Pedestrian and Part Position Detection using a Regression-based Multiple Task Deep Convolutional Neural Network

Takayoshi Yamashita
Computer Science Department
Chubu University
Aichi 487–8501, Japan
yamashita@cs.chubu.ac.jp

Hiroshi Fukui
Computer Science Department
Chubu University
Aichi 487–8501, Japan
fhiro@vision.cs.chubu.ac.jp

Yuji Yamauchi
Robot Department
Chubu University
Aichi 487–8501, Japan
yuu@vision.cs.chubu.ac.jp

Hironobu Fujiyoshi
Robot Department
Chubu University
Aichi 487–8501, Japan
hf@cs.chubu.ac.jp

***Abstract*—In driving support systems, it is not only necessary to detect the position of pedestrians, but also to estimate the distance between a pedestrian and the vehicle. In general approaches using monocular cameras, the upper and lower positions of each pedestrian are detected using a bounding box obtained from a pedestrian detection technique. The distance between the pedestrian and the vehicle is then estimated using these positions and the camera parameters. This conventional framework uses independent pedestrian detection and position detection processes to estimate the distance. In this paper, we propose a method to detect both the pedestrian and their position simultaneously using a regression-based deep convolutional neural network (DCNN). This simultaneous detection method is possible to train efficient features for both tasks, because it is attention to head and leg regions from given labels. In the experiments, our method improves the performance of pedestrian detection compared with the DCNN which detects only pedestrian. The proposed approach also improves the detection accuracy of the head and leg positions compared with the methods that detect only these positions. Using the results of position detection and the camera parameters, our method achieves distance estimation to within 5% error.**

I. Introduction

Traffic accidents between cars and pedestrians are a likely occurrence. In support systems for elderly drivers and autonomous cars, pedestrian detection is an important technology for avoiding accidents or alerting the driver to dangerous situations. To estimate the distance between a pedestrian and a vehicle, researchers have developed monocular camera-based approaches, stereo camera-based approaches, and LIDAR-based approaches. LIDAR measures the time taken for an irradiating laser beam to return from an object. It has a low resolution, as there is a limit to the number of laser irradiations. In addition, the expense of LIDAR systems makes them unfeasible in most publicly available vehicles. In contrast, methods using stereo cameras estimate distances from the parallax of the two cameras. Although such systems are increasingly common on public vehicles, there is a limit to the miniaturization of the apparatus. Methods using monocular cameras consist of three processes: pedestrian detection, position estimation of the upper and lower parts of the pedestrian, and distance estimation using this positional information with the camera parameters. Monocular camera systems are relatively inexpensive, and can be miniaturized. Because each process is conducted separately, the performance of monocular camera systems is worse than that of other methods. Thus, monocular camera-based approaches require improved pedestrian detection and position detection methods. We propose a method that detects both the pedestrian and their position simultaneously, using a regression-based deep convolutional neural network (DCNN). DCNN [1] have achieved state-of-the-art performance in various fields, including pedestrian detection [2]. In addition, DCNN make it possible to perform multiple tasks in a single network. Zhang proposed a method for the simultaneous detection of facial points, face poses, glasses, smiles, and gender [3]. We apply a DCNN to detect pedestrians and their head and leg positions using a regression technique in a single network. Our approach can also estimate the distance between the pedestrian and the vehicle using the positions obtained from the DCNN.

II. Related work

Methods to estimate the distance between pedestrians and vehicles can be categorized into approaches using monocular cameras, stereo cameras, or LIDAR. Methods based on LIDAR obtain a point cloud from an irradiating laser and recognize objects such as pedestrians from the shape of the point cloud. While this method has the advantage that it is barely affected by weather conditions, it has the critical issue of being too expensive for publicly available vehicles.

Stereo camera approaches recognize the shape and position of three-dimensional objects using a disparity map obtained from the two cameras. In a vehicular environment, two cameras are installed onboard the vehicle, and pedestrians are detected from the parallax of the two cameras. Zhao proposed a method that extracts edge features using a disparity map and color image, and inputs features to a multi-layer perceptron. This allows high-accuracy pedestrian detection to be performed in real time [4]. Because the disparity between the stereo cameras occurs from the differences in position of a common object in each image, it becomes difficult to recognize pedestrians standing in front of other objects such as vehicles or buildings.

