Pedestrian Detection Based on Deep Learning

Hyung-Min Jeon
Department of Electrical
and Computer Engineering
Sungkyunkwan University
Suwon, Republic of Korea
Email: hmjeon@skku.edu

Vinh Dinh Nguyen
Department of Electrical
and Computer Engineering
Sungkyunkwan University
Suwon, Republic of Korea
Email: vinhnd@skku.edu

Jae Wook Jeon
Department of Electrical
and Computer Engineering
Sungkyunkwan University
Suwon, Republic of Korea
Email: jwjeon@skku.edu

***Abstract*—While it is a hot issue whether cars could drive by themselves in emergent situations without any kind of human interference, pedestrian detection is the key technology in autonomous driving cars. Though current pedestrian detection technologies have come to a point in which they are accurate in normal conditions and surroundings, existent systems are inaccurate in harsh situations, such as when there are too many pedestrians, when there is too much light or when it is too dark, or when it is raining or snowing heavily. This problem may be solved by integrating deep learning and combining a new type of local pattern with the RGB raw image as input, instead of using just the RGB image as input. We will introduce a new type of local pattern called Triangular Patterns, which is effective in extracting more detailed and stable features from local regions. Here in this paper, we propose a pedestrian detection system in which deep learning is used, along with combining the RGB raw image with Triangular Patterns for input.**

***Keywords*—Pedestrian Detection, Deep Learning, Local Patterns, Machine Learning, Computer Vision.**

I. Introduction

Due to the attention on autonomous driving cars, many companies including Hyundai Automobiles, Google and Tesla are participating in the study of autonomous driving, and many related experiments are being made even outside them, such as an open source simulating software for such systems[1]. Pedestrian detection is one of the most important technologies in making autonomous cars ready for the market, so naturally, studies on making computers detect pedestrians accurately are being done together with studies on autonomous driving. Deep learning is also a hot topic in the area of artificial intelligence. After a game-playing robot brought attention to deep learning worldwide, engineers working in various areas are seeking how they could use deep learning technology in their work.

Consequently, many studies are integrating deep learning into pedestrian detection. Pedestrian detection systems with deep learning neural networks are better than those without, in terms of accuracy and sometimes in terms of speed, too. However, even with deep learning, existing pedestrian detection methods face a major problem: Harsh weather conditions and or surroundings with severe lighting, or too crowded environments often interfere with the accuracy of the pedestrian detection system.

To solve this problem, in this paper, we will use local patterns along with the original RGB raw image as input for the deep learning neural networks input nodes. Here, instead of existing types of local patterns, we use a new type called Triangular Patterns. Instead of methods like linear binary patterns (LBP) and linear ternary patterns (LTP), this method will calculate histograms triangularly, though based on a three by three grid, just like LBP and LTP. Since Triangular Patterns consider three cells at once to calculate the histogram value for one cell, it will extract more detailed and stable features from local regions and thus the system will be more accurate than when LBP or LTP is used. Using Triangular Patterns along with RGB raw image for input can result in extracting more detailed and stable features from the local region, which can help to improve the performance of pedestrian detection. Thus, combining the RGB raw image with its Triangular Pattern can improve accuracy under hostile conditions: Under crowded situations, under illumination changes due to shadow or sunny conditions, or under heavy rain or snow.

II. Background & Related Research

Many researches based upon both non-learning-based methods and learning-based methods have been done on pedestrian detection.

A. *Non-learning-based methods for pedestrian detection*

Many non-learning-based methods for pedestrian detection exist, such as Haar wavelet-based cascade, neural network using local receptive fields, HOG with linear SVM, combined shape-texture-based pedestrian detection, and temporal integration. Among these, the HOG-based linear SVM was significantly better than all other methods in terms of performance. This implies that pedestrians, which have complex features in the view of a computer, are well captured by feature representations using local edge orientation. As tighter processing constraints are imposed, the Haar wavelet-based cascade approach outperforms all other detectors considered[2]. However, no matter how superior a single method is compared to other non-learning-based methods, they still have issues. According to another survey on non-learning-based methods for pedestrian detection for moving pedestrians in a video, the accuracy and speeds of the methods need to be improved, limitations in the input devices computational capabilities and memory.

