https://doi.org/10.20965/jrm.2017.p0649

Paper:

Autonomous Mobile Robot Searching for Persons with Specific Clothing on Urban Walkway

Ryohsuke Mitsudome, Hisashi Date, Azumi Suzuki, Takashi Tsubouchi, and Akihisa Ohya

University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan
E-mail: mitsudome-r@roboken.iit.tsukuba.ac.jp
[Received March 8, 2017; accepted June 4, 2017]

In order for a robot to provide service in a real world environment, it has to navigate safely and recognize the surroundings. We have participated in Tsukuba Challenge to develop a robot with robust navigation and accurate object recognition capabilities. To achieve navigation, we have introduced the ROS packages, and the robot was able to navigate without major collisions throughout the challenge. For object recognition, we used both a laser scanner and camera to recognize a person in specific clothing, in real time and with high accuracy. In this paper, we evaluate the accuracy of recognition and discuss how it can be improved.

Keywords: mobile robot, object recognition, neural network, Tsukuba Challenge

1. Introduction

Tsukuba Challenge is an open experiment for autonomous mobile robots in outdoor environments. The robots are required to travel more than 2 km and search persons wearing specific clothes.

In previous Tsukuba Challenges, some teams proposed target-searching methods based on point cloud information or on camera images [1–3]. The identification accuracy of the method of Akimoto et al. [1] was 92.3%. The robots of Nomatsu et al. [2] and of Eguchi et al. [3] missed several search targets, and their identification accuracy could be further improved.

In recent years, an identifier using a convolutional neural network (CNN) has produced good results in image recognition [4, 5]. Neither of the groups in Tsukuba Challenge of 2014, or before that year, employed the CNN. The authors began using the CNN to identify search targets in Tsukuba 2015.

Most image-recognition technologies use input images mostly consisting of a foreground scene. However, in Tsukuba Challenge, the images acquired by the robots contain various objects, and additional techniques would be necessary to identify a single object located in a part of the image. Some CNNs assume an area where the object could be located and recognize it, but most of them run in real time and a GPU needs to be installed on the robots [6, 7]. In addition, because it is necessary to specify not only labels but also object areas as a train set, the workload to create the train set is larger than that of a simple identifier which only makes object recognition.

We therefore developed a system that could perform object recognition, at high speed and with high accuracy, by processing the information data from laser sensors and cameras in a step by step manner. First, a point cloud, obtained from the laser sensors, is segmented, dividing it into point clouds of individual objects. Next, the recognition targets are narrowed down according to the physical size of the objects. Because the point clouds indicate the position and size of objects more directly than images, an object can be detected at a lower computational cost than when using a CNN, which estimates a foreground scene from an image. Therefore, in order to achieve high accuracy, the image area where the object is located is extracted from the image and the CNN is used to identify the object.

This study is intended at realizing a system that allows a robot to travel in a city making highly accurate recognition of objects lying in the surrounding environment. We participated in Tsukuba Challenge to verify the validity of the developed system. In that challenge, a system is considered to have sufficient performance if it could autonomously navigate and reach the goal, successfully detecting the four search targets.

In Tsukuba Challenge 2015, during the learning process of the CNN, the system accuracy to detect target objects was 99.5%. However, in the final run of Tsukuba Challenge 2015, the robot detected wrong objects as search targets, and approached them without reaching the goal. Even if it had reached the goal, the wrong recognition could have caused an unexpected movement of the robot and endangered the people around. Thus, it was found in Tsukuba Challenge 2015 that extremely high accuracy in object recognition is required for the robot to not only recognize the object but also take action based on the recognition result.

After a full review of this failure, we improved the robot's recognition ability for the final run of Tsukuba Challenge 2016. In this paper, we introduce and discuss the improvement and its result. Nevertheless, it was found in the final run of Tsukuba Challenge 2016 that the im-

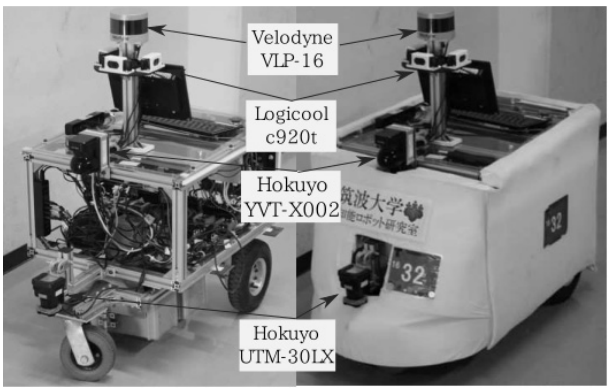

Fig. 1. Appearance of Kerberos without (left) and with (right) external cover.