Monocular camera systems estimate distance by first detecting the pedestrian, then detecting the positions of upper and lower parts of their body, such as the head and legs. Finally, the distance between the pedestrian and the vehicle is estimated using this positional information. Dalal proposed

a breakthrough pedestrian detection method that extracts gradient features using HOG [5], and various improved methods have since been developed [6][7][8]. To handle variations in the shape of pedestrians, the deformable part model detects the whole body and partial regions simultaneously [9].

DCNN [1] have attracted attention for their ability to extract efficient features, and have achieved state-of-the-art performance on various benchmark datasets, including one focused on pedestrian detection [2]. Ouyang proposed the two-stage structure of joint deep learning, which obtains scores for each body part and detects the pedestrian region from the score and feature map of the DCNN. Joint deep learning is robust to various body poses as it considers information about separate body parts. One of the great advantages of DCNN is their ability to handle multiple tasks. Zhang proposed an improved method for facial part detection that simultaneously recognizes face poses, whether the person is wearing glasses or smiling, and their gender [3]. In this paper, we employ this advantage of DCNN for pedestrian detection. Our DCNN detects the presence of a pedestrian and simultaneously determines their head and leg positions.

Estimating the distance between a pedestrian and a vehicle using a monocular camera requires the upper and lower positions of the pedestrian region. Kishino has proposed a method for estimating this distance using a projective method [11]. We also employ this projective method to the estimation process using the head and leg positions obtained from the DCNN.

III. Proposed method

We propose a regression-based DCNN that detects pedestrians and simultaneously regresses their head and leg positions. First, we introduce the structure and training process of the DCNN for this regression task.

A. DCNN for regression task

As shown in Fig. 1, a DCNN consists of three layer types, namely a convolutional layer, pooling layer, and fully connected layer. The convolutional layers and pooling layers are arranged hierarchically, and the fully connected layers are positioned after them. The DCNN can accept any type of input image (e.g., color, grayscale, gradient, or preprocessed). In the convolutional layer, the input image is convoluted and filtered to size $k_x \times k_y$, and the whole input image is subjected to the same filter. The convolved value x is input to the activation function $f(x)$ to give one value of the feature map. The convolutional layer has M filters, and feature maps are obtained from each filter. The traditional activation function is a sigmoid form, although Rectified Linear Units (ReLU) and Maxout are becoming more common, because the gradient of these functions does not vanish in the training process. Because the sigmoid function $f(x)$ outputs values of approximately 1 for large values of x, large gradient values cannot be obtained. However, the ReLU proposed by Krizhevsky outputs the same value as the input when the input value is larger than 0. This linear nature ensures that, even when x is large, the gradient is equal to 1. Maxout selects the maximum value h' in a particular position h_k across all feature maps [12]. When there are M filters in the convolution layer, the feature maps are split into sets of k and the maximum value h' is selected at each position, as described by Eq. (1).

$$h' = \max_{k \in [1,K]} h_k \tag{1}$$

The feature maps are applied in the reduction process at the pooling layer. There are several pooling processes, such as max pooling, average pooling, and Lp pooling. Max pooling selects the maximum value within a specified small region (i.e., 2×2), whereas average pooling calculates the mean value in this specific region. Among the various pooling types, max pooling is the most popular approach.

The convolutional layers and pooling layers are hierarchically layered to construct a deep network. The fully connected layers, which are the same as in conventional neural networks, follow this layered network, as shown in Eq. (2). The input vector to the fully connected layer contains flattened values from previous feature maps. The output value $h_i(x)$ obtained from the activation function $f(\cdot)$ is the same as in the convolutional layer.

$$h_i(\boldsymbol{x}) = f\left(\sum_{j=1}^{N} w_{ij} x_j + b_i\right) \tag{2}$$

In recognition tasks, the output layer has the same units as the number of recognition classes. A specific unit outputs a class probability of close to 1, whereas other units output values close to 0. In our method, the DCNN for regression has two units for pedestrian detection corresponding to the probability of pedestrian and background, and six units for position detection corresponding to the x and y coordinates of the head and the right and left legs. The output layer in this network has a total of eight units, each of which output values in the range $[0, 1]$. By multiplying the image size, we obtain the coordinate values in the image. The differences in how the DCNN handles recognition and regression tasks lie in the calculation of output values. Whereas the probability of each class is given by the Softmax function in recognition tasks, it comes from an identity function in regression tasks.

