ICIC Express Letters
Part B: Applications
Volume **10**, Number **5**, May **2019**

ICIC International ©2019 ISSN 2185-2766
pp. **395–403**

DETECTION OF OBJECTS JUMPING IN FRONT OF CAR USING DEEP LEARNING

Mitsutaka Nakano[1], Taiki Kuwazawa[2] and Yoshifumi Oyama[1]

[1]Department of Human-Oriented Information Systems Engineering
National Institute of Technology, Kumamoto Collage
2659-2 Suya, Koshi City, Kumamoto 861-1102, Japan
{ nakano; oyama }@kumamoto-nct.ac.jp

[2]Mitsubishi Electric Engineering Company Limited
1-13-5 Kudankita, Chiyoda-ku, Tokyo 102-0073, Japan

Received October 2018; accepted January 2019

Abstract. *In recent years, the number of cars having driver assistance systems is growing. On the other hand, those systems require specialized equipment. This makes implementation of the systems into existing cars difficult. In our laboratory, we have developed the system to detect objects jump in front of car using monocular camera to address this problem. This research has used the Optical Flow, which causes some flaws. To solve the problems, we explore to use the Deep Learning to detect the objects. Applying the Deep Learning is expected to improve the problems, especially improving detection errors. As a result, a trained neural network is managed to react with objects moving from edge to center of frames. The neural network marked better detection error than the system using the Optical Flow.*

Keywords: Deep neural network, Deep Learning, Driver assistance system, Video processing, Image processing

1. **Introduction.** Accidents with wild animals are on a moderately increasing trend. Collision accidents with wild animals have not only impacts on the ecosystem but also secondary damage by subsequent vehicles. Also, collision accidents with large animals can directly cause human damage. In recent years, many automobiles are equipped with an automatic brake system as a driver assistance system. In fact, in Japan, automobile equipped vehicles in new cars are expected to exceed 90% in 2020, including optional equipment [1]. However, driver assistance systems installed in new cars use special equipment such as radar and multi-lens camera, and it is difficult to incorporate them into existing cars, such as requiring delicate adjustment [2]. Also, the widespread use of drive recorders and smartphones has made cameras-equipped devices familiar. Therefore, in our laboratory we studied to detect objects jump in front of car using only monocular cameras [3]. Monocular cameras are easy to procure equipment and install and adjust to existing cars. In the previous research, we constructed a system to detect objects jump in front of car using Optical Flow. Optical Flow is a vector representing movement of a pixel caused by movement of an object or a camera between two images. The optical flow calculation function detects feature points between two frames and combines two similar feature points [4]. Next, the Optical Flow is calculated by calculating the vector between the combined two points. Therefore, when the feature points in the video increase or decrease between two frames, there are cases where two similar but different two points are combined as shown on the left side of Figure 1. In such a case, the motion of an object which is not actually calculated is calculated, and a problem of erroneous recognition occurs. In recent years, Deep Learning has become widely used in the field

DOI: 10.24507/icicelb.10.05.395

FIGURE 1. Failure example of Optical Flow

of image processing. Deep Learning has an advantage that it can deal with the problem by learning parameters of the neural network. We also propose a method using Deep Learning in our laboratory [5]. Therefore, in this research, we solve the problems that occurred in the previous research by constructing a system to detect objects that jump in front of car using the Deep Learning and making it learn.

2. **The Composition of Deep Learning Used in This Research.** Deep Learning is a machine learning using a network called a Deep Neural Network (DNN) in which neural networks are layered in multiple layers. Layers constituting the neural network include all connection layers, convolution layers, recursive layers, and the like. In particular, a convolutional layer is widely used for image processing. In this paper, we want to recognize not only whether there is an object that jumps in front of car but also where the object is. Therefore, we decided to use a neural network of FCN (Fully Convolutional Network) [6]. FCN is a new type of network in which all layers are composed of only a convolution layer. FCN extracts various features by increasing the number of channels in the convolution layer. The input data is processed by repeating data compression at the pooling layer. The processed data is expanded to the same size as the input at the deconvolutional layer and output through the convolutional layer. In this way, the feature of FCN is to obtain the same size output as the input. The FCN can handle not only something but also where the object is. Therefore, it is expected to be utilized particularly for object recognition. In this research, we aim to make use of FCN not only to find the object that jumps in front of car but also to recognize where the object is.