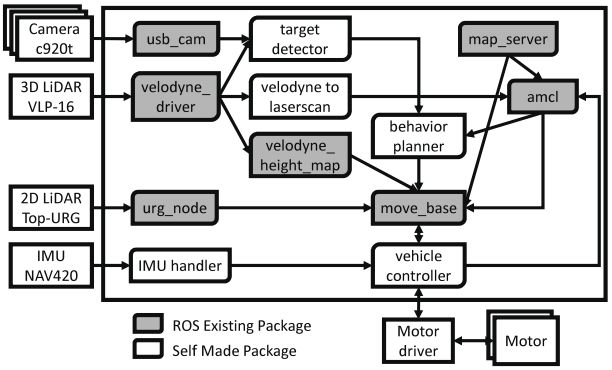

Fig. 2. Software configuration. The white color shows existing packages, and gray shows those created by our group.

proved recognition system was not sufficiently accurate. Therefore, we made an additional improvement of the recognition system, which is also explained in this paper.

In Section 2, we explain the autonomous navigation system of the robot. In Section 3, the procedure of the object recognition algorithm is explained. The result of the robot's performance in Tsukuba Challenge 2016 is shown in Section 4, and the improvement in its object-recognition accuracy is included in Section 5. Section 6 presents a summary and discusses problems to address in the future.

2. Autonomous Navigation System

2.1. Hardware Configuration

Figure 1 shows the appearance of our developed robot, Kerberos. Each rear wheel was equipped with a DC motor and a differential steering system was used to control the movement of the robot. A urethane board with waterproof cloth covered the exterior of the robot to enhance its water resistance.

An internal sensor was used, with motor encoder and NAV420 (Crossbow) IMU, and the translational motion distance obtained from the encoder and the azimuth angle obtained from the IMU were used to calculate the odometry quantities. As external sensor, a VLP-16 (Velodyne) was attached at the height of about 1 m, with three web cameras of c920t (Logicool) directly under the VLP-16, and a UTM-30LX (TOP-URG) (Hokuyo) laser scanner at the height of 0.3 m in the front and rear sides. The horizontal view-field angle of each camera was about 70°. The wide horizontal view-field angle was realized by placing the cameras with angle intervals of 90°. Although the c920t cameras were able to shoot images as large as 1920×1080 pixels, the resolution of each camera was reduced to 854×480 pixels to suppress communication load on the USB. On the front side of the robot, a YVT-X002 (Hokuyo) 3D laser scanner was mounted upside down for securely detecting objects located near the front road surface at various heights.

As power supply, two lithium-ion batteries originally made for electric-assist bicycles were used, with the voltage transformed and connected in parallel. Because each of them could perform hot swapping, the batteries could be replaced without stopping the working program.

A barebone kit (Shuttle) with an Intel Core i7-6700 processor was used as PC.

2.2. Software Configuration

Figure 2 shows the entire configuration of the software. The robot operating system (ROS) [8] was used for the implementation of the software. Readily available packages were used for the sensor driver, self-localization, and map creation, while the software for point cloud processing and image recognition was created by the authors. Other libraries, such as OpenCV for image deformation and Caffe for CNN recognition, were used [9]. Our developed YP-Spur was also used to calculate the rotation speed of each motor from the translational speed and rotation angular speed of the robot.

2.3. Autonomous Navigation

The autonomous navigation of the robot was mostly realized by combining existing packages. The slam_gmapping package was used for map creation, the amcl package for self-localization, and move_base of the navigation package for action planning.