B. *Learning-based methods for pedestrian detection*

According to a study comparing many methods in the development of learning-based pedestrian detection, the improvement in features of images was the main key in how the pedestrian detection technology could develop[3]. This would mean that obtaining detailed and accurate features from images would be very important in pedestrian detection.

Though not on pedestrian detection, there is a study using tiled convolutional neural networks for object detection. Based on color images in RGB, the paper uses Tile-Convolutional Neural Networks (T-CNN) for object detection in images[4]. Though it is effective, it is hard to use this on pedestrian detection since it slides the detection window throughout the image, line by line. Long computing times are needed for such detection methods, which makes this learning-based method quite inappropriate for detecting pedestrians in videos.

III. Proposed System

We plan to implement a pedestrian detection system based on deep learning. Our system is composed of two big steps: Obtaining the local patterns of the original image and using both the local patterns and the original RGB raw image for input into either T-CNN or Faster R-CNN to detect pedestrians. In this paper, we propose a new local pattern called Triangular Patterns.

A. *Triangular Patterns*

We propose a new histogram calculation method called Triangular Patterns. This new method for obtaining histograms will be used along with the 3D RGB image: Triangular patterns. In this histogram calculation method, in a three by three grid the system, unlike LBP and LTP, considers three cells at once to calculate the histogram value for one cell. Due to this Triangular Patterns would extract more detailed and stable features from local regions than LBP or LTP, resulting in higher pedestrian detection accuracy than when LBP or LTP is used.

Triangular Patterns are calculated like the following: In the image we would like to calculate, we consider a three by three-pixel area, like in LBP and LTP. Then the system draws, not actually but conceptually, a right-angled triangle with one of its points placed on the pixel we want to calculate first. After doing so, for the pixels with the triangles three points on it, we calculate the mean values for red, green and blue each, since it is an RGB image. A simple diagram explaining the triangle is shown in Figure 1. The same goes for the other eight pixels in the three by three-pixel area.

When Triangular Patterns are used together with the original RGB images, it would be possible to detect pedestrians more accurately even under harsh conditions, such as when the scene of the image is very crowded, when theres either too much light or too less light, or when heavy snow or rain is affecting the scenery of the image. Fig. 2 shows a Triangular Pattern and the according original RGB raw image.

B. *Input of Triangular Patterns plus RGB images in T-CNN*

We first planned using Triangular Patterns with T-CNN.

T-CNN can take two types of data at once for input. Here, Triangular Patterns will also be used as input along with the original RGB image instead of using only one type of image as input or combining the two to make one image. Figure 3 shows this as a visual diagram.

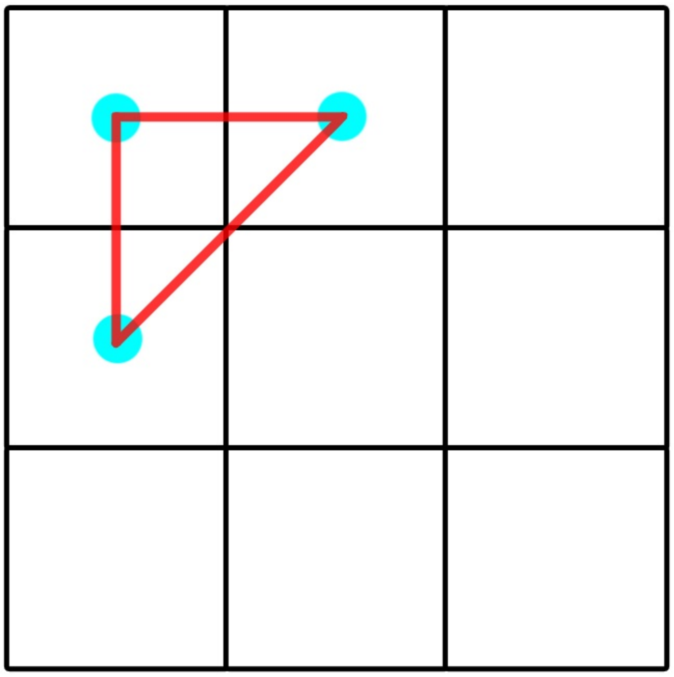
Fig. 1: The conceptual triangle used in calculating Triangular Patterns.

