2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

Behavior Cloning for Aircraft Autopilots with Semantic Segmentation under Various Lighting Conditions

Satoshi Hoshino Yudai Teranishi

Abstract— **In this paper, we propose an aircraft autopilot specifically designed for autonomous landing flights. The autopilot is trained through behavior cloning using a dataset of control command outputs provided by a human pilot in response to image inputs captured from the cockpit view. However, in unknown environments with different lighting conditions, even a trained autopilot struggles to determine appropriate control command outputs for visually different image inputs. To address this challenge and improve generalization capability across varying lighting conditions, we apply semantic segmentation to the original RGB images to classify runway pixels, and use the resulting segmentation images as inputs to the autopilot. Unlike RGB images, the segmentation images correctly classify only the runway regardless of lighting changes, producing visually consistent representations across all environments. Simulation experiments demonstrate that the proposed autopilot achieves improved generalization compared to the previous RGB-based autopilot, successfully landing on the runway in unknown evening and nighttime environments.**

I. INTRODUCTION

Amid a growing shortage of pilots [1], there is increasing demand for fully autonomous aircraft maneuvers. Existing autopilots enable aircraft to proceed toward their destinations at cruising altitude based on instrument readings after takeoff. However, landing flights are considered the most difficult because they involve physical contact between the aircraft and the runway. Therefore, landing is still performed by human pilots, who rely not only on instrument readings but also on visual observation from the cockpit. This paper is dedicated to the development of an aircraft autopilot system with a particular focus on landing flights.

Compared to the aircraft described above, vision-based approaches using RGB cameras have been applied to self-driving vehicles aiming for level 5 autonomy [2][3][4]. These autonomous driving systems were developed using deep neural networks trained via imitation learning [5], in which a human driver provides desired driving maneuvers such as steering, acceleration, and braking for given camera image inputs, allowing the system to learn the mapping from images to control command outputs. In the field of aviation, the authors have also proposed an autopilot system using a deep neural network trained through behavior cloning [6], where control surface commands are determined from cockpit-view RGB image inputs.

For the autopilot described above, a convolutional neural network (CNN) [8] was employed as the deep neural network. Behavior cloning, a fundamental approach in imitation learning, was used to train the autopilot: desired flight maneuvers using control surfaces were instructed by a human pilot for given image inputs, and the autopilot was trained to reproduce these maneuvers. Furthermore, we proposed a scenario classifier to estimate the current flight scenario as either descent or landing flare [7]. As a result, the aircraft was able to land successfully on the runway by switching between the two flight controllers in the autopilot according to the estimated scenario.

In our previous works [6][7], however, the aircraft based on the autopilot attempted landing flights only in environments where desired flight maneuvers using control surfaces had been instructed by a human pilot. In other words, the environments were already known to the aircraft. In contrast, aircraft landings in general are conducted under a wide range of airport conditions, including variations in lighting and weather. Therefore, to realize aircraft autopilots capable of landing, the ability to generalize to various unknown environments is essential. For our autopilot, the pixel values of RGB image inputs change depending on the flight environment. In such cases, the extracted features differ from those obtained during the instructed flights, making it difficult to perform landings toward the runway. Providing flight instructions that cover all possible environments would clearly increase the instruction cost in an imitation learning framework.

In this paper, lighting changes such as daytime, evening, and nighttime are treated as environmental conditions. In unknown environments with different lighting conditions, even a trained autopilot struggles to determine appropriate control command outputs for visually different image inputs. To address this challenge and improve generalization capability across varying lighting conditions, we apply semantic segmentation to the original RGB images to classify runway pixels. Furthermore, we propose an autopilot that uses the resulting segmentation images as inputs. Unlike RGB images, the segmentation images correctly classify only the runway regardless of lighting changes, producing visually consistent representations across all environments. Therefore, as long as the runway is successfully segmented from RGB images, the autopilot extracts image features similar to those obtained during the instructed flights.