Maps were created from the odometry of the manual travelling performed in advance and the log data of the VLP-16. The odometry was calculated based on the distance obtained from the rotation of the wheels and the azimuth angle obtained from the IMU to reduce error from tire slipping. Lasers close to horizontal were extracted from the sixteen lasers of the VLP-16 to create a two-dimensional map, which is shown in **Fig. 3**. The created map was saved as an occupancy grid map, with a 0.05 m grid, and loaded with a map server package during the navigation. The package amcl was used for self-localization during the navigation. As in the map creation, one of the lasers of the VLP-16 was used for input to the amcl.

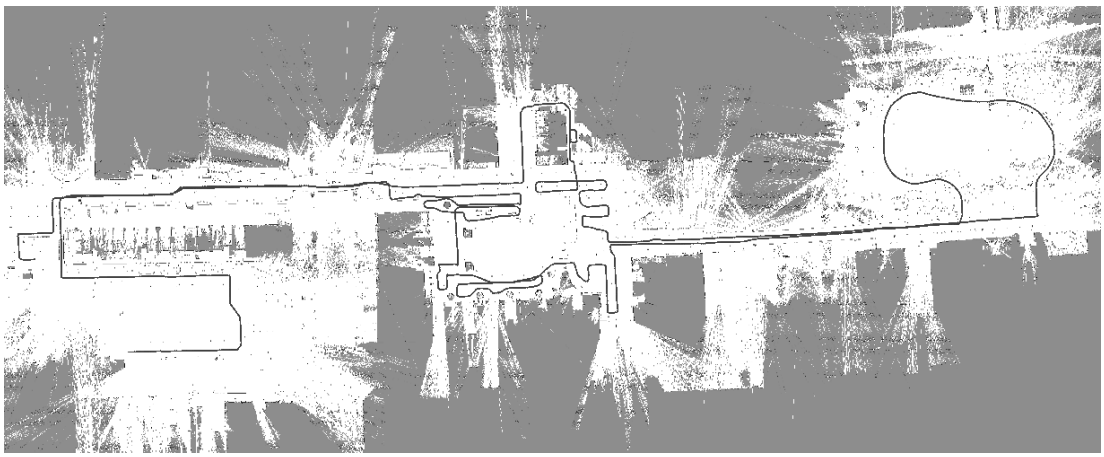

Fig. 3. Occupancy grid map created by slam_gmapping. The white color shows the areas where the robot can travel, black shows the obstacles, and gray shows unknown areas. Arrows indicate the waypoints placed with intervals of 1 m.

Data from both the Top-URG and VLP-16 were used for obstacle detection. A height map package was used to process three-dimensional point cloud data from the VLP-16 for obstacle detection. The height map creates a two-dimensional grid map of the area surrounding the robot and identifies an obstacle by judging whether the difference between the highest and lowest points in each grid exceeds a certain threshold. The obstacle detection was conducted in an area of size 20 m $\times$ 20 m around the robot, with the grid size being 0.1 m and the height threshold being 0.075 m. However, the VLP-16 has a sensor at a high position, and hence, has a blind area. Moreover, it cannot get data of the area within 1 m from the sensor. Thus, we added data from the TOP-URG to allow detection of low-height obstacles near the sensor. Because a slope on which the robot was supposed to travel was sometimes wrongly detected as an obstacle, an elevation angle of 2° was set to the TOP-URG to avoid wrong detection, although this was done somehow on an ad-hoc basis.

If the robot position, obstacles around the robot, and goal are provided, move_base of the navigation package calculates the travelling path and speed of the robot. For the robot position, the calculation result from amcl was provided, whereas for the goal position, waypoints with intervals of 1 m were placed along the travelling path, as shown in **Fig. 3**. However, owing to the specifications of move_base, a travelling path for the waypoints on which the obstacles were laid was not calculated. If an obstacle was detected on a waypoint, this waypoint was skipped and the next one was immediately referred to. If 20 or more waypoints were skipped, the next one was referred to every 15 s.

When a search target was found, the robot approached it by adding a waypoint 0.6 m before the search target. The waypoint of the search target was placed behind the closest waypoint to the search target, among the waypoints that the robot had not reached. When an obstacle was found on a waypoint to the search target, the waypoint was moved to a point on an arc having the center at the search target.