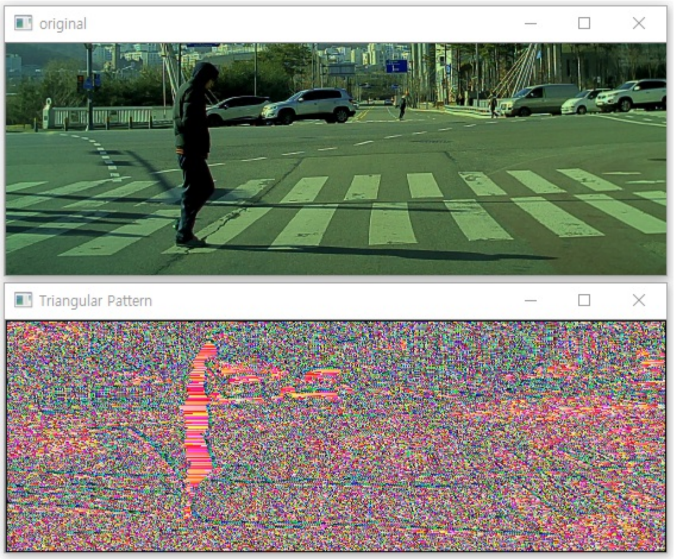

Fig. 2: An image and its Triangular Pattern shown on two different windows.

However, a major disadvantage of using T-CNN in pedestrian detection exists: Since T-CNN searches for the object it is supposed to detect throughout the whole image by sliding the window line by line, it takes time. Pedestrian detection, however, needs to be a real-time process, and this makes T-CNN nearly impossible to use for pedestrian detection.

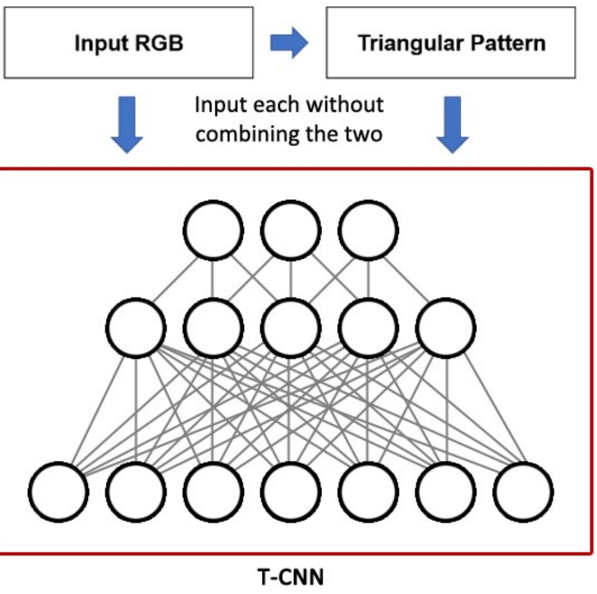

Fig. 3: Diagram of how the Triangular Patterns and RGB raw images will be used as input for T-CNN.

C. Input of Triangular Patterns plus RGB images in Faster R-CNN

To detect pedestrians with our newly altered input image, we will use Faster Recurrent Convolutional Neural Network (Faster R-CNN) , instead of the relatively slow T-CNN[5].