The remainder of this paper is organized as follows. In Section II, we first describe the relationship between the aircraft's control surfaces and its flight dynamics. In Section III, we explain behavior cloning within the framework of imitation learning. In Section IV, we propose an autopilot that uses segmentation image inputs and develop it through behavior cloning. In Section V, we conduct landing flight

S. Hoshino and Y. Teranishi are with the Department of Mechanical and Intelligent Engineering, Utsunomiya University, 7-1-2 Yoto, Utsunomiya, Tochigi 321-8585, JAPAN hosino@cc.utsunomiya-u.ac.jp

978-1-6654-5784-2/26/$31.00 ©2026 IEEE

experiments in a simulator and compare two autopilots that use either RGB images or segmentation images as input, both in the instructed and unknown environments. Finally, Section VI concludes the paper.

II. FLIGHT DYNAMICS BASED ON CONTROL SURFACES

In order for an aircraft to fly autonomously without a human pilot, it must maintain its attitude appropriately through the autopilot. To this end, we first define the aircraft's controllable surfaces and their corresponding attitudes. **Fig. 1** illustrates the relationship between the aircraft's control surfaces and the resulting rotational motions.

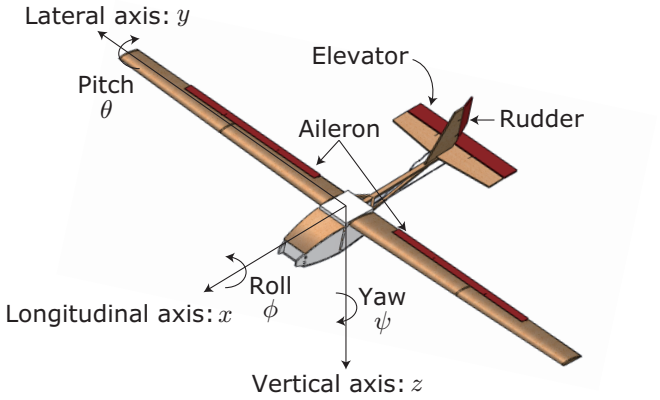


Fig. 1. Aircraft control surfaces: aileron, elevator, and rudder

The aircraft is equipped with control surfaces such as the ailerons, rudder, and elevator. By adjusting these surfaces, the aircraft can rotate about three axes: lateral, longitudinal, and vertical, each perpendicular to its center of gravity. The rotational motion about the lateral axis, y, is controlled by the elevator. Because the elevator governs the rotational motion about the lateral axis, y, it changes the pitch angle, θ, in the vertical direction. The rotational motions about the longitudinal and vertical axes, x and z, are controlled by the ailerons and the rudder, respectively. These surfaces adjust the roll and yaw angles, ϕ and ψ, in the lateral direction.

As described above, the motion of an aircraft can be separated into vertical and lateral components [13]. In terms of vertical attitude, the pitch angle, θ, is influenced by both thrust and elevator deflection. Thrust is controlled by the throttle lever. Therefore, in this paper, elevator deflection and throttle are treated as control targets in the vertical direction, while aileron and rudder deflections are treated as control targets in the horizontal direction. Finally, based on image inputs captured from the cockpit, the autopilot determines these control targets in both directions.

III. BEHAVIOR CLONING

In this paper, the proposed autopilot is developed using behavior cloning. In behavior cloning, a human pilot first provides desired flight maneuvers using the control surfaces for a given set of image inputs. The autopilot is then trained to reproduce these maneuvers through supervised learning with the instructed dataset. **Fig. 2** illustrates the learning framework, which consists of two phases: instruction and training.