3. Recognition of Search Target

A CNN was used for identification of the images obtained from the cameras to find a search target. To keep high identification capability, objects to be recognized need to be large enough on the image. However, the images from the cameras were occupied mostly by background scenes and the objects to be recognized were not correctly identified. In Ref. [10], the CNN extracted an object area, which increased the load to create a train set.

Therefore, we solved the problem by preprocessing the images based on the point cloud data from the VLP-16. If one can measure the position and size of an object near the robot using a laser scanner, the image area where the object is located can be calculated. We detected possible search targets and extracted their image areas using data from the VLP-16, and the CNN was used to identify search targets in the extracted image areas. The detailed procedure for recognition of the search targets is described below.

3.1. Object Detection Based on Point Cloud

First, the three-dimensional point cloud data obtained from the VLP-16 are segmented, dividing the point cloud into those corresponding to each object. The segmentation method that we used was the same as that in Kikuchi et al. [11], which used a vicinity search based on the VLP-16 data structure, and was faster than segmentation using Kd-tree. Because the VLP-16 captures three-dimensional point cloud data of the shape of surrounding objects by rotating 16 vertically aligned lasers to the horizontal direction, the scanning data obtained in a single scan of the VLP-16 lasers can be processed as panorama distance images, containing elevation and azimuth angles. The three-dimensional distance from a target point to each of eight

pixels next to it is measured and, if the distance is shorter than a threshold value, the pixel is labelled as the one belonging to the same object as the target point.

Next, the obtained segments are tracked. The tracking starts from the centroid of each segment. Calculating the distance from the centroid of the latest segment of an object and that of the second latest segment, we identify the pair of segments that have the shortest inter-distance as the same object to track. Because the movement speed of the centroid of a segment can be calculated once it is successfully tracked, the centroid can be forecasted in the next scanning. Therefore, in the judgment in the second or subsequent tracking, the centroid forecasted from the past segment data is used for stable tracking of a moving object.

Next, the size of a rectangular parallelepiped that contains a segment is calculated. If the size is close to that of a person sitting on a chair and if the rectangular parallelepiped has a sufficiently small velocity, the rectangular area that contains the rectangular parallelepiped is extracted from the camera image. The extraction method will be explained in detail in the next section. Rectangular parallelepipeds, which enclosed segments with depth and width of $0.2 \sim 0.8$ m and height of $1.0 \sim 2.0$ m, and that moved at the speed of 0.5 ms^{-1} or lower, were extracted. This size was chosen in order to be sufficiently large, because the upper and lower bodies of a person sitting on a chair were sometimes classified in different segments.

3.2. Extraction of Object from Image

To extract an object area from a camera image, the apexes of the rectangular parallelepiped that contains the segment of the object were mapped onto the image.

When the focal length of the camera is known and the camera follows a pinhole camera model, the correspondence between a three-dimensional coordinate point and a pixel on the image can be geometrically determined. The focal length was calculated with a USB cam package, which identifies internal parameters of the camera using the method of Zhang et al. [12]. The three-dimensional coordinates of the apexes of the rectangular parallelepiped were converted to the camera image coordinates and a rectangle with the mapped apexes was extracted.

However, it is difficult to accurately measure the relative positions of the camera and 3D LiDAR, and the projection of a three dimensional point onto a camera image contains an error. The image error was the largest, as large as 7 pixels, near the edge of the image because of the lack of distortion correction of the lens. We therefore extracted a little larger area to contain the whole body of the object. The margin of the extracted image is set to be proportional to the area of the extracted image, in such a way that the ratio of the foreground area to the background area does not significantly change, even with the extracted image of a distant object.

Figure 4 shows an example of the segmented point clouds, original image, and extracted image areas. The extraction was conducted every 0.1 s, which was the same period as the sensor's image acquisition period.

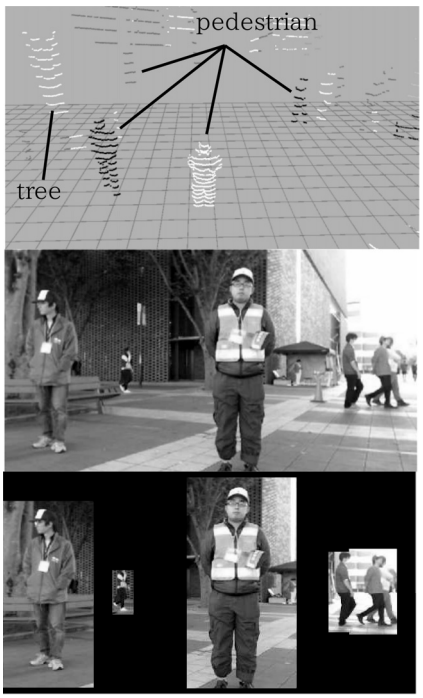

Fig. 4. Point cloud segmentation result (upper), original image (middle), and extracted possible search targets (lower). The point clouds are colored differently depending on the segments.