B. Training the DCNN

The parameters of the DCNN consist of filter values and connection weights, along with their biases. These numerous parameters are optimized by an iterative process using stochastic gradient descent and backpropagation. The parameters are randomly initialized. Backpropagation calculates the error between the output values and supervised label values of each image E_m, and accumulates these to give the total error E, as described by Eq. (3).

$$E = \frac{1}{2} \sum_{m=1}^{M} E_m \tag{3}$$

where $\{m|1, ..., M\}$ is the set of training samples. The mean square error, which is the difference between the output value y_k and label t_k in class k, is calculated using Eq. (4).

$$E_m = \sum_{k=1}^{C} (t_k - y_k)^2 \tag{4}$$

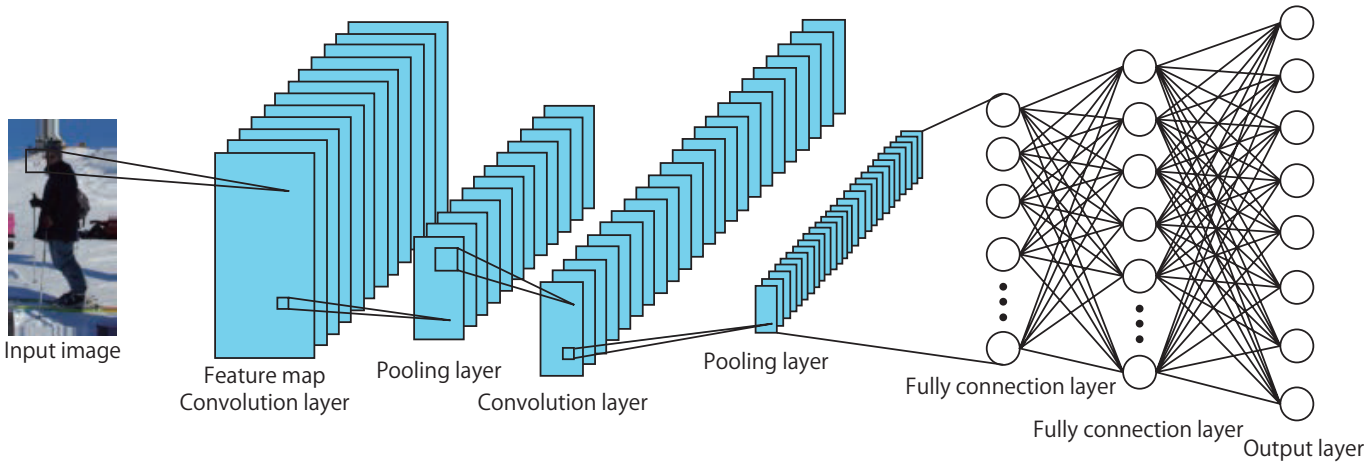

Fig. 1. Structure of the Deep Convolutional Neural Network. It consists of three layer types, namely a convolutional layer, pooling layer, and fully connected layer. The convolutional layers and pooling layers are arranged hierarchically, and the fully connected layers are followed.

Note that C indicates the number of units in the output layer. Backpropagation updates the filter values and weights so as to minimize the total error E by the gradient descent method. The updated value of each parameter comes from the partial derivative of E, as given by Eq. (5).

$$w_{ji}^{(l)} \leftarrow w_{ji}^{(l)} + \Delta w_{ji}^{(l)} = w_{ji}^{(l)} - \lambda \frac{\partial E}{\partial w_{ji}^{(l)}} \tag{5}$$

where λ is the learning rate of the training process, and $w_{ji}^{(l)}$ is the weight of the connection between unit i in layer l and unit j in layer $l+1$. The update amount and gradient are obtained from Eq.(6) and Eq.(7), respectively.

