Faster Depth Estimation for Autonomous Flying Drones

Takuya Kosaka[1,*], Xiangbo Kong[2,†], Tomoyasu Shimada[1,†], Hiroki Nishikawa[3,†] and Hiroyuki Tomiyama[1,†]

[1]Graduate School of Science and Engineering, Ritsumeikan University
[2]Department of Intelligent Robotics, Factory of Engineering, Toyama Prefectural University
[3]Graduate School of Information Science and Technology, Osaka University

Abstract

In this study, we propose a fast depth estimation generator in deep learning to improve the collision avoidance performance of autonomous flying drones. Since drones move at high speeds, a wide range of depth images is required for collision avoidance. However, it is not easy to equip drones with small, lightweight, and high-performance depth sensors. To address this issue, it has become possible to perform collision avoidance using only a monocular camera at a low cost by converting visible light images into depth images using deep learning. However, in cases where resources are limited, such as onboard computers on drones, the processing speed of the deep learning model used for estimating depth images may not be sufficient, resulting in inadequate collision avoidance. Therefore, this paper aims to enhance collision avoidance for autonomous flying drones by developing a high-speed neural network model for depth estimation in deep learning. The experimental result shows that processing speeds were reduced by up to 20%. In addition, collision rates improved in many environments.

Keywords

depth estimation, AirSim, Pix2Pix

1. Introduction

Drones have become increasingly popular in recent years and are expected to be used in a variety of applications, including logistics, security, aerial photography, surveying, equipment inspection, agriculture, and saving lives during disasters. For example, an algorithm for using drones to help detect disasters is considered[1].In other cases, agricultural drones are used to increase the efficiency of crop spraying[2]. Some autonomous drones use the information obtained from onboard cameras to avoid obstacles. However, when many sensors, high-performance devices, and devices with large weights and footprints are installed, energy consumption increases. In addition, when energy consumption increases, long-distance flight becomes impossible with only the limited energy of the battery. Infrared cameras are particularly popular as sensors for

ATAIT 2023: The 5th International Symposium on Advanced Technologies and Applications in the Internet of Things, August 28–29, 2023, Kusatsu, Shiga, Japan

*Corresponding author.
†These authors contributed equally.
ri0100kp@ed.ritsumei.ac.jp (T. Kosaka); kong@pu-toyama.ac.jp (X. Kong); nishikawa.hiroki@ist.osaka-u.ac.jp (H. Nishikawa); ht@fc.ritsumei.ac.jp (H. Tomiyama)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3459/paper11.pdf

collision avoidance. Infrared camera called Time of Flight (ToF) camera can obtain depth maps by using near infrared light reflection and measuring distance. However, high-performance infrared cameras that can be mounted on drones and can avoid collisions are expensive. To address this issue, depth estimation methods are proposed[3][4]. Therefore, drones often employ deep learning-based depth estimation to obtain depth images from a monocular camera. Monocular cameras are less expensive and smaller than infrared cameras. Using monocular cameras to obtain depth images solves energy consumption increasing. However, there is a problem with this method, as the estimation of the image using deep learning requires significant processing time.

Based on the above, this paper proposes a fast and processable deep learning generator for autonomous flying drones. In this study, we use Pix2Pix as a model for deep learning, which converts color images into depth images, and we use AirSim, a drone simulator, in our collision avoidance experiments.

The rest of this paper is organized as follows: Section 2 shows the related research of collision avoidance of drones, and generation of depth estimation, which is a deep learning method. Section 3 describes a proposal method. Section 4 shows the experimental. Section 5 concludes this paper and discusses future issues.

2. Related Work

In paper [5], the authors gave an overview of the simulator, AirSim, and conducted a comparison experiment between the flight characteristics in the real world and in the simulation. The results of the experiment show that the flight characteristics of each quadcopter in the simulation are qualitatively close. Therefore, we believe that AirSim can reproduce the flight in the real world. In paper [6], the authors described the difference between the real world and the virtual environment, and stated that reinforcement learning can be performed on AirSim and can be performed in the real world. Similarly, in paper [7], AirSim operates in real time, and there is no delay in the data and images of the drone for deep learning, and stated that it is suitable as a virtual environment in deep learning for autonomous flying drones.