3.3. Identification by CNN

The learning of the CNN model was performed offline in advance by using DIGITS. The learning of the network was performed with random initial parameters. For the learning, a GPU GeForce GTX TITAN X (RAM 12 GB) was used, and the solver was set to a stochastic gradient descent, the initial learning ratio to 0.01, and the number of epochs to 30. The learning took about 1.5 hours.

The train set consisted of images of the trial run and final run of the last Tsukuba Challenge and the trial run of this year's Tsukuba Challenge. The train set contained images having various weather conditions, and the brightness and details are shown in **Table 1**. The amount of precipitation in **Table 1** was based on the 10-minute-period weather data from the Meteorological Agency and the sunlight was based on the hourly record from the agency. In **Table 1**, 10% of the images were used as test data and 90% as train set.

Recognition on each image extracted by the CNN, for which the learning had been made, was conducted during navigation of the robot to find whether it contained a search target. Because the robot did not have a GPU, only the Intel Core i7-6700 CPU was used for the recognition.

3.4. Changes from FY2015

The following four changes were made during the period from FY2015 to FY2016:

- Change of network.
- Increase of train set.

Table 1. Description of train set.

Date	# of target images	# of non-target images	Weather	Precipitation [mm]	Sunlight [MJm^{-2}]
2015/11/03	5445	113,462	Fine	–	1.90
2015/11/08	317	53,478	Rain	2.0	0.09
2016/09/22	308	160	Rain	0.5	0.33
2016/11/05	301	281	Fine	–	2.07
Total	6371	167,381	–	–	–

- Decrease of the number of recognition operations of a single object.
- Change in number of judgment operations of search targets.

The first change was made to the CNN model. We used AlexNet in FY2015 and GoogLeNet [10] in FY2016. This was because GoogLeNet achieved a higher score than AlexNet in the image recognition competition ILSVRC and because GoogLeNet was one of the standard models of DIGITS, and hence, could be easily used.

In previous studies, the authors confirmed that the identification accuracy could be increased even when smaller images are used for GoogLeNet [13]. Thus, to reduce the computational cost, we changed the input images of GoogLeNet from 224×224 pixels to 64×64 pixels. After a learning process of AlexNet and GoogLeNet with the same learning data and hyper parameters, AlexNet and GoogLeNet were used to perform image recognition for about 170,000 images acquired on November 6, 2016 to measure the percentage of correct answers, which was found to be slightly higher for GoogLeNet than for AlexNet. However, the difference was within 0.0001% and the change of network did not significantly affect the performance.

The second change was to increase the amount of the train set. In FY2015, only images extracted from a trial run on November 3, 2015 were used for the train set. In FY2016, images extracted from a trial run in Tsukuba Challenge FY2016 were added. The images extracted on November 3 and 8, 2015 were manually labeled. The CNN learning was performed with these images and then the images extracted from the FY2016 data that the CNN identified as search targets were manually labelled and added to the train set. The number of images in the train set finally increased from 118,907 to 173,752, and consisted of 6,371 images of search targets and 167,381 other images. **Fig. 5** shows examples of the train images of the search targets and others. This increased the percentage of correct answers by 0.1 points, and thus, the percentage of correct answers reached 99.6%.

The third change was the enhancement of the identification speed. As in Ref. [13], the mean identification time per image with a size of 64×64 pixels is about 0.06 s with the Intel Core i7-6700 CPU. This indicates that, if only one object is in the image, real time processing is feasible because the image extraction cycle period is 0.1 s. However, a single camera image often contains multiple objects. We took the following measure to avoid a situation where there are too many images to be processed by the CNN identification processing. The number of times that each object was recognized was measured from the tracking data of the point cloud segments, and the probability of performing recognition processing was changed in accordance with that number, thus avoiding too many recognition operations of a single object. In other words, if the number of times that an object was recognized was n, then the recognition operation was performed with the probability of $1/(n+1)$.