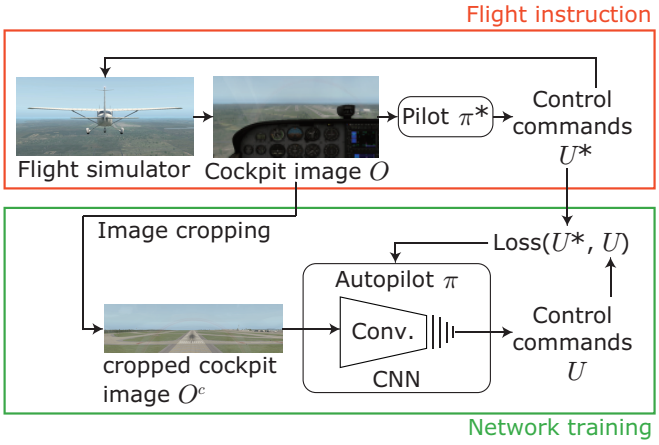


Fig. 2. Flight instruction and network training for autopilot

The autopilot system is developed in a flight simulator. In the flight instruction phase, a human pilot visually observes an RGB image showing the view from the cockpit and determines the control commands U^* corresponding to the elevator, throttle, aileron, and rudder, as described in Section II. While the aircraft is operated by the human pilot, the instruction dataset is recorded as follows:

$$D = \{\langle O_i, {U_i}^* \rangle\}_{i=1}^{N}, \tag{1}$$

where O_i represents the view from the cockpit. It is an RGB image cropped to show the cockpit windshield area, as illustrated by O^c in Fig. 2. ${U_i}^*$ denotes the control command provided by the human pilot based on the cockpit view O_i. N denotes the size of the dataset.

In behavior cloning, it is assumed that the supervisor, that is, the human pilot, follows an optimal policy π^* and determines the optimal control command outputs U^* for the observed image inputs O. This relationship can be expressed as ${U_i}^* = \pi^*(O_i)$. Similarly, the autopilot's policy π, which determines the control command outputs U for the cropped image inputs O^c, is represented as follows:

$$U_i = \pi(O_i^c; w), \tag{2}$$

where w represents the parameters of the CNN, including the filters and connection weights.

In the network training phase, the parameters w in Eq. (2) are updated to minimize the following objective function:

$$\underset{w}{\text{minimize}} \sum_i \ell(\pi^*(O_i), \pi(O_i^c; w)), \tag{3}$$

where ℓ represents the loss function. As illustrated in Fig. 2, the loss is computed based on ${U_i}^*$ and U_i, which respectively denote the optimal control command output instructed by the human pilot and the control command determined by the autopilot according to its policy during network training.

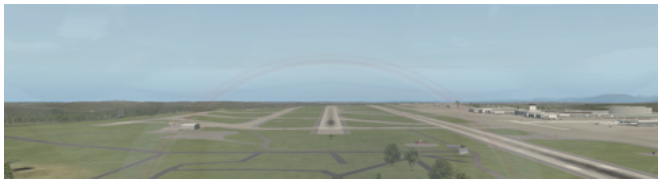
Through the optimization process using Eq. (2), the autopilot's policy π is ultimately optimized to approximate the

optimal human policy π^*. Therefore, after behavior cloning, the optimized policy π in Eq. (2) is used in the autopilot.

IV. AUTOPILOT USING SEGMENTATION IMAGES

A. Classification of Runway Using Semantic Segmentation

In our previous autopilots [6][7], which used RGB image inputs, the aircraft had difficulty landing on the runway when lighting conditions changed, as these were treated as unknown flight environments. This difficulty arose because the pixel values of the RGB images changed with lighting conditions. For example, **Fig. 3** shows RGB images captured from the cockpit view at the same position in daytime, evening, and nighttime environments at the same airport.



(a) Daytime environment

(b) Evening environment

(c) Nighttime environment

Fig. 3. RGB images in different lighting environments

In Fig. 3(a), the position and shape of the runway are clearly visible in the image. However, in Fig. 3(b), the runway becomes less distinct. In Fig. 3(c), the position and shape of the runway are barely recognizable without approach lights. Even at the same position, the pixel values of RGB images can vary significantly depending on lighting conditions.

For the previous autopilots, landing flight instructions were provided in the daytime environment, as illustrated in Fig. 3(a). Therefore, while daytime image inputs are known to the autopilots, the other two inputs captured in the evening and nighttime environments, as illustrated in Fig. 3(b) and Fig. 3(c), are unknown. For these visually different image inputs, the extracted features differ from those obtained during the instructed flights. As a result, even a trained autopilot struggles to determine appropriate control command outputs for landing on the runway.