There has been a lot of work associated with autonomous drones. Collision avoidance algorithms are especially critical for autonomous drone flight. In paper [8], the authors conducted an experiment on collision avoidance with AirSim The method is to create a section of depth images acquired from depth sensors divided into five sections in the transverse direction, in which obstacles close to the drone fly. In paper[9], the authors described collision avoidance using depth images based on collision avoidance experiments such as in paper [8], taking into account the up-down movement of the drone, as well as the method shown in Fig. 1 as an algorithm for specifying the direction. First, depth images are acquired using the depth sensor of the drone. Next, the acquired depth images are divided into overlapping vertical and horizontal sections of 17 × 17, and the upper left section is numbered (0, 0) and the lower right section is numbered (16, 16). Then, the sum of the pixel values of each section is calculated to determine the maximum section, and the center section is set to be the maximum section when the average of the pixel values of the center section is 200 or more, so that the center section proceeds in the center and there is no meandering. Next, we describe an algorithm for controlling the speed

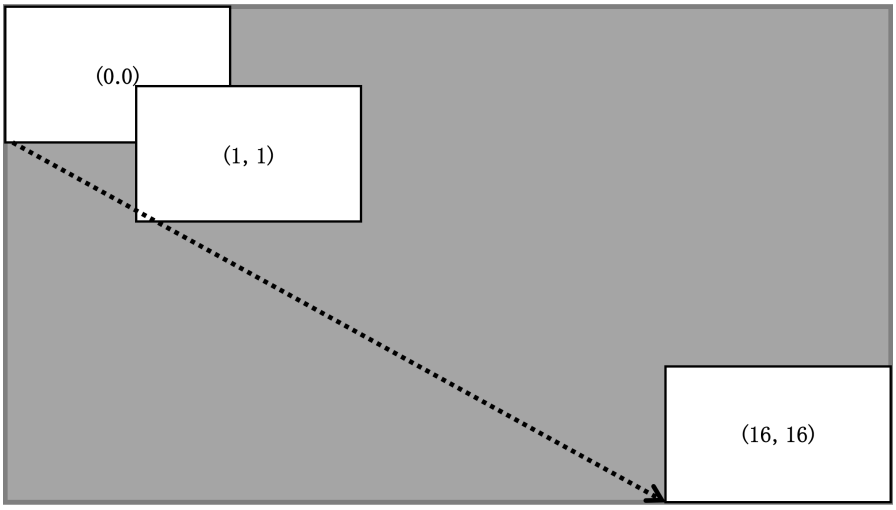

Figure 1: Sectioning of the depth image [9]

of the drone so that the maximum section is centered in the next processing. PID control is a type of feedback control, derived from the acronym Proportional, Integral, and Differential. Feedback control is a method of controlling the output so that it reaches an appropriate target value by sending the previous output back to the input side. The content of PID control is the deviation between the target value to be followed and the previous control amount, and the result of the calculation of the proportional, integral, and differential operations is sent back to the next input to determine the next operation amount for the target value, which enables speed control in the collision avoidance direction of the drone.

In this work, we use AirSim to evaluate flight performance, which is the virtual environment for autonomous vehicles and drones.

In paper [10][11] using Conditional Generative Adversarial Network (cGAN) to generate paired images, which is called Pix2Pix[12]. On AirSim, depth images in a range of more than 200 m can be obtained, but cameras with inexpensive depth sensors, which can be mounted on drones in the real world, can measure only short ranges, such as 10 m to 20 m. Therefore, the above study can be used to generate long-range depth images from monocular images