For Faster R-CNN, we first need to concatenate the features by combining the Triangular Pattern with the original RGB raw image. The method of combining the RGB raw image with its Triangular Pattern is simple; calculating the mean RGB value. If we say the red, green and blue values of a certain pixel of the RGB raw image are R_A, G_A, and B_A, the red, green and blue values of the according pixel in the Triangular Pattern as R_T, G_T, and B_T, and the red, green and blue values of the final combined image as R_I, G_I, and B_I,:

$$R_I = (R_A + R_T)/2,\ \ G_I = (G_A + G_T)/2,\ \ B_I = (B_A + B_T)/2$$

This final combined image of RGB value R_I, G_I, and B_I is the image we will use as input for Faster R-CNN. When the concatenated features are put in the input nodes as input for the Faster R-CNN, anchors/boxes of various sizes will be readied, since the system does not know what size the pedestrians will be in. Then the regions in which the pedestrians might exist will be proposed within the system, and after finding such regions, only the top anchors are kept. In other words, regions that contain only a small amount of what the system thinks we seek to find are deleted, for example a box with only a pedestrians hand in it. Since we use Faster R-CNN instead of R-CNN, ROI pooling comes afterwards. ROI pooling helps enhance the speed of extracting features. After ROI pooling, two steps are done in parallel at the same time: Classifying whether the image inside the box is an object we want-in this systems case a pedestrian-or whether it is not, and finding the most refined bounding box using the regressor. If the classifier determines that the image inside the region is a pedestrian instead of part of the background, and the regressor succeeds in finding the refined bounding box, the system detects it as a pedestrian. Otherwise, if either the classifier identifies the region as a part of the background or the regressor fails in finding the refined bounding box, the region will be not marked in the window showing the final pedestrian detection result. Fig. 4 shows the whole process in a simple diagram.

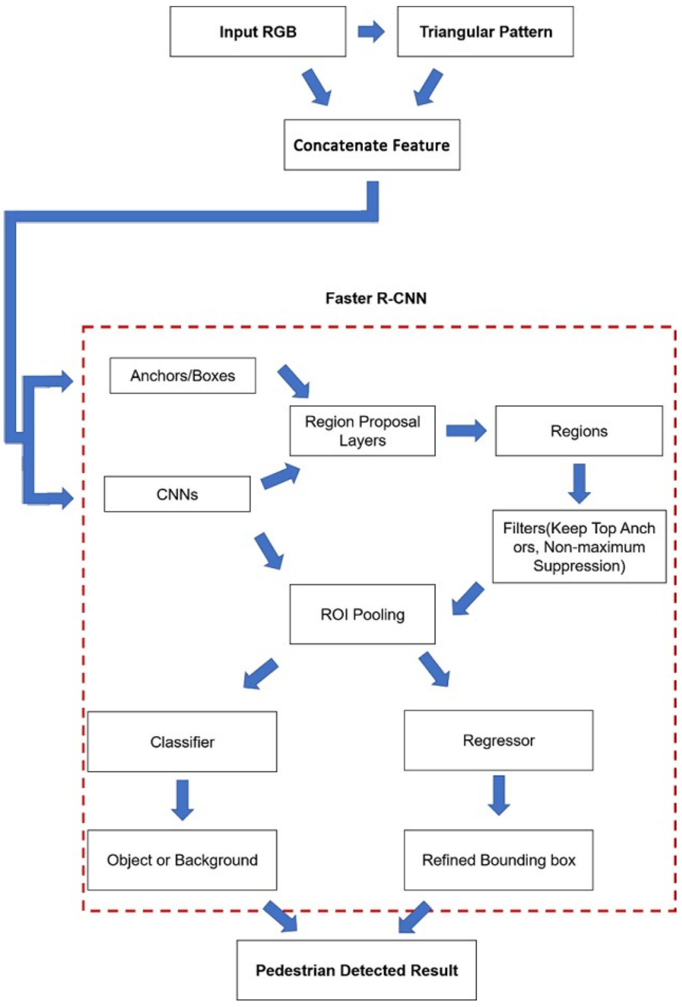

Fig. 4: A graphic overview of the system using Faster R-CNN.

D. Implementation

The system, overall, was programmed with the C programming language in a development environment of a computer with Nvidias Titan X graphic card, and Windows 10 Home for an OS.

The deep learning neural network used in the system was built using the Caffe framework tool[6]. It is composed of 224*224 input nodes, nine layers, and two output nodes. Each of the two output nodes stand for pedestrians are existent here and pedestrians are non-existent here.