To address this challenge and improve generalization capability across varying lighting conditions, we classify the pixels in RGB images into two classes, runway and non-runway, using U-Net [10], a widely used semantic segmentation technique. **Fig. 4** illustrates segmentation images generated from the same RGB image under different lighting conditions using U-Net.

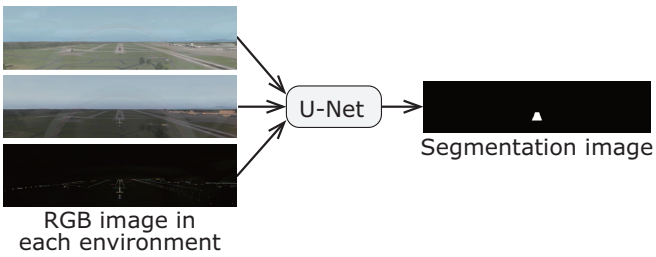


Fig. 4. Generated segmentation image using U-Net

Using U-Net, each pixel in the RGB images is classified as either runway or non-runway. Pixels corresponding to the runway are shown in white, while all other pixels are shown in black. As a result, visually similar segmentation images are generated across different environments.

Even though the landing flight instructions were provided in the daytime environment, as illustrated in Fig. 3(a), the segmentation images used as inputs to the proposed autopilot, which were captured in daytime, evening, and nighttime environments, appear visually similar. Therefore, the aircraft equipped with the autopilot is able to determine appropriate control commands for landing without being affected by environmental changes.

B. Proposed Autopilot

As described in Section IV-A, the proposed autopilot determines control command outputs based on segmentation image inputs, where only the runway is shown in white. For these image inputs, the autopilot is developed using a CNN. **Fig. 5** illustrates the architecture of the autopilot. In the convolutional layers, each filter is specified as (channels|size|stride). As described in Section II, the motion of an aircraft can be represented separately in the vertical and lateral directions. For this reason, the autopilot includes separate branches for the vertical and lateral outputs within the fully connected layers.

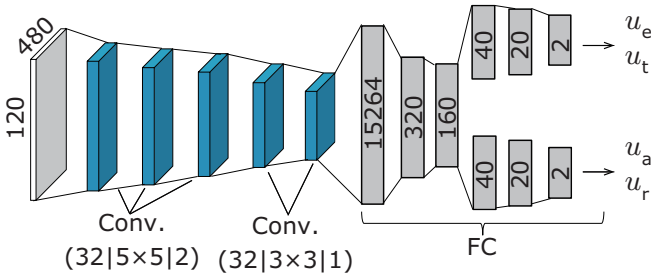


Fig. 5. Architecture of autopilot based on CNN

For segmentation images of size 480×120 pixels, cropped from the cockpit viewpoint and denoted as O^c, five convolutional operations are applied. Image features essential

for determining the control commands are extracted through these operations. The resulting feature maps are flattened and fed into 15,264 input neurons of the fully connected layers. Through these layers, the control commands U are output. In U, the elevator deflection and throttle, u_{e} and u_{t}, correspond to the vertical direction, while the aileron and rudder deflections, u_{a} and u_{r}, correspond to the lateral direction.

During forward propagation, ReLU is used as the activation function in both the convolutional and fully connected layers. The loss function ℓ, introduced in Eq. (3) and described in Section III, is calculated as the mean squared error:

$$\ell(U_i^*, U_i) = \frac{1}{N}\sum_{i=1}^{N}\left(U_i^* - U_i\right)^2, \tag{4}$$

where N is the size of the training dataset. As shown in Fig. 2, U_i^* denotes the instructed optimal outputs recorded in Eq. (1), while U_i denotes the outputs determined by the autopilot according to its policy during training. Adam [11] is used to optimize the parameters w in Eq. (2).