Fig. 5. Example of train set of CNN. Upper images show the search targets and the lower show others.

The last change was made in the post-processing of the CNN identification. As a countermeasure against erroneous recognition of the CNN, the system was set to approach an object only when it was judged as search target three or more times at the same position.

4. Result of Final Run

4.1. Result of Final Run and Discussion

In the final run, the robot had to drop out at a position 530 m from the start after detecting one search target. **Fig. 6** shows the travelling path of the final run. The path after the robot entered in the search area where search targets were located was significantly deviated from the waypoints in **Fig. 3**, because the robot wrongly recognized ordinary people 15 times and approached them, thus deviating from the waypoints. **Fig. 7** shows an example of one of the objects incorrectly detected. Most of the erro-

Fig. 6. Travelling path in the final run. The map of the Central Park is omitted as the robot stopped travelling in the search area.

Fig. 7. Search targets and example of wrong recognition (false positive).

neously detected objects in the images had the same colors as the search targets, namely, orange, green, or blue. Finally, the robot erroneously recognized ordinary people in the no-entry area as search targets, entered the area, and hence, dropped out.

The following three were considered as causes for the drop-out.

- Identification accuracy of the CNN.
- Threshold setting for search target recognition.
- Erroneous designation of no-entry area.

The first cause was related to the fact that the last year's network was designated in the final run by mistake. Last year's network uses the smaller train set of FY2015, and is less accurate than the above-designed network. This caused many objects to be wrongly recognized as search targets. The travelling log data indicated that six incorrect recognitions would have been made, even if the network model of FY2016 had been used. The second cause was closely related to the first one. To reduce the influence of the CNN incorrect recognition, the robot was set in such a way that it approaches an object only when the object is recognized as search target three times at the same position. However, this threshold was implemented a day before the final run, and hence, there was not sufficient examination of the threshold. We set the threshold to be relatively low because the rules of Tsukuba Challenge did not charge a penalty to the erroneous recognition of objects, and this setting of the threshold caused many incorrect recognitions. In particular, the robot wrongly recognized an ordinary person in the no-entry area as a search target, which was one of the direct causes of the dropout.

The third cause was a human error. Part of the search area was designated as a no-entry area, for safety reasons. Pylons were placed at constant intervals around the no-entry area. Because there was no signs or other indications on the boundary that could be detected by a sensor, the no-entry area was manually set as additional information on the map created by the slam_gmapping. However, we did not check the actual area in the final run and did not notice that our designated no-entry area was smaller than the actual one, which caused entry of the robot into the no-entry area.

4.2. Goal Attainment Level

The attainment level of a robot system that recognizes an object while travelling along a path, which was the primary objective of the present study, is here discussed based on the results of the final run and the cause of the dropout.

The mean time from the moment the laser sensor extracted an image to when the CNN finished the identification process was about 0.15 s. The search target recognition took as short as about 0.45 s, since the final judgment

was made when the CNN identified three times the same object as search object. This was fast enough for the mobile robot travelling along the path of Tsukuba Challenge at the speed of 4 km/h or lower. Only the Intel Core i7-6700 processor was used for object recognition, indicating that a robot without GPU could recognize objects at sufficiently high speed.

However, the recognition accuracy is not enough and needs to be improved, because the robot dropped out owing to incorrect recognition.

5. Improvement of Recognition Accuracy

We made 15 incorrect recognitions in the final run of Tsukuba Challenge, and found from the log data that 6 incorrect recognitions would have been made even if the CNN model of FY2016 had been used. Incorrect recognition causes apparently meaningless actions of the robot, which disturbs people in the surrounding area. Therefore, it is important to enhance the recognition accuracy in the future.

For the improvement of recognition accuracy, we revised the algorithm of the post-processing of the accumulated judgment results of the CNN, and enhanced the recognition accuracy of the CNN itself.