2.1. **Preparation of teacher data.** For the Deep Learning, it is necessary to prepare learning data sets and corresponding teacher data and make them learn. The data used for learning of machine learning can be expected to be effective even in a practical environment if the number of variations and data is large. In this time, we created a data set to be teacher data by combining objects with the video of the drive recorder. The braking distance of a passenger car at 60km/h is 37m. When the viewing angle of the camera is 120 degrees, the visual field of 37m ahead is 128m. At this time, if the resolution in the horizontal direction of the image is 1,024 pixels, an object with a size of 1m ahead of 37m can be set to 8 pixels. Therefore, we determined 8 pixels to be the minimum size of the projecting object. Also, assuming that the aspect ratio is 16 : 9, the size of the image to be processed is 1,024 $\times$576. We used 15 in-vehicle camera images obtained from YouTube-8M for video data used [7]. These images were subdivided into 100 frame units. Among them, we used 1,186 videos, except for scene transitions, excluding those in which the video moves extremely. To these videos, we randomly combined 21 images that were distributed as free to use. In the compositing process, change in hue and rotation processing were randomly applied to the image to be synthesized. Then, the direction in which it jumps, the place where it jumps, the speed of movement and enlargement, etc., are decided at random. We synthesized them and generated various video data. Also, by

(a) A part of the teacher data (b) A part of the correct answer data

FIGURE 2. The generated data set

adjusting the brightness of the object according to the brightness of the image, we devised it so that it looks as natural as possible. Correct answer data was generated as a binary image in which the projected object portion was white, with the same size and length as the image data. Figure 2 shows an example of the generated image and teacher data. In Figure 2(a), an image of a police car at the center of the screen is synthesized. Figure 2(b) is the correct answer data of the frame corresponding to (a). Among the synthesized police cars, it can be seen that only the right side, which is the direction which jumped out, is a binary image which turned white.

2.2. **Network configuration.** The image of the data set is 1,024 ×576 pixels. However, when using this for input as it is, memory of GPU (Graphic Processing Unit) which speeds up learning is insufficient and it was not possible to construct a sufficiently deep network. For this reason, the input data was transformed into 512×64 pixels divided vertically into two and horizontally divided into nine. In addition, to detect an object that jumps out, it is necessary to process consecutive frames as images, rather than processing each frame as an image. In this time, the previous and next frames are combined and 6-channel data of 512×64 pixels is input to the neural network, so it is used for video processing. The entered data passes through 10 layers of convolutional layer and 5 layers of pooling layer. Furthermore, it is returned to the same size as the original data with 5 layers of deconvolutional layer. Then, the last convolutional layer outputs 2 channels of 512×64 pixels. The outputs of these two channels represent the probability that one of them has no object and the other one represents the probability that there is an object. Figure 3 shows the configuration of the network. In addition to the layer shown in Figure 3, layers of batch normalization, layers of Gaussian noise, and layers of dropout are included in the respective convolutional layers and deconvolutional layers. The batch normalization layer is a technique for speeding up the learning of the network [8].

2.3. **Network learning.** In order to support learning, the input data to the network is subtracted from the average value for one batch of data and divided by the maximum value. As a result, the input data is normalized. The average value is 0 and the data width is 1. In learning of the network, as shown in Figure 2(b), the proportion occupied by the jumping out object in the frame is small. Therefore, if you learn directly, the degree of influence on learning of the projected object becomes small. Therefore, the ratio of the absence of the object to the learning for detecting the object is multiplied by the ratio of the existence of the object to the learning which does not detect the object as a coefficient. As a result, the influence of each data on learning is normalized.