V. SIMULATION EXPERIMENTS ON LANDING FLIGHTS

A. Settings for Experiments

In this experiment, the effectiveness of the proposed autopilot for landing flights is evaluated. The flight simulator X-Plane [12], developed by Laminar Research, is used for the evaluation. For comparison, our previous autopilot using RGB image inputs is also applied to the aircraft. The proposed and previous autopilots are denoted as π_{new} and π_{old}, respectively. **Fig. 6** shows the aircraft used in the simulation, the Cessna 172.

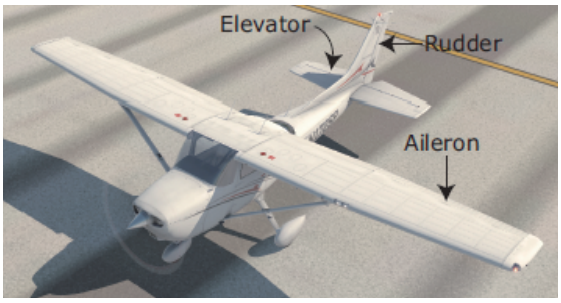
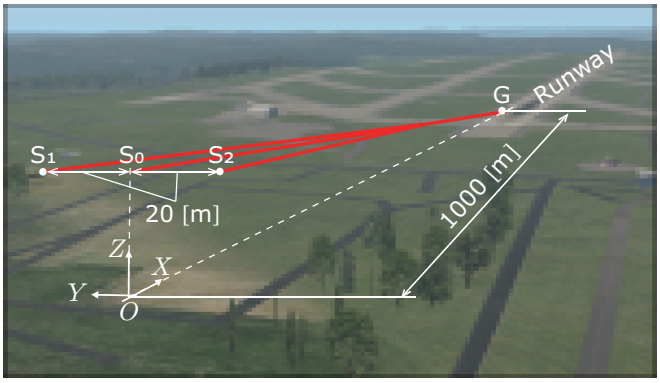


Fig. 6. Cessna 172 light aircraft used in landing simulation

As with the airframe illustrated in Fig. 1, the Cessna 172 is equipped with three types of control surfaces: ailerons, rudder, and elevator. Accordingly, the aircraft's attitudes around the lateral, longitudinal, and vertical axes, namely the pitch, roll, and yaw angles θ, ϕ, and ψ, are influenced by both the control surfaces and the throttle. **Fig. 7** shows the target environment in the flight simulator. The aircraft shown in Fig. 6 attempts to land on the runway ahead using the autopilots π_{new} and π_{old}. In this environment, the X, Y, and Z axes correspond to the longitudinal, lateral, and vertical directions, respectively.



(a) Seattle Tacoma International Airport

(b) Runway

Fig. 7. Flight environment settings in X-Plane simulator

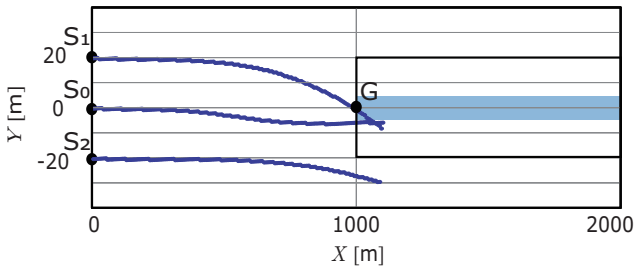
In Fig. 7(a), the origin of the Cartesian coordinate system is set at a point 1,000 meters behind the aiming point G along the X-axis, where G is located at the runway threshold. Three red lines from S_0, S_1, and S_2 toward G on the runway indicate the trajectories of the aircraft during the instructed landing flights. Fig. 7(b) shows a view from the cockpit just before landing. The width of the runway is 40 [m]. To avoid collisions with approach lights installed at the runway edges, the aircraft is required to land within ± 5 [m] of the centerline, which is indicated by the blue area. Accordingly, the human pilot instructed the aircraft to land within this range. These landing flight instructions were provided in a daytime environment at 1 p.m., and the training dataset obtained during this phase was used to train π_{new} and π_{old} through behavior cloning.