$$\Delta w_{ji}^{(l)} = -\lambda \delta_k^{(l)} y_j^{(l-1)} \tag{6}$$

$$\delta_k^{(l)} = e_k f(V_k^{(l)}) \tag{7}$$

where

$$V_k^{(l)} = \sum_j w_{kj}^{(l)} * y_j^{(l-1)} \tag{8}$$

and $y_j^{(l-1)}$ is the output value of unit j in layer $(l-1)$, e_k is the error of unit k, and $V_k^{(l)}$ is the weighted accumulation value of unit k. The parameters are updated iteratively until the maximum number of iterations has been reached or some convergence condition has been satisfied. There are three ways to input the training samples and obtain the error E: *full-batch*, *online*, or *mini-batch*. *Mini-batch* updates the parameters according to a small subset of samples at each iteration. This can obtain a sufficient change over each iteration, and the update time is relatively short. For these reasons, *Mini-batch* is commonly used to train DCNN.

The dropout method improves the robustness of DCNN and prevents overfitting. This method sets the value of randomly selected units to 0 and continues training the remaining units at each iteration, as shown in Fig. 2. The number of units set to 0 is pre-defined, and is commonly 50%. Dropout enables the update amount to propagate to lower layers, and the network is able to recognize the connection between different parts.

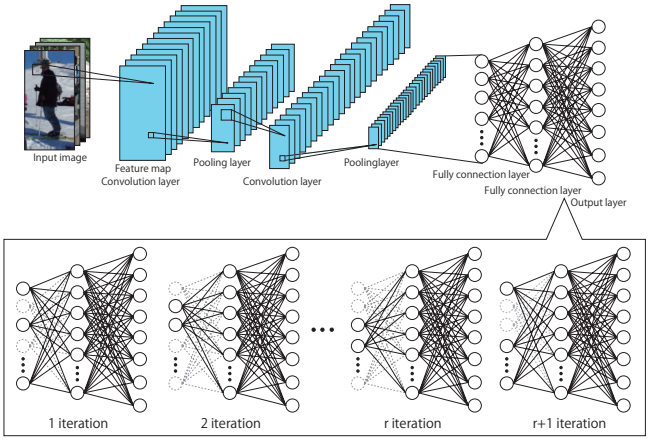

Fig. 2. Fully connected layer with dropout. It sets the corresponding value of randomly selected units to 0 and continues training the remaining units at each iteration.

C. Estimating the distance to a pedestrian

Methods of estimating the distance between a pedestrian and a vehicle using monocular cameras can be categorized into approaches that use the size of the observed object and those that use its position. In the case of the object size, it is difficult obtain a stable distance value as errors are introduced by differences in height. We therefore employ an approach that uses the object position and assumes that the legs are grounded. When the camera is fixed at a height of h and the plane of the camera is parallel to the ground, the y coordinate of an image is given by the following equation:

$$y = \frac{fh}{d} + \frac{H}{2} \tag{9}$$

Note that d is the distance between the pedestrian and the camera, f is the focal distance, and H is the height of the image. The distance is given by Eq. (10).

Fig. 3. Examples from the datasets. To reduce the processing time and evaluate the efficient of output structure, the test images are cropped in pedestrian and background regions.

$$d = fh\frac{2}{2y - H} \tag{10}$$

IV. Experiments

We evaluated the proposed regression DCNN to confirm the effectiveness of our method. In these experiments, we evaluated the accuracy of pedestrian detection, accuracy of position detection, and estimation accuracy of distance in real environments. We used the INRIA Person Dataset and Daimler Mono Pedestrian Dataset for pedestrian detection and position detection. The Daimler dataset contains 31,320 positive samples and 254,356 negative samples for training, and 21,790 samples for testing. We applied a data augmentation technique to increase the number of positive samples to 250,560. The INRIA dataset contains 2,100 positive samples and 50,000 negative samples for training, and 1,000 positive samples and 9,000 negative samples for testing. We again used data augmentation to increase the number of positive samples in the training set to 50,000.