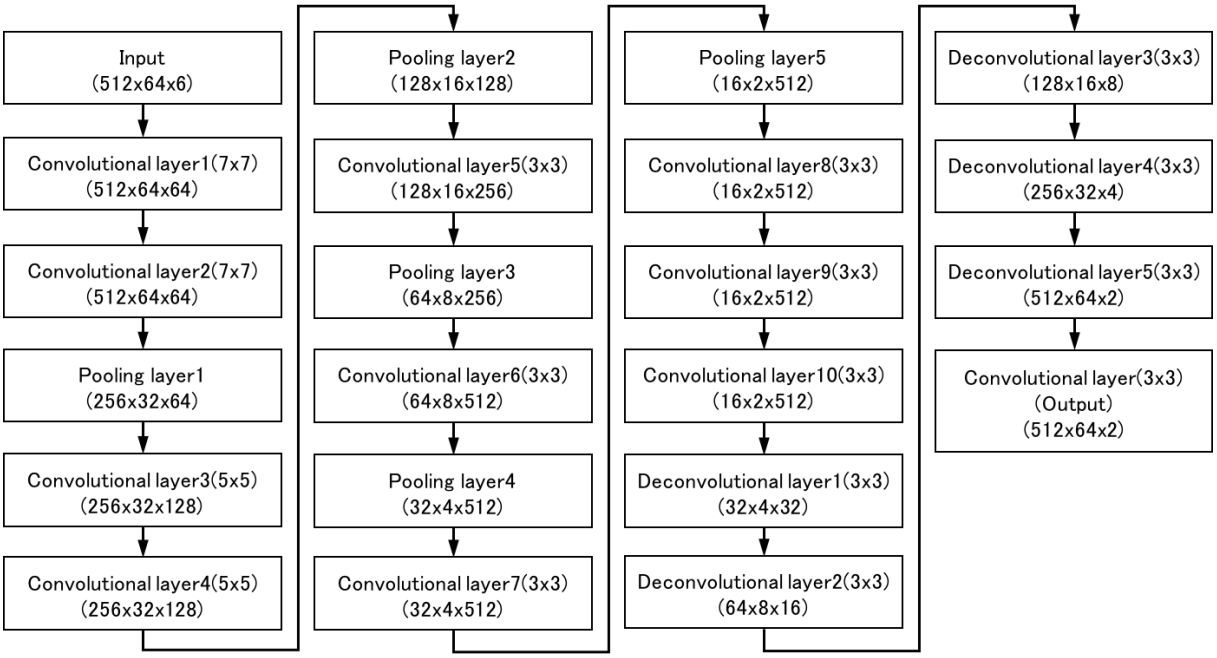

FIGURE 3. Network configuration

TABLE 1. The environment of verification

CPU	Intel Core i7-6700 @ 3.40GHz
Main memory	8GB
GPU	NVIDIA Geforce GTX1060 3GB
OS	Windows 10
Python IDE	pyCharm Community 2017.3.2

TABLE 2. Software and library

Name	Version
Python	3.5.4
Keras	2.1.1
Tensorflow-GPU	1.4.0
cuDNN	6.0
Numpy	1.14.0
OpenCV	3.3.0.10

TABLE 3. Setting parameters

Parameter	Setting
Epoch number	23
Number of images per epoch	300
Number of data per image	882
Learning algorithm	RMSprop

3. **Verification.**

3.1. **Method of verification.** Table 1 shows the environment of the verification experiment. Table 2 shows the software and libraries we used. Table 3 shows the setting of learning of the neural network when the experiment result is obtained. Since there are 300 images per epoch, we are learning with 6,900 images, 6,023,700 data with 23 epochs.

3.2. **Verification result.** This chapter shows a part of the video of the processing result by the neural network. In this paper, we compare the processing result by Optical Flow with the processing result by the proposed method. The result image is composed of four images into one image. The upper left part of the result image shows the part recognized as having an object by the neural network in white. The lower left is the actual image and the upper right is the neural network superimposed on the actual picture with the part recognized as having the object. The image in the lower right is the output image of the previous study.