Training and testing were done with the KITTI dataset and our labs dataset combined. Fig. 5 shows two images from the KITTI dataset and Fig. 6 shows an image from the lab dataset. Though the size of the images in the two types of datasets were different from each other, the KITTI dataset having 1224*370 pixels and the lab dataset 1280*672 pixels, the images were reduced to 224*224 pixels for the input so the difference in size did not matter for the training and testing. The total number of RGB raw images for the test set was 900, and the training set consisted of 2000 RGB raw images.

Fig. 5: Images from the KITTI training dataset[7].

Fig. 6: An image from the lab dataset.

IV. Results

A. Evaluation

We evaluated the performance on a computer with Intel Core i7, 4GHz, 16GB RAM. Nvidias GPU Titan X was also used to accelerate the performance of the proposed system. The average processing time was approximately 25ms per frame.

To evaluate the performance of the proposed system we used metrics, recognition rate per frame like the following:

-Description: Evaluating the accuracy of the proposed system under normal conditions, recognition all of people when there are many people.

-When finding pedestrians, do not detect: People hidden from camera whom more than 30% of their bodies are hidden, people who leaned more than 30%, people who look shorter than 85cm to the camera, people on motorcycles/bicycles

-Evaluation method: The overlapped region of the detected result and the ground truth region must be greater than 50% of the ground truths region, like the sample shown in Figure 7.

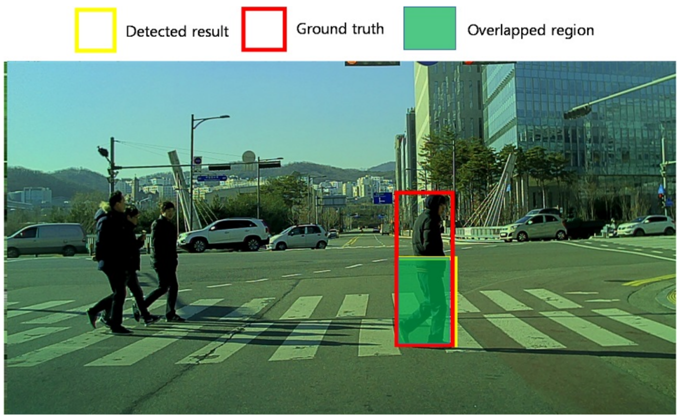

Fig. 7: Sample of the ground truth box, the detected results box and the overlapped region between the two.

To evaluate the performance of the proposed system on our lab data set, we selected three sequences with various scenarios from our lab data in order to evaluate the performance of the proposed system as shown in Figures 8, 9 and 10.

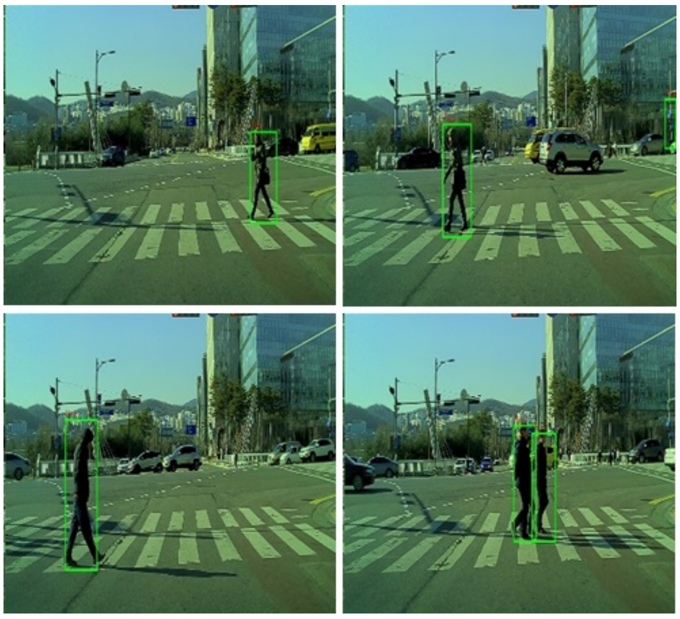
Fig. 8: Detected results using Sequence 1 of our lab dataset.