To evaluate the pedestrian detection performance, we compared the results given by our method with those from a detection-only DCNN. This DCNN had two units in its output layer, corresponding to the probabilities of pedestrian and background, as in recognition tasks. For the position detection experiment, we compared our results with those from a position-only DCNN. This DCNN had six units, corresponding to the x and y coordinates of the head and the left and right legs. Table I summarizes the structure of each DCNN. The proposed and comparison methods consist of three convolution and pooling layers. The fully connected layers correspond to the input size of each DCNN, i.e., 96×48 grayscale images for the Daimler dataset and 128×64 color images for the INRIA dataset.

To determine the accuracy of the distance estimates, we require the camera parameters and the true distance between the pedestrian and the camera. However, neither the Daimler dataset nor the INRIA dataset include these data. Thus, we examined the datasets to evaluate these distances.

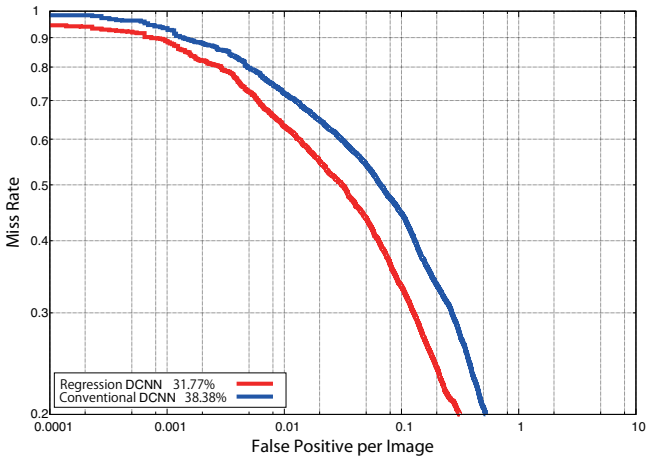

Fig. 4. Comparison result using the Daimler dataset. The dataset contains 21,790 grayscale images for testing.

A. Comparison of pedestrian detection

In general pedestrian detection based on DCNN, candidate regions are obtained by another classifier during a preprocessing stage to reduce the processing time and the number of false positives. Thus, we employed HOG- and SVM-based methods to detect candidate regions. Fig. 4 shows the detection accuracy using the Daimler dataset. For a false positive per image (FPPI) rate of 0.1 the miss rate of the conventional DCNN is 38%, whereas our method achieves a lower rate of 32%. Our DCNN is able to focus on characteristic pedestrian regions, since the coordinates of the head and legs are given as supervised labels.

Fig. 5 shows the detection accuracy on the INRIA dataset. This dataset contains variations in human poses. Because the HOG and SVM classifiers find it difficult to detect pose variations, we evaluated the pedestrian region from annotations and a randomly cropped background. As a result, Fig. 5 shows that the miss rate of our method was just 14% at an FPPW of 0.1. This represents a significant improvement over the conventional DCNN, which achieved a miss rate of 39%.

TABLE I. STRUCTURE OF THE DCNN FOR EACH DATASET

(a) Daimler dataset

Method	Input	Layer1			Layer2			Layer3			Layer4	Layer5	Layer6	Output
		Conv	Max Pooling	Maxout	Conv	Max Pooling	Maxout	Conv	Max Pooling	Maxout	# of unit	# of unit	# of unit	
detection	96x48x1	8,5x3	2x2	2	16,5x4	2x2	2	32,5x4	2x2	2	1,000	500	100	Softmax 2
regression	96x48x1	8,5x3	2x2	2	16,5x4	2x2	2	32,5x4	2x2	2	1,000	500	100	Sigmoid 6
proposed	96x48x1	8,5x3	2x2	2	16,5x4	2x2	2	32,5x4	2x2	2	1,000	500	100	Sigmoid 8

(b) INRIA dataset

Method	Input	Layer1			Layer2			Layer3			Layer4	Layer5	Layer6	Output
		Conv	Max Pooling	Maxout	Conv	Max Pooling	Maxout	Conv	Max Pooling	Maxout	# of unit	# of unit	# of unit	
detection	64x128x3	20,9x5	2x2	2	64,5x3	2x2	2	32,3x3	2x2	2	1,000	500	100	Softmax 2
regression	128x64x3	20,9x5	2x2	2	64,5x3	2x2	2	32,3x3	2x2	2	1,000	500	100	Sigmoid 6
proposed	128x64x3	20,9x5	2x2	2	64,5x3	2x2	2	32,3x3	2x2	2	1,000	500	100	Sigmoid 8