3. Generator model based on REF-Net

In this section, we present the model that fast processing speed for estimating depth images. REF-Net[13] is Derived from Robust, Efficient, and Fast network. In this study, we created a model for drone depth estimation based on REF-Net and named it Fast REF-Net. As shown in Figure 2, one of the characteristics of REF-Net is that the structure of the generator model is asymmetric. As shown in Figure 3, each block is branched from the input and connected before the output, similar to a skip connection. This structure is called a residual skip connection. Skipped layers can be connected to each other by a detour route, and feature maps can be

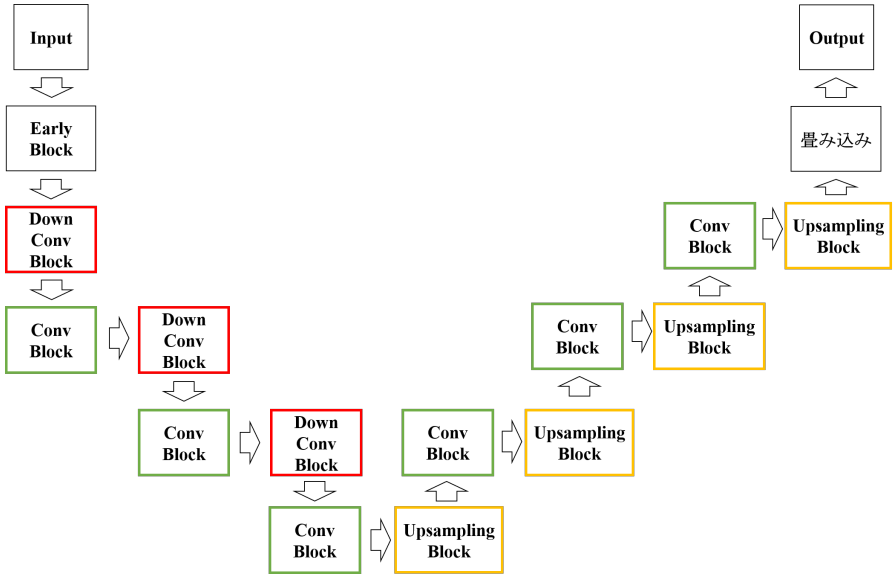

Figure 2: An overview of the proposed methodology

propagated to the next and subsequent layers. Therefore, it allows the feature map of the input to be retained, making it possible to recover the feature values pixel by pixel. We use nearest neighbor interpolation as an upsampling method to restore the feature map to its original input size. Nearest neighbor interpolation is a simple and non-parametric interpolation technique used in computer graphics and image processing. In this method, when we need to increase the size of the feature map, each pixel in the original feature map is replicated to fill the new, larger space. The value of the new pixels is set to be the same as the value of the nearest pixel in the original feature map. Since this interpolation technique does not involve complex mathematical computations or parameter learning, it is computationally efficient and has the advantage of faster processing speed compared to other interpolation methods. While this simple algorithm may cause data degradation, it has the advantage of faster processing speed compared to other interpolation methods. Also, unlike the data expansion in the inverse convolution layer used in many generator decoders, there are no trainable parameters, so the memory and estimation time requirements can be reduced. Changes from REF-Net include a reduction in the number of blocks passed through to increase processing speed. The model structure was deepened one step to increase the accuracy of the generated images. In addition, the convolution, normalization, and activation layers for each block were reduced to improve processing speed.

4. Experiments

In this section, we evaluate our method in terms of accuracy and performance to avoid collisions. We use Intel Core i7-9700K (64 GB of main memory) and NVIDIA GeForce RTX 2070 SUPER. Dataset, which is used for training, validation, and testing, is collected from four maps provided in the AirSim environment; City Environment, Coastline, Neighborhood, and Blocks.