Among the three sequences, Sequence 01, which detection result samples are shown in Figure 8, was recorded under normal conditions. Sequence 02, which detection result samples are shown in Figure 9, was captured under illumination changes. Last, Sequence 03, which detection result samples are shown in Figure 10, had scenery under the shadows of buildings. Here, Sequence 2 and Sequence 3 can be seen as hostile conditions.

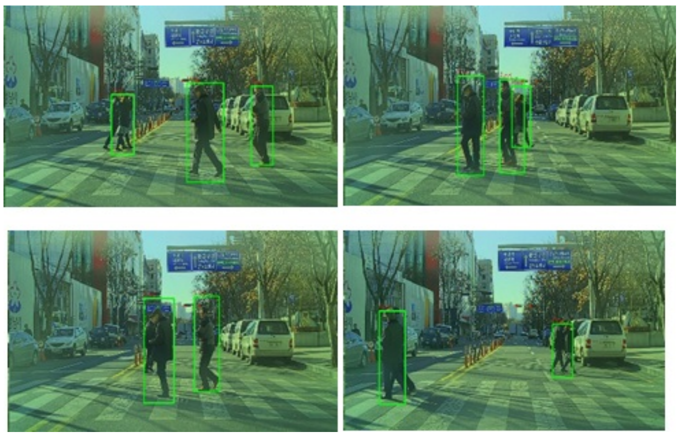

Fig. 9: Detected results using Sequence 2 of our lab dataset.

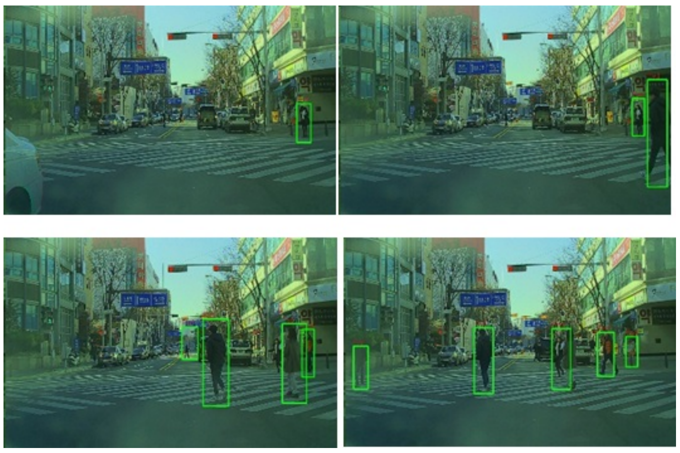

Fig. 10: Detected results using Sequence 3 of our lab dataset.

B. Results & Analysis

The processing speed of our proposed system was 25 frames per second.

Fig. 11 shows the Triangular Pattern of a certain image. Fig. 12 shows the pedestrian detection results for the image which Triangular Patterns are shown in Fig. 11.

It is visible that the detection results are quite accurate. Then how accurate is it in numbers, and how much is it accurate compared to normal Faster R-CNN? Table 1 shows the accuracy of the proposed system for the three sequences used for evaluation. Table 2 shows the accuracy of the original Faster R-CNN under the same three sequences.

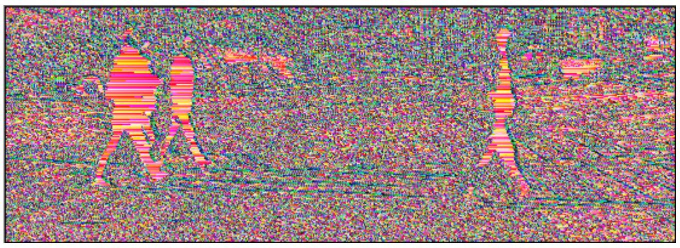

Fig. 11: The Triangular Pattern of an RGB raw image.