We compare the flight performance of the aircraft using π_{new} and π_{old} not only in the instructed (known) environment but also in unknown environments. For this purpose, the aircraft attempts to land on the runway in unknown evening and nighttime environments at 6 p.m. and 10 p.m., which differ from the daytime lighting conditions shown in Fig. 7(a). In each environment, the aircraft with the trained π_{new} and π_{old} begins autonomous landing flights from three initial positions, S_0 to S_2.

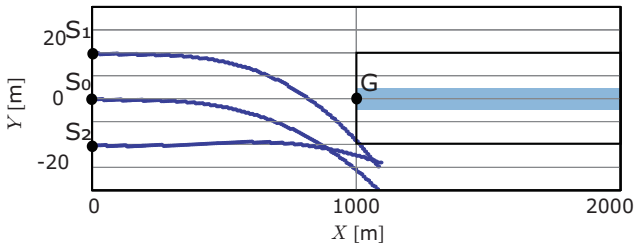
B. Flight Results

In the experiments, a landing flight was regarded as successful if the aircraft landed within the blue area on the runway, as shown in Fig. 7(b). In the daytime environment, the aircraft successfully completed the landing flights from all initial positions, S_0 to S_2, regardless of whether π_{new} or π_{old} was used. These results indicate that segmentation images were equally effective to RGB images for reproducing

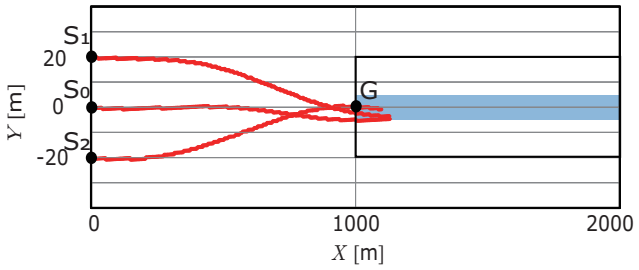
the instructed flights. **Fig. 8** shows the flight trajectories of the aircraft using the two autopilots during landings in the unknown evening and nighttime environments.



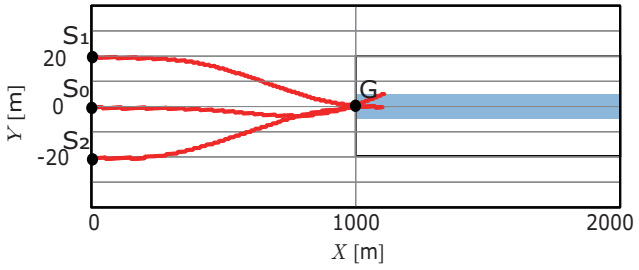
(a) π_{old} in evening environment



(b) π_{old} in nighttime environment



(c) π_{new} in evening environment



(d) π_{new} in nighttime environment

Fig. 8. Comparison of flight trajectories of aircraft in X-Y plane

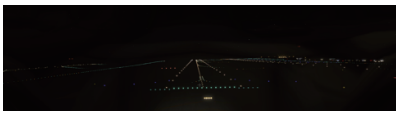
In Fig. 8(a), although the aircraft using π_{old} landed on the runway from S_0 and S_1 in the evening environment, the final landing points were 5.5 [m] and 8.1 [m] away from the centerline, respectively. Furthermore, from S_2, the aircraft landed outside the runway. In Fig. 8(b), the situation deteriorated, as the aircraft deviated even farther from the runway, veering to the right. This deviation was due to the inherent turning tendency of a propeller-driven aircraft. As a result, the aircraft failed in all landing attempts. To further examine these results, we analyze the feature maps extracted from RGB images within the autopilot, as shown in **Fig. 9**.