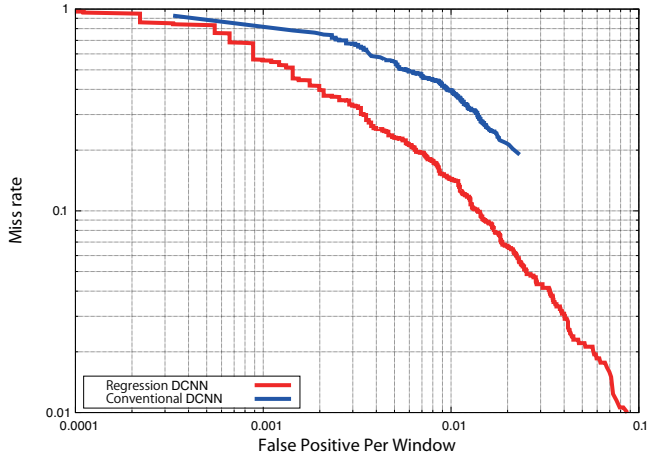

Fig. 5. Comparison result using the INRIA dataset. We collects1,000 positive color images that are cropped from annotations and 9,000 negative color images that are randomly cropped from test dataset.

TABLE II. ERROR IN THE POSITION OF EACH PART [PIXELS]

(a) Daimler dataset

methods	part			average
	head	left leg	right leg	
regression DCNN	6.1	5.9	10.7	7.6
Proposed DCNN	4.2	5.3	9.4	6.3

(b) INRIA dataset

methods	part			average
	head	left leg	right leg	
regression DCNN	8.1	8.7	9.9	8.9
Proposed DCNN	6.5	8.0	9.2	7.9

B. Comparison of position detection

Table II indicates the accuracy of each position with the Daimler and INRIA datasets. The proposed DCNN reduces the error compared with the regression DCNN, giving a 6% error with respect to the image size of 96×48 in the Daimler dataset. The proposed method also reduces the position error to 7.9 pixels with the INRIA dataset, an error of 5.5% for the image size of 128×64. Fig. 6 shows examples of the position detection results using the INRIA dataset. The green points denote position detection results and the red points are the ground truth. These results indicate that our method obtains better position detection results for various poses.

TABLE III. ACCURACY OF DISTANCE ESTIMATION

distance	estimation	error[%]
5m	4.89m	2.2
10m	9.26m	5.3
15m	14.12m	5.8

C. Comparison of distance estimation in a real environment

We evaluated the distance between the pedestrian and the camera using the position detection results. The evaluation images were captured at distances of 5 m, 10 m, and 15 m with known camera parameters. At each distance, seven walking images were obtained. We then evaluated the mean distance accuracy at each distance. The estimation accuracy and examples of the result images are given in Table III and Fig. 7, respectively. The green points denote position detection results and the red points are the ground truth. Our method is clearly able to estimate head and leg positions for various pedestrian poses. The estimation distance is 4.89 m at the 5 m distance, an error of 2.2%. At greater distances, although the estimation accuracy is worse, the error remains less than 5%, even at a distance of more than 10 m.

V. CONCLUSION

In this paper, we have proposed a regression-based DCNN that simultaneously detects pedestrians and the positions of their head and legs. The output layer of our DCNN has units for the probabilities of pedestrian and background, and for the x, y coordinates of the head and legs. By performing multiple tasks that have strong relationships, it is possible to obtain effective features. As a result, the accuracy of both pedestrian detection and position detection improved as compared with a single-task DCNN. In addition, the proposed method achieved a distance estimation error of less than 5% using the position detection results and camera parameters.

Fig. 6. Examples of position detection using the INRIA dataset. Red points are ground truth and Green points are detection results.

