

Fig. 7.: Flight Results for Drone-Pedestrian Distance Control

angle ψ_d, the target position $p_t = [x_t, y_t]^T$, the relative angle between the drone and the target ψ_r, and the relative distance between the drone and the target D_{XY}.

Assume the desired distance between the drone and the target is $D_{desired}$, and the intended offset is $o_{desired}$. The positional error between the drone and the target is defined as $p_e = p_d - p_t$. The controller design aims to achieve $\lim_{t\to\infty} \psi_r = 0$ and $\lim_{t\to\infty} p_e = o_{desired}$.

The target position in the map frame is unknown, but it can be expressed as follows: $p_t = p_d + D_{XY}[cos(\psi_d + \psi_r), sin(\psi_d + \psi_r)]^T$. Consequently, $p_e = p_d - p_t = D_{XY}[cos(\psi_d + \psi_r), sin(\psi_d + \psi_r)]^T$.

In this paper, We propose a simple P-type controller.

$$\psi_c = \psi_d + K_\psi \psi_r$$
$$p_c = p_d + K_p p_e - o_{desired}$$

p_c and ψ_c are drone position control commands and heading angle control commands respectively.

3. EXPERIMENT

3.1 Experimental Setup

The primary objective of the experiment is to maintain a predetermined distance between a pedestrian (simulated by a dummy) and a drone. In order to track the pedestrian, a straightforward algorithm has been devised.

The algorithm operates in two distinct modes. The initial mode activates when the pedestrian is not detected. In this mode, the drone initiates a 45° rotation to scan the surroundings. The subsequent mode engages when the drone detects the pedestrian. During this phase, the drone's Yaw angle is adjusted to align with the pedestrian's center, utilizing straightforward K_ψ control with a gain of 1. Additionally, the drone's position is adjusted using K_p control based on the difference between the desired distance and the current distance, employing a gain of 1.

To ensure precise mode changes only when the pedestrian detection or loss is certain, the "Detect/Not detect count" is established and updated within the control loop. Mode shifts are executed exclusively when the "Detect count" surpasses a predefined threshold. Specifically, as YOLO may not provide continuous detection even during the "Detected" mode, an increase in the "Not detect count" might inadvertently trigger a mode shift, leading to a 45° drone rotation. Consequently, to counteract this, an "Not detect count" reset is necessary whenever the "Detect count" increases by even a single instance.

We incorporated our newly developed monocular depth estimation technology and YOLO into the system introduced in Part 1. This integration has led to an updated system architecture, as illustrated in Fig. 6. The depth and coordinate information from the pedestrian's bounding box, extracted using YOLO and the depth estimator, play a pivotal role in the ground control PC's position control. Further details regarding the algorithms for the position controller are provided in the subsequent sections. To gain a visual understanding of the architecture constructed through ROS nodes and topics, we encourage you to view the demonstration video available at the following link: https://youtu.be/hDO4oRkx9R4.

3.2 Indoor Flight Experiment

For the experiment, we positioned the drone and the dummy diagonally across an empty room. We conducted the flight experiment with the drone initially unable to see the pedestrian. The initial separation between the pedestrian and the drone was approximately 2.5 me-

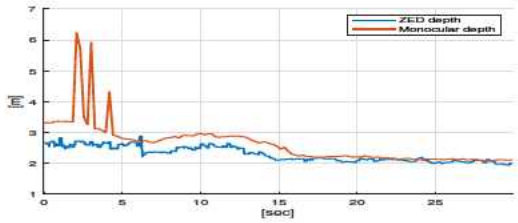

Fig. 8.: Comparison of Depth Information from ZED and Monocular Depth Estimation

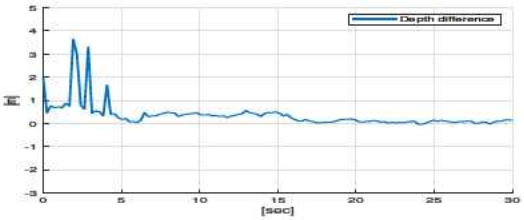

Fig. 9.: Difference in Pedestrian Depth Between ZED and Monocular Depth Estimation

ters both horizontally and vertically. The control parameters were set as follows: $D_{desired} = 2$ and $o_{desired} = [D_{desired}, D_{desired}]^T$.

In Fig. 7, you can observe the drone's movement (in blue), position command (in red), and the pedestrian's relative position (in yellow) to the drone's initial position in the x and y directions. The drone begins flying and then adjusts its heading angle by 45° to track the pedestrian. It flies towards the pedestrian for about 4 seconds. Afterward, the drone maintains a distance of 1.75 meters in the x-direction and 1 meter in the y-direction from the pedestrian, which results in an overall distance of 2 meters diagonally. To sustain this distance between the pedestrian and the drone, the center of the pedestrian is tracked, and the distance between the center of the pedestrian and the drone is used to control the drone's yaw.

In our approach, we determined the distance between the pedestrian and the drone by calculating the average distance from 17 points, centered around the bounding box obtained from the depth map. Given that the ZED camera has a depth accuracy of 1% [13] within the near range, we used the ZED camera as the ground truth for determining the accuracy of distance measurements. Fig. 8 displays the depth information from the ZED camera (blue) and the monocular camera (red). It's evident that the trend of the two depth information sources is nearly identical. Therefore, we have confirmed that the distance measurement method using a single camera can effectively replace the depth information obtained from the ZED camera.