(a) Image input in daytime environment

(b) Feature map of Fig. 9(a)

(c) Image input in evening environment

(d) Feature map of Fig. 9(c)



(e) Image input in nighttime environment

(f) Feature map of Fig. 9(e)

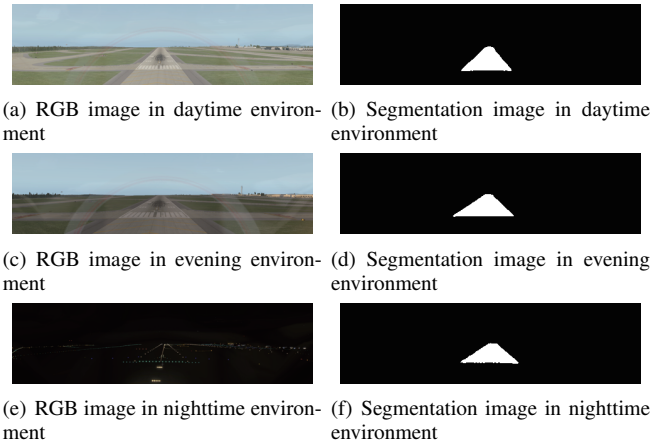
Fig. 9. Comparison of feature maps extracted by π_{old} from RGB image inputs in different environments

For the RGB image in the daytime environment (see Fig. 9(a)), Fig. 9(b) shows that features of the runway and other airport buildings were successfully extracted. This feature map indicates that the autopilot π_{old} was trained to determine control commands by focusing on the position and shape of these landmarks. However, for the RGB images in the evening and nighttime environments (see Fig. 9(c) and Fig. 9(e)), features were extracted from all pixels below the horizon and across the entire image, as shown in Fig. 9(d) and Fig. 9(f). Consequently, the autopilot failed to focus on the runway and buildings, and it was unable to determine appropriate control commands, resulting in unsuccessful landings.

In contrast to the above results, the aircraft using the proposed autopilot π_{new} successfully completed all landing flights in the evening and nighttime environments, as shown in Fig. 8(c) and Fig. 8(d), regardless of the initial positions S_0 to S_2. To investigate the reasons for this improved performance, we first present the segmentation images generated by U-Net from the RGB images captured in each environment, as shown in **Fig. 10**.

For the visually different RGB images captured in the daytime, evening, and nighttime environments, as shown in Fig. 10(a), Fig. 10(c), and Fig. 10(e), the segmentation images correctly classified only the runway despite the changes in lighting conditions. As a result, the segmented runway, shown in white in Fig. 10(b), Fig. 10(d), and Fig. 10(f), appeared visually similar across all environments. **Fig. 11** shows the feature maps extracted from these segmentation images at the final convolutional layer of the autopilot.

Compared to the RGB image in Fig. 10, the runway appears in the same position and with the same shape in the segmentation images, even under different environmental conditions, as shown in Fig. 11(a), Fig. 11(c), and Fig. 11(e). For these segmentation image inputs, runway features were extracted not only in the daytime environment, as in Fig.

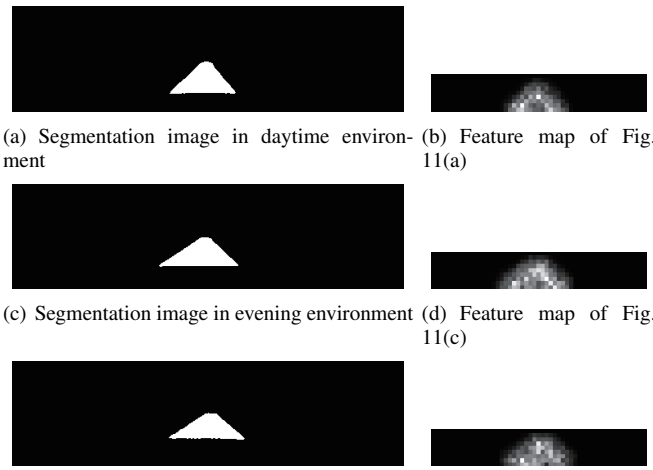


(a) RGB image in daytime environment (b) Segmentation image in daytime environment

(c) RGB image in evening environment (d) Segmentation image in evening environment

(e) RGB image in nighttime environment (f) Segmentation image in nighttime environment