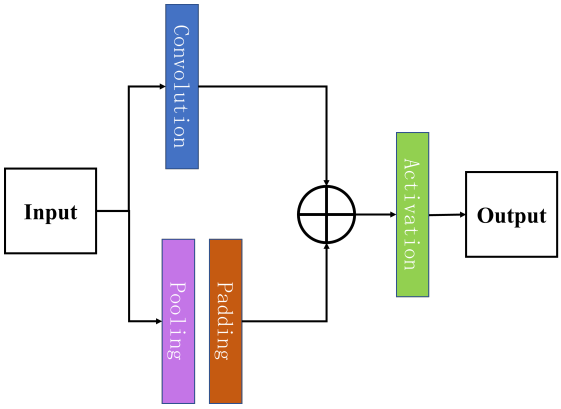

Figure 3: Early Blocks

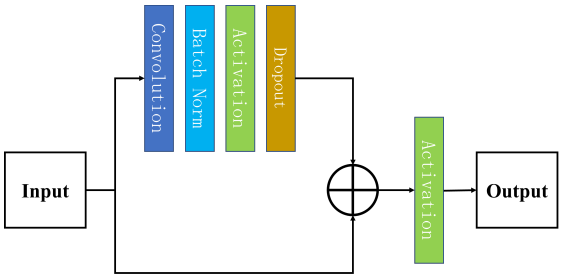

Figure 4: Conv Blocks

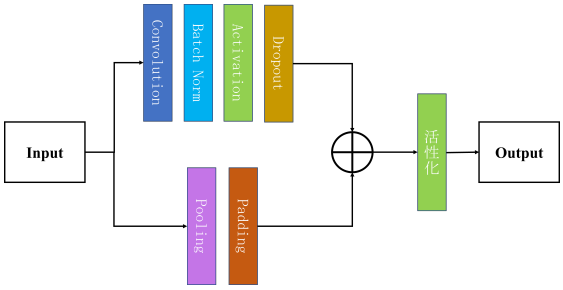

Figure 5: Down Conv Blocks

4.1. Comparison of processing speed and accuracy of each model

In this section, we evaluate Fast REF-Net generator models. The paired images used for learning were RGB image and depth image pairs obtained from the 4 maps published by AirSim: Blocks, CityEnviron, AirSimNH, and Coastline. 2000 of each map were studied out of a total of 8000 images.

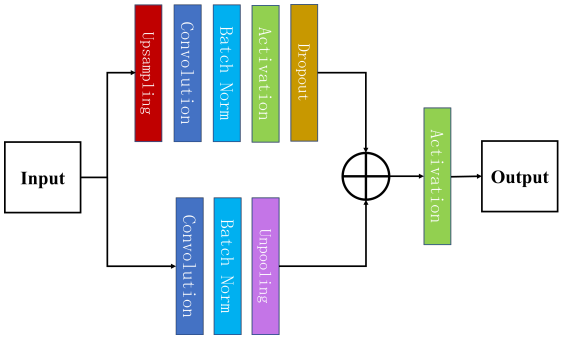

Figure 6: Upsampling Blocks

Table 1
Comparison of processing speed

Device	Average process time(s)		
	U-Net	REF-Net	Proposed
PC	0.331	0.322	0.250
Jetson Nano	0.014	Error	0.035
Jetson Xavier NX	0.012	0.433	0.021

4.1.1. Comparison of processing speed

Next, we conduct an experiment to measure the processing speed of each model. In the experimental environment, we opted to use an AMD Ryzen 5 HS with 8 GB of RAM PC and Jetson Nano and Jetson Xavier NX for measuring processing speed due to resource limitations on the drone platform. Autonomous flight drones typically have limited computing resources, and while high-performance hardware like Intel Core i7 and GeForce RTX could potentially yield more precise results, may result in minimal differences in processing time among the various depth estimation models. As such, it becomes challenging to discern significant discrepancies in processing speed, and any differences observed could be within the margin of error. Therefore, the use of the AMD Ryzen 5 HS for measuring processing speed provides a clearer distinction between the generators and helps to validate the effectiveness of our approach in real-world drone applications with limited computing resources. As a measurement method, we measure the inference time of each model when 1000 images of 3 ×256 ×256 are input, and calculate the average time. The processing speed of the generator in each experimental environment is shown in Table 1.