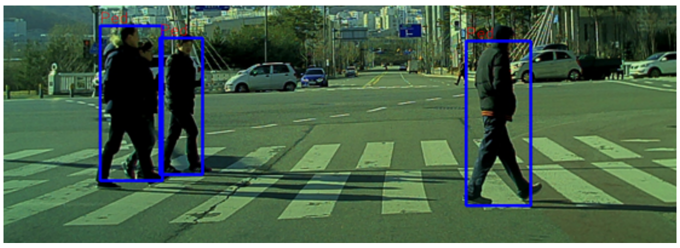

Fig. 12: Pedestrian detection results for the image which Triangular Pattern is shown in Fig. 11.

TABLE I: Accuracy of the proposed system under various sequences and overlapped ratio of height.

	Overlapped 30%	**Overlapped 40%**	**Overlapped 50%**	**Overlapped 60%**	**Overlapped 70%**
Sequence 01 (300 images)	98.55%	98.55%	98.55%	98.55%	98.55%
Sequence 02 (300 images)	82.45%	82.45%	82.45%	82.45%	81.15%
Sequence 03 (300 images)	75.42%	75.42%	75.42%	75.42%	74.35%

For Sequence 01, for our proposed system, cases in which the detection result and the ground truths overlapping region was 30 70% of the ground truths region were 2.20% more than the original Faster R-CNN. For Sequence 02, cases in which the detection result and the ground truths overlapping region was 30 60% of the ground truths region were 2.30% more than the original Faster R-CNN, and for cases in which the detection result and the ground truths overlapping region was 70% of the ground truths region, the percentage was 3.30% higher. For Sequence 03, cases in which the detection result and the ground truths overlapping region was 30 60% of the ground truths region were 3.07% more than the original Faster R-CNN, and for cases in which the detection result and the ground truths overlapping region was % of the ground truths region, the percentage was 3.20% higher. Through these comparison results, for all three sequences, we could find out that the proposed system obtained better performance than the original Faster R-CNN under these experiments.

Table 3 shows the accuracy of when LBP is used instead of Triangular Patterns (LBP is combined with the RGB image as input for Faster R-CNN) for the three sequences. Table 4 shows the accuracy of when LTP is used instead of Triangular Patterns (LTP is combined with the RGB image as input for Faster R-CNN) Faster R-CNN for the three sequences.

Comparing Table 1 with Tables 3 and 4, we can see that for Sequence 01, our proposed system is 1.00% more accurate than

TABLE II: Accuracy of the original Faster R-CNN under various sequences and overlapped ratio of height.

	Overlapped 30%	**Overlapped 40%**	**Overlapped 50%**	**Overlapped 60%**	**Overlapped 70%**
Sequence 01 (300 images)	96.35%	96.35%	96.35%	96.35%	96.35%
Sequence 02 (300 images)	80.15%	80.15%	80.15%	80.15%	77.85%
Sequence 03 (300 images)	72.35%	72.35%	72.35%	72.35%	71.15%

TABLE III: Accuracy of when LBP is used in place of Triangular Patterns under various sequences and overlapped ratio of height.

	Overlapped 30%	**Overlapped 40%**	**Overlapped 50%**	**Overlapped 60%**	**Overlapped 70%**
Sequence 01 (300 images)	97.55%	97.55%	97.55%	97.55%	97.55%
Sequence 02 (300 images)	81.25%	81.25%	81.25%	81.25%	78.35%
Sequence 03 (300 images)	73.26%	73.26%	73.26%	73.26%	72.15%

TABLE IV: Accuracy of when LTP is used in place of Triangular Patterns under various sequences and overlapped ratio of height.