Fig. 10. Segmentation images generated by U-Net from RGB images



(a) Segmentation image in daytime environment (b) Feature map of Fig. 11(a)

(c) Segmentation image in evening environment (d) Feature map of Fig. 11(c)

(e) Segmentation image in nighttime environment (f) Feature map of Fig. 11(e)

Fig. 11. Comparison of feature maps extracted by π_{new} from segmentation images

11(b), but also in the evening and nighttime environments, as in Fig. 11(d) and Fig. 11(f), within the autopilot. Consequently, even in unknown environments, the autopilot was able to determine control commands in the same manner as in the daytime environment, without being affected by changes in lighting conditions, by consistently focusing on the runway features.

To summarize, in the instructed environment, both the previous and the proposed autopilots, using RGB images and segmentation images as inputs respectively, enabled the aircraft to achieve comparable flight performance. However, in the unknown environments, the previous autopilot failed to determine appropriate control command outputs due to visually different image inputs resulting from changes in lighting conditions. In contrast, the proposed autopilot received visually consistent segmentation images in which only the runway was correctly classified, even under varying lighting conditions. This allowed it to extract similar runway features and determine control commands in the same manner as in the instructed environment. Consequently, the aircraft using the proposed autopilot demonstrated improved generalization capability to unknown evening and nighttime environments and successfully landed in all flight experiments.

VI. CONCLUSIONS

In this paper, we proposed an aircraft autopilot specifically designed for autonomous landing flights. However, cockpit-view images used as inputs to the autopilot vary with lighting conditions. To improve the generalization capability of the proposed autopilot to environments with different lighting, we used segmentation images in which only the runway, represented in white, is used as input. Simulation experiments demonstrated that the aircraft equipped with the proposed autopilot achieved sufficient generalization to unknown evening and nighttime environments, successfully landing on the runway. In future work, we will consider additional factors such as weather variations and different airports, and evaluate the effectiveness of the autopilot for landing flights in these broader unknown environments.

REFERENCES

[1] K. Hancock, "The Airline Pilot Shortage: A Result of Age Discrimination or Excessive Training Requirements?," Journal of Air Law and Commerce, Vol. 88, No. 2, pp. 535–572, 2023.

[2] M. Bojarski *et al.*, "End to End Learning for Self-Driving Cars," *arXiv preprint*, arXiv:1604.07316v1, pp. 1–9, 2016.

[3] H. M. Eraqi *et al.*, "End-to-End Deep Learning for Steering Autonomous Vehicles Considering Temporal Dependencies," *Neural Information Processing Systems*, pp. 1–8, 2017.

[4] L. Liangzhi *et al.*, "Human-Like Driving: Empirical Decision-Making System for Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, pp. 1–10, 2018.

[5] B. D. Argall *et al.*, "A Survey of Robot Learning from Demonstration," *Robotics and Autonomous Systems*, Vol. 57, pp. 469–483, 2009.

[6] K. Ujiie *et al.*, "Behavior Cloning for Aircraft Autopilot in Approaching and Flaring Scenarios," *SICE Conference on System Integration*, pp. 3248–3252, 2022 (in Japanese).

[7] I. Tanaka and S. Hoshino, "Aircraft Autopilot Composed of Scenario Classifier and Flight Controllers based on CNNs for Landing Flights," *Journal of Robotics and Mechatronics*, 2025 (accepted).

[8] Y. Lecun *et al.*, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[9] S. Minaee *et al.*, "Image Segmentation Using Deep Learning: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 44, No. 7, pp. 3523–3542, 2022.

[10] O. Ronneberger *et al.*, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.

[11] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference for Learning Representations*, pp. 1–15, 2014.

[12] L. R. Ribeiro and N. M. F. Oliveira, "UAV Autopilot Controllers Test Platform Using Matlab/Simulink and X-Plane," *IEEE Frontiers in Education Conference*, pp. S2H-1-S2H-6, 2010.

[13] M. V. Cook, "Flight Dynamics Principles," *Butterworth-Heinemann*, 1997.