4.1.2. Error and Accuracy of Generated Images

We describe the evaluation of the output image of each model. For the evaluation method, we input the color image obtained by AirSim into each model, and use the output image and the depth image relative to the input image obtained by AirSim. We use the root mean square error (RMSE) and the relative error (Rel.) to calculate the error, and the threshold density (trial) to

Table 2
Error and Accuracy of Generated Images

Generator	Error(↓)		Accuracy(↑)		
	RMSE	Rel.	δ1	δ2	δ3d
U-Net	9.354	0.232	0.776	0.897	0.943
REF-Net	12.856	0.365	0.740	0.862	0.914
Proposed	15.467	0.514	0.719	0.843	0.895

evaluate the accuracy between the two images.

Equation (1) shows RMSE.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{1}$$

Equation (2) shows Rel..

$$Rel. = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{\hat{y}_i} \tag{2}$$

Equation (3) shows threshold accuracy.

$$\delta_n = \frac{Card(\{y_i : max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i} < 1.25^n\})}{Card(\{y_i\})} \quad \text{(n=1, 2, 3)} \tag{3}$$

$\hat{y}_i$ and y_i are the ground truth and predicted depths respectively of pixels, and N is the total number of pixels in all the evaluated images. The error and accuracy of generated images are shown in Table 2. Figure 7 shows the input images, the ground truth images, and the output image from each model.

4.2. Safe Flight Evaluation in AirSim Environment

This section describes the collision avoidance experiment on AirSim.

According to Table 3, when depth estimation was performed using each generator, there was an improvement in the collision rate in some environments. This is thought to be because the processing speed is slower and collision avoidance is delayed than when depth images were acquired using AirSim alone, causing the drone to move further away from obstacles. In addition, when compared to U-Net, there is an improvement in the collision rate in some maps for our proposal method. This is thought to be due to the accuracy of the generated image, which was able to avoid obstacles at a higher processing speed, but decreased when the speed was increased.

5. Conclusion

In this paper, we implemented a high-speed generator for drone collision avoidance. Based on previous research [9][12], we implemented a program to acquire color images on AirSim,

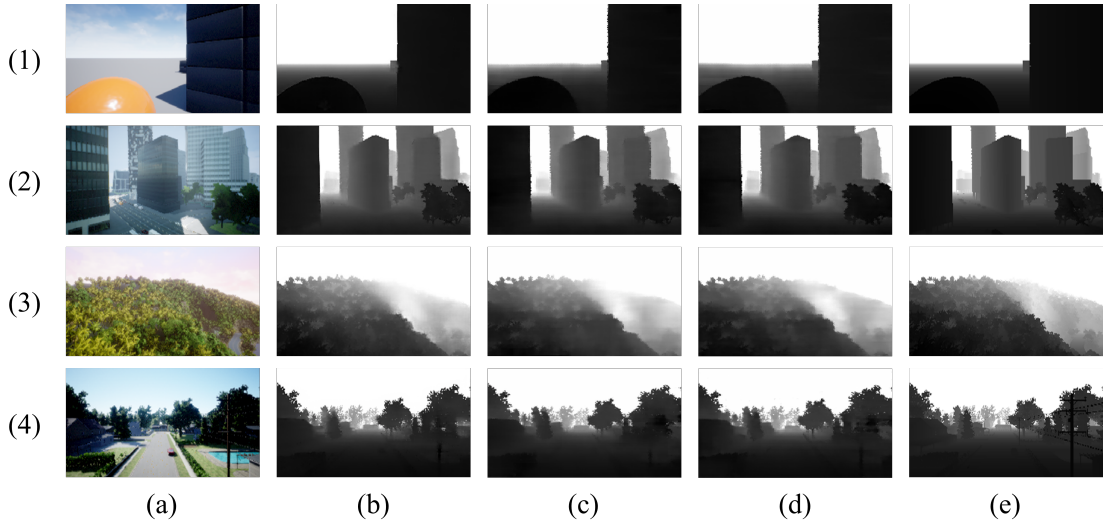

Figure 7: Inputs and outputs each method :(a)monocular image, (b)U-Net, (c)REF-Net ,(d)Proposed, (e)ground truth, (1)Blocks, (2)CityEnviron, (3)Coastline, (4)AirSimNH