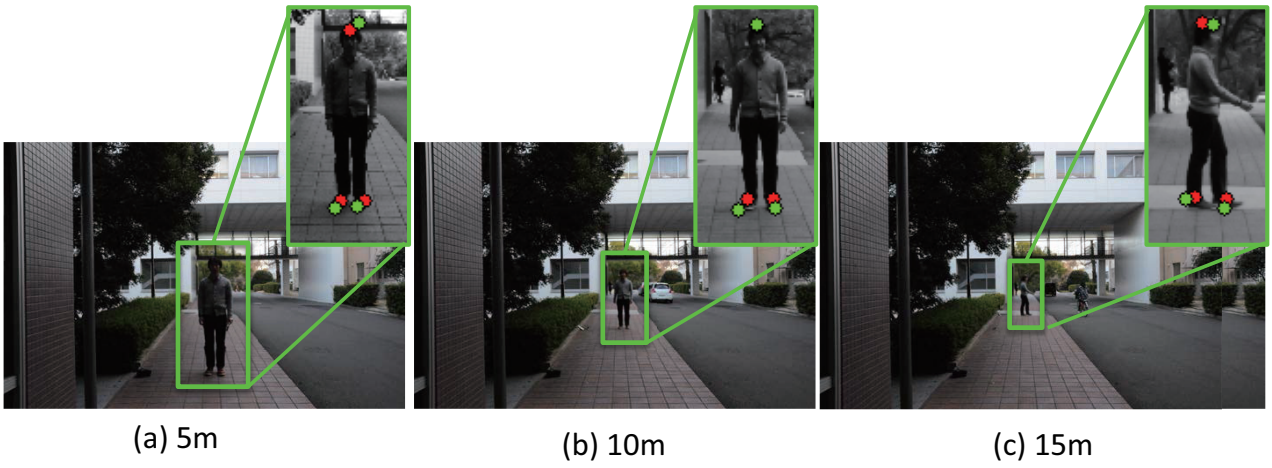

Fig. 7. Examples of distance estimation. It is clearly able to estimate head and leg positions for various pedestrian poses. At the 5m distance, the estimation distance is 4.89m, it is an error of 2.2%.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 1998.

[2] W. Ouyang, X. Wang, Joint Deep Learning for Pedestrian Detection, Computer Vision and Pattern Recognition, 2013.

[3] Z. Zhang, P. Luo, C. Change, T. Xiaoou, Facial Landmark Detection by Deep Multi-task Learning, European Conference on Computer Vision, 2014.

[4] L. Zhao, C. E. Thorpe, Stereo- and Neural Network-Based Pedestrian Detection, IEEE Transactions on Intelligent Transportation Systems, Vol.1, No.3, pp.148-154, 2000.

[5] N.Dalal, B. Triggs, Histograms of oriented gradients for human detection, Computer Vision and Pattern Recognition, 2005.

[6] X. Wang, T. X. Han, S. Yan, An HOG-LBP Human Detection with Partial Occlusion, International Conference on Computer Vision, 2009.

[7] J. Marin, D. Vazquez, A. Lopez, J. Amores, B. Leibe, Random Forests of Local Experts for Pedestrian Detection, International Conference on Computer Vision, 2012.

[8] W. Nam, B. Han, J. H. Han, Improving Object Localization Using Macrofeature Layout Selection, International Conference on Computer Vision Workshop on Visual Surveillance, 2011.

[9] P. Felzenszwalb, D. McAllester, D. Ramaman, A Discriminatively Trained, Multiscale, Deformable Part Model, Computer Vision and Pattern Recognition 2008.

[10] P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. LeCun, Pedestrian Detection with Unsupervised Multi-stage Feature Learning, Computer Vision and Pattern Recognition, pp.3626-3633, 2013.

[11] T. Kishino, Sun Zhe, R. Micheletto, A Fast and Precise HOG-Adaboost Based Visual Support System Capable to Recognize Pedestrian and Estimate Their Distance, Lecture Notes in Computer Science, Vol. 8158, pp.20-29, 2013.

[12] I. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Couville, Y. Bengio, Maxout Network, International Conference on Machine Learning, pp.1319-1327, 2013.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, S. Ilya, R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, Clinical Orthopaedics and Related Research, vol. abs/1207.0, 2012.