Fig. 9 illustrates the difference in measured values between the ZED camera and our monocular depth estimation. It's worth noting that the error observed within the initial 7 seconds signifies that pedestrian detection has not yet taken place. The average absolute error was found to

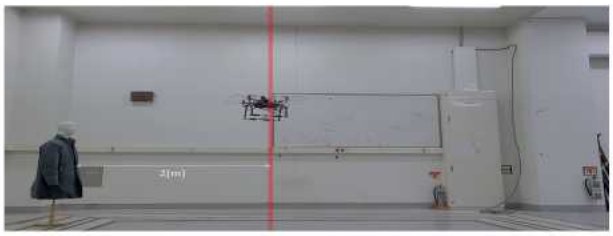

Fig. 10.: Results of the Flight Experiment

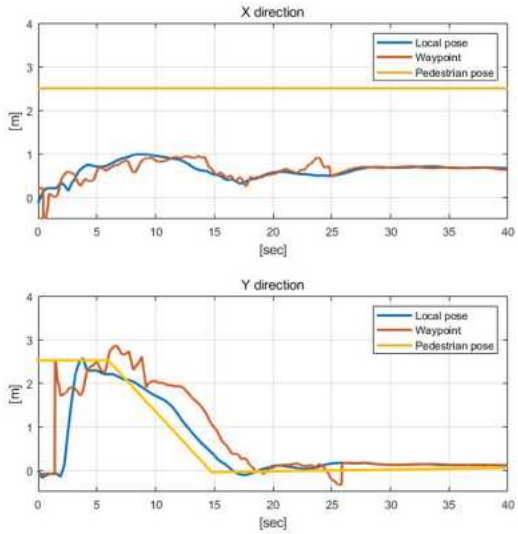

Fig. 11.: Flight Results for Drone-Pedestrian Distance Control During Pedestrian Movement

be 0.214, and the root mean square error (RMSE) was calculated to be 0.122.

As depicted in Fig. 10, the drone successfully converges to a distance of 2 meters. However, there is a slight oscillation. Given that the custom drone used in the experiment is relatively large for the environment, it's highly likely that significant ground effects were at play, exerting external forces on the system. When an unmodeled external force continuously affects the system, it is expected that the controller may not converge but instead maintains the drone around an equilibrium point. Therefore, it can be concluded that the drone's flight was conducted successfully even in challenging conditions where such external forces were in play.

We have successfully conducted tracking experiments not only when the pedestrian is stationary but also when it is in motion. Fig. 11 depicts the actual movement of the drone. This time, the pedestrian moved a distance of 2.5 meters in the y-direction. To demonstrate the tracking performance clearly, we set $o_{desired}$ as $[0, D_{desired}]^T$ so that the drone follows the pedestrian in parallel in the y direction. Upon observing the drone's movement (blue), we can see that after initiating the flight, the drone detects the pedestrian, moves to a position 2 meters away from the pedestrian in the x-direction, and as the pedestrian moves, the drone adjusts its position in the y-direction to

maintain a 2-meter distance. While there is some overshooting in both the x and y directions, this behavior can be attributed to the drone operating close to its maximum payload capacity. It's crucial to emphasize that the algorithm effectively generated the waypoint commands, and the drone's position control accurately followed the intended scenario. For a visual demonstration of this experiment, please refer to the following YouTube link: https://youtube.com/shorts/5YqxjTELpD8?feature=share.

4. CONCLUSION

This research presents the second part of a two-part series on an indoor drone system for pedestrian tracking, which utilizes edge computing and neural networks. Building on the previous installment that discussed SLAM and EKF integration for indoor drone flight, this paper focuses on using the developed system to address human detection and tracking challenges. Notably, it introduces Monocular Depth Estimation as a cost-effective alternative to traditional cameras.

The paper introduces an indoor drone system that utilizes AI for real-time human detection and tracking through vision data. This system processes data to generate pedestrian bounding boxes, SLAM-based localization details, and monocular camera-based pedestrian location data. Experimental results showcase the drone's ability to autonomously track pedestrians without GPS, maintaining a consistent distance from a specific individual.

In summary, this paper offers valuable insights into the development and application of an indoor drone system for pedestrian tracking, emphasizing the integration of edge computing, neural networks, and monocular depth estimation. The findings and developments presented have the potential to revolutionize indoor drone navigation and tracking systems.

5. ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT)(2022R1A2B5B03001459), and was also supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government [22ZK1100, Honam region regional industry-based ICT convergence technology advancement support project].

REFERENCES

[1] J.-I. Ham, I.-C. Ryu, J.-O. Park, J.-W. Joeng, S.-C. Kim, and H.-S. Ahn, "Indoor drone pedestrian following system with edge computing and neural networks. part 1: System design," Proceedings of the 23rd International Conference on Control, Automation and Systems (ICCAS 2023), Yeosu, Korea, Oct. 17∼20, 2023.

[2] M. Alhafnawi, H. B. Salameh, A. Masadeh, H. Al-Obiedollah, M. Ayyash, R. El-Khazali, and H. Elgala, "A survey of indoor and outdoor uav-based target tracking systems: Current status, challenges, technologies, and future directions," *IEEE Access*, 2023.

[3] L. Zhao and S. Li, "Object detection algorithm based on improved yolov3," *Electronics*, vol. 9, no. 3, p. 537, 2020.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[5] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[6] D. Kim, W. Ga, P. Ahn, D. Joo, S. Chun, and J. Kim, "Global-local path networks for monocular depth estimation with vertical cutdepth," *arXiv preprint arXiv:2201.07436*, 2022.

[7] L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple baselines for image restoration," *arXiv preprint arXiv:2204.04676*, 2022.

[8] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[9] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[10] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[11] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4009–4018, 2021.

[12] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, "From big to small: Multi-scale local planar guidance for monocular depth estimation," *arXiv preprint arXiv:1907.10326*, 2019.

[13] STEREOLABS, "A brief guide to recreational pyromania." http://www.blowinglotsofweirdstuffup.com/guide.html (2023/03/12).