	Overlapped 30%	**Overlapped 40%**	**Overlapped 50%**	**Overlapped 60%**	**Overlapped 70%**
Sequence 01 (300 images)	97.75	97.75	97.75	97.75	97.75
Sequence 02 (300 images)	81.65%	81.65%	81.65%	81.65%	78.65%
Sequence 03 (300 images)	73.55%	73.55%	73.55%	73.55%	72.85%

when LBP is used instead of Triangular Patterns and 0.80% more accurate than when LTP is used instead of Triangular Patterns. For Sequence 02, cases in which the detection result and the ground truths overlapping region was 30 60% of the ground truths region were 1.20% more than when LBP was used instead of Triangular Patterns and 0.8% more than when LTP was used instead of Triangular Patterns, and for cases in which the detection result and the ground truths overlapping region was 70% of the ground truths region, our proposed system showed a 2.80% higher percentage when compared to using LBP instead of Triangular Patterns and 2.50% higher percentage when compared to using LTP instead of Triangular Patterns. As for Sequence 03, our system compared to using LBP instead of Triangular Patterns and to using LTP instead of Triangular Patterns had 2.16% and 1.87% more cases each in which the detection result and the ground truths overlapping region was 30 60% of the ground truths region, and had 2.2% and 1.5% more cases each in which the detection result and the ground truths overlapping region was 70% of the ground truths region.

Such comparisons prove that using Triangular Patterns and combining it with the RGB raw image for input is more effective than combining LBP or LTP with the RGB raw image in increasing the accuracy of pedestrian detection, especially under hostile conditions. Not only are Triangular Patterns conceptually more accurate than LBP or LTP, but also more accurate in terms of actual results.

V. Conclusion

By evaluating our system with metrics, evaluating the original Faster R-CNN with the same metrics and method and then comparing the two results, we could know that our proposed system showed better performance in terms of accuracy. This proves that using Triangular Patterns with the RGB raw image for input is effective in increasing the accuracy of pedestrian detection.

In the future, we plan to implement the proposed system on Nvidias Drive PX2 hardware, which has a different GPU architecture with a normal computer. Shown in Figure 13, Drive PX2 has a computation power about that of a desktop computer equipped with a Nvidia Geforce GTX TITAN X graphics card and is used for various projects worldwide including projects for autonomous driving.

After we implement the system on Nvidias Drive PX2, we also plan to increase the accuracy of our system even for harsh environments by training and testing our program on hostile conditions. In this paper, we could not show image results for scenes under hostile conditions. Since our proposed system is more accurate than the original Faster R-CNN under normal conditions, it would likely also be more accurate under hostile conditions. However, pedestrians may walk in front of autonomous cars not only on clear days with appropriate illumination, but also days with harsh weather or with too much light or at night, so we must make our pedestrian detection system as accurate as possible even for such conditions.

Acknowledgment

This work was supported by the Technology Innovation Program/Industrial Strategic Technology Development Program (10080011, Development of front/side camera sensor for autonomous vehicle) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea).

References

[1] Sangho Lee, Janghee Cho, and Da Young Ju. 2013. Autonomous Vehicle Simulation Project. IJSEIA 7, 5 (Sept. 2013), 393402.

[2] Markus Enzweiler and Dariu M. Gavrila. 2009. Monocular Pedestrian Detection: Survey and Experiments. IEEE Transactions on Pattern Analysis Machine Intelligence 31 (2009). 2179-2195.

[3] Rodrigo Benenson, Mohamed Omran, Jan Hendrik Hosang, and Bernt Schiele. 2014. Ten Years of Pedestrian Detection, What Have We Learned? CoRR abs/1411.4304 (2014). arXiv:1411.4304 http://arxiv.org/abs/1411.4304

[4] Jiquan Ngiam, Zhenghao Chen, Daniel Chia, Pang W. Koh, Quoc V. Le, and Andrew Y. Ng. 2010. Tiled convolutional neural networks. In Advances in Neural Information Processing Systems 23, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta (Eds.). Curran Associates, Inc., 12791287.

[5] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. CoRR abs/1506.01497 (2015). arXiv:1506.01497

[6] Yangqing Jia, Evan Shelhamer, and Jeff Donahue Sergey Karayev. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In Proceedings of the 22nd ACM international conference on Multimedia (MM 14). ACM, New York, NY, USA, 675678.

[7] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets Robotics: The KITTI Dataset. International Journal of Robotics Research (IJRR) (2013).

[8] Fred Lambert. 2016. All new Teslas are equipped with NVIDIA's new Drive PX 2 AI platform for self-driving. Electrek. Retrieved 25 January 2017.