Table 3
Collision Rate

Generator	Collision Rate(%)			
	AirSim	U-Net	REF-Net	Proposed
Blocks	7.50	6.00	9.00	8.50
CityEnviron	20.75	19.00	23.00	19.50
AirSimNH	3.50	1.00	4.00	0.00
Coastline	0.00	1.50	2.00	0.25

estimate depth images using U-Net as the generator, and perform collision avoidance from the generated depth images. Next, we implemented REF-Net as the generator so that the created program could operate at high speed. Since the implemented REF-Net did not show any improvement in processing speed compared to U-Net, FAST REF-Net was implemented and improved to run at higher speed. Next, we compared the implemented high-speed generator models with U-Net.As an experiment, we measured and compared the error, accuracy, and processing speed of the images generated by each model, and found the improvement in processing speed while the error and accuracy were worse than other methods. Finally, we performed experiments on collision avoidance using sensors on the AirSim and collision avoidance using each model, and the experimental results showed improvements in collision rates on some maps than those using U-Net. Future research will focus on creating generator models with better processing speed and accuracy, or improving collision avoidance algorithms.

Acknowledgments

This work is partly commissioned by NEDO (Project Number JPNP22006).

References

[1] A. V. Savkin, H. Huang, Navigation of a network of aerial drones for monitoring a frontier of a moving environmental disaster area, IEEE Systems Journal 14 (2020) 4746–4749. doi:10.1109/JSYST.2020.2966779.

[2] S. D. Panjaitan, Y. S. K. Dewi, M. I. Hendri, R. A. Wicaksono, H. Priyatman, A drone technology implementation approach to conventional paddy fields application, IEEE Access 10 (2022) 120650–120658. doi:10.1109/ACCESS.2022.3221188.

[3] F. Liu, C. Shen, G. Lin, I. Reid, Learning depth from single monocular images using deep convolutional neural fields, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (2016).

[4] X. Li, M. Ding, D. Wei, X. Wu, Y. Cao, Estimate depth information from monocular infrared images based on deep learning, in: International Conference on Progress in Informatics and Computing, 2020.

[5] S. Shah, D. Dey, C. Lovett, A. Kapoor, Airsim: High-fidelity visual and physical simulation for autonomous vehicles, in: Field and Service Robotics Conference, 2017.

[6] C. Y. Ho, S. Y. Tseng, C. F. Lai, M. S. Wang, C. J. Chen, A parameter sharing method for reinforcement learning model between airsim and uavs, in: International Cognitive Cities Conference, 2018.

[7] S. Wang, J. Chen, Z. Zhang, G. Wang, Y. Tan, Y. Zheng, Construction of a virtual reality platform for uav deep learning, in: Chinese Automation Congress, 2017.

[8] C. Ma, Y. Zhou, Z. Li, A new simulation environment based on airsim, ros, and px4 for quadcopter aircrafts, in: International Conference on Control, Automation and Robotics„ 2020.

[9] T. Shimada, H. Nishikawa, X. Kong, H. Tomiyama, A dynamic path planning method for multirotor using depth images in airsim, in: International Workshop on Nonlinear Circuits, Communications and Signal Processing, 2021.

[10] T. Shimada, H. Nishikawa, X. Kong, H. Tomiyama, Pix2pix-based depth estimation from monocular images for dynamic path planning of multirotor on airsim, in: International Symposium on Advanced Technologies and Applications in the Internet of Things, 2021.

[11] T. Shimada, H. Nishikawa, X. Kong, H. Tomiyama, Pix2pix-based monocular depth estimation for drones with optical flow on airsim, Sensors 22 6 (2022).

[12] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[13] O. Bekhzod, K. Jeonghong, P. Anand, Ref-net: Robust, efficient, and fast network for semantic segmentation applications using devices with limited computational resources, IEEE Access 9 (2021) 15084–15098. doi:10.1109/ACCESS.2021.3052791.