3.2.1. *Pictures with misrecognition in previous research.* Figure 4 shows the result of processing using the neural network constructed in this research on the video outputted in Figure 1 in the previous study. When an object is detected by the proposed method, the detected part is displayed in red. In Figure 4, there is no object popping out, so there is no object displayed in red. For that reason, in the previous research, misrecognition occurred on the left side of the screen, but in this research misrecognition disappeared.

FIGURE 4. Comparison with previous research

3.2.2. *Image generated simultaneously with teacher data.* Figure 5 shows an output obtained by inputting the image generated simultaneously with the teacher data. This video is composed of images of red cars. In the research image on the lower left, erroneous recognition has occurred due to the pattern of the road surface. On the other hand, in the output by the neural network at the upper right, the part of the car turns white and you can see that it recognizes it.

FIGURE 5. Output result of data set

3.2.3. *Processing with actual video.* Figure 6 shows the output obtained by inputting the image that a person is jumping in front of car. It is taken at the time of the previous research. In the neural network, we have succeeded in recognizing the part of the foot and displaying it in red. In the system of the previous research, it did not respond in this frame, but began to react after several frames. However, the previous research was more strongly responding to the person than this research. Figure 7 shows the output when turning the curve. In the processing result by the Optical Flow, erroneous recognition is occurring in the building behind. On the other hand, no false recognition has occurred in the processing result by the neural network. However, both of them are misrecognized in the mesh pattern of the dashboard. Figure 8 shows the case of a failure occurring in the

FIGURE 6. Results in actual video output

FIGURE 7. Results in case of curve

FIGURE 8. False Negative

image obtained in Figure 6. Figure 8 shows a case (False Negative) when it is not possible to recognize the object to be recognized. In the proposed method, we cannot recognize the person who is jumping out from the right side. However, the same misrecognition has occurred in the system of the previous research. Based on the above processing results, the neural network constructed in this research improved the misrecognition by the Optical Flow and showed stronger response to the object which is jumping out. Moreover, it can be said that the detection accuracy is comparable to that of the previous research.

4. **Discussion.**

4.1. **Impact of changes in the number of frames.**

Figure 9 and Figure 10 show the output results of the neural network learned by changing the input data from 2 frames to 3 frames. However, only 15 epochs were learned and the number of data was 3,928,500. Figure 9 shows a scene corresponding to Figure 8, and Figure 10 shows a scene corresponding to Figure 7. As shown in Figure 9, improvement was observed for False Negative. On the other hand, it can be seen from Figure 10 that the misrecognition at the time of curve is increased. This may be due to the fact that the optimization has not progressed sufficiently due to the short learning. From the above, in the case of three frames of input data, False Negative that could not recognize the jumping out object from the case of two frames was improved. It responded more strongly to the object that jumps in front of car. Therefore, it can be confirmed that it is effective to increase the number of frames to be simultaneously processed from 2 frames to 3 frames. At this time, the change of the neural network is merely the input from $512 \times 64 \times 6$ to $512 \times 64 \times 9$. The

FIGURE 9. Three frame processing in Figure 8

FIGURE 10. Three frame processing in Figure 7

processing time per data is also about 16ms to 20ms, which is not much different from 15ms to 25ms at 2 frames. Therefore, this change does not change the processing time of the neural network. However, it is expected that processing time will be longer than in the case of 2 frames in preprocessing until inputting the video frame into the neural network.

4.2. **Usefulness of real-time processing.** The time taken to process one data of 512×64 pixels in the neural network this time is 15ms to 25ms. In order to shorten the processing time, it is assumed that the object jumps into the center of the image. If only the central 8 data (size is $1{,}024 \times 256$ pixels) is processed, it takes about 200ms to process at once. This is the speed at which processing is possible five times per second. On the other hand, there are companies that assume that the time for forward negligence by checking the speedometer is 0.5 seconds [9]. In other words, at least twice processing will be performed while checking the speedometer, so it seems that it functions as an alarm system at emergency. However, this is the processing speed on a stationary computer. Considering the case of incorporating it into an existing car, it is not suitable for real-time processing at present. In order to solve this problem, we think that it is necessary for real-time processing to install on one processor capable of higher speed processing such as FPGA and to reduce unnecessary layers and channels.

4.3. **Detecting objects farther away.** As a reaction to a long distance object, we could detect a person of about 20m distance. This distance is calculated assuming that a person is 1.7m in height, since a person is displayed as 26 pixels when responding. However, in a situation where an animal or the like jumps out unexpectedly, the car may be traveling at high speed. In such a case, the neural network of this time cannot inform the driver of the danger sufficiently early. Therefore, when expecting to work effectively in reality, it is required to detect objects farther away. First of all, we focused on improving the performance of the camera installed in the drive recorder. Some of the cameras installed in the drive recorder have a recording function with FHD ($1{,}920 \times 1{,}080$ pixels). For example, the input data to the neural network does not reduce the image, and it cuts out the central part of images where there is the object that jumps in front of car. Assuming that the viewing angle of the FHD camera is 120 degrees, the image of 1,024 pixels horizontally processed by the neural network can be thought of as having a viewing angle of about 62 degrees. Assuming 8 pixels on an image with a viewing angle of 62 degrees and 1,024 pixels as 1m, the 1m object is about 110m ahead. Therefore, it is possible to deal with an object three times as much as the image processed by the neural network without enlarging it, and in this case it is considered that it is possible to detect an object about 60m ahead.

5. **Conclusions.** In this paper, we constructed a detection system for objects that jump out in front of cars using Deep Learning. As a result, it was confirmed that the object can be recognized by the constructed neural network. Moreover, compared to the system of the previous research, it can be said that it stands in particular in terms of display to the user. On the other hand, there were still many cases where false recognition occurred. In future prospects, we think that a more detailed study of data set generation is necessary. For this time, we created a data set with emphasis on objects that jump out so that we can construct a neural network that reacts strongly to "objects that jump out in front of cars". Therefore, in order to build a system that can be used more realistically, we felt that it is necessary to add data from the viewpoint that not only we want it to react, but also that we do not want to react. In addition to the data set for learning, a data set for performance evaluation of protruding object detection is also necessary.

REFERENCES

[1] Ministry of Land Infrastructure Transport and Tourism, Recent trend of automatic driving, *The 3rd Vehicle Safety Countermeasure Study Meeting*, 2018.

[2] S. Usui, N. Nomura, H. Kumagai and H. Sekine, Development of improvements to driver assistance system "eyesight" for reduction of traffic accidents, *Journal of the Japan Society of Applied Electromagnetics and Mechanics*, vol.25, no.4, pp.383-389, 2017.

[3] E. Nagata, Y. Oyama and T. Nagasako, Study on detection of wildlife jump-out using monocular in-vehicle camera, *The Institute of Industrial Applications Engineers*, S6-4, 2015.

[4] *Open CV, Optical Flow*, https://docs.opencv.org/3.3.1/d7/d8b/tutorial_py_lucas_kanade.html, accessed 2018-05-10.

[5] T. Kuwazawa, Y. Oyama and M. Nakano, *Proposal of a Method to Detect Objects Jumping in Using Deep Learning*, Committee of Joint Conference of Electrical, Electronics and Information Engineers, Kyushu, 2017.

[6] E. Shelhamer, J. Long and T. Darrell, Fully convolutional network for semantic segmentation, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] Google, *YouTube-8M Dataset*, https://research.google.com/youtube8m/, accessed 2018-05-10.

[8] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *Proc. of the 32nd International Conference on Machine Learning (ICML'15)*, pp.448-456, 2015.

[9] Continental Automotive, *Continental Head-up Display Argumented Reality HUD*, http://continental-head-up-display.com/jp/, accessed 2018-05-10.