5.1. Judgment Algorithm Improvement Using Tracking Information

It is extremely difficult to eliminate all incorrect recognitions of the CNN identifier. So far, we had set the rule that, when an object was identified as a search object three times during the tracking of the point cloud data objects, the identification was assumed a correct one in the final judgment. However, this rule largely depends on the number of extracted images of the object and is therefore affected by the image extraction period, robot travelling path, or travelling speed. To circumvent the problem, we set a new rule, according to which the final judgment was made not from the number of times that an object was recognized as a search target but from the ratio of this number to the total number of times that an object was recognized. For example, when the ratio of the number of times that an object is recognized as a search target is 0.5 or higher, the object is judged as a search target. This results in a judgment that does not depend on the number of extracted images of the object.

The condition on the existence of a threshold to prevent incorrect recognition with the present method is expressed by the following inequality.

$$\min_{i \in T} \left(\frac{t_i}{n_i} \right) > \max_{i \in \bar{T}} \left(\frac{t_i}{n_i} \right) \quad \ldots\ldots\ldots\ldots (1)$$

T, $\bar{T}$ are a set of search targets and a set of the others, respectively. t_i is the number of times that the object i was identified as search target, and n_i is the number of times that the object i was recognized. The left hand side of the inequality shows the "minimum value of the percentage of the number of times that a search target was identified as search target," and the right hand side shows the "maximum value of the percentage of the number of times that an object other than the search targets was identified as search target."

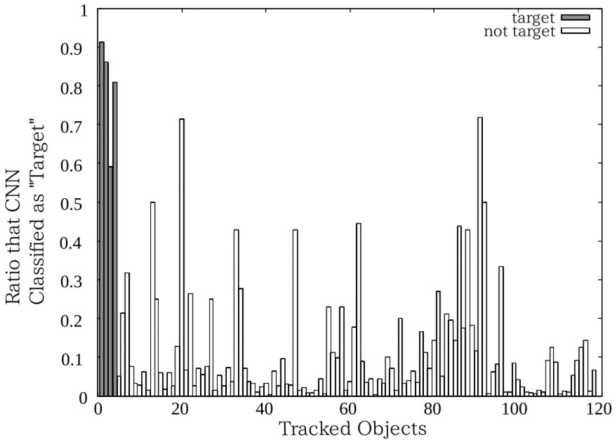

Fig. 8. Ratio that CNN classified objects as search targets. Gray shows data of the search targets and white shows data of other objects. An appropriate threshold cannot be designated because some objects other than the search targets are repeatedly recognized as search targets.

Table 2. Recognition result of all extracted images.

	True	False
Target (Positive)	491	705
Not Target (Negative)	175,169	277

It is checked whether an appropriate threshold could be found with the CNN for which learning was conducted this fiscal year. Images were extracted and saved every 0.1 s during the single-speed playback of the travelling log data of November 6, 2016, and the object recognition operations were conducted for all of these extracted images. Since we had only the travelling data of the final run of Tsukuba Challenge, collected until the first search target was found, we used log data of the robot travelling in the search area, collected after the final run. A total of 872 images of the search targets and 175,660 other images were extracted. However, the images where the search targets appeared with a small size in the background were eliminated from those to be analyzed.

Figure 8 shows the percentage of the number of times that an object was recognized by the CNN as a search target at least once. From the figure, one can see that there was an object, other than the search targets, that was wrongly recognized in a repetitive way, which did not satisfy the inequality (1), indicating the absence of a threshold to find all search targets with no wrong recognition. Therefore, the identification accuracy of the CNN needs to be enhanced.

For reference, we show the detailed results of the identification in **Table 2**. "Positive" indicates the search targets and "Negative" the others.

Fig. 9. Image recognized as target (left) and that wrongly recognized as object other than target (right). The left image's resolution is low as it is extracted from a distant area image.

5.2. Diversification of Train Set

In Section 5.1, we found that the network did not have a sufficiently high identification capability. Because the images that looked with almost the same characteristics were sometimes judged differently (**Fig. 9**), the learning of versatile characteristics failed and over-learning occurred. An ordinary countermeasure against over-learning is to extend data by parallel translation or color change of images [4, 14]. We improved the train set in the following three points.

The first improvement is to increase the number of kinds of the train set. The train set of the previous identifiers contain data mostly from FY2015 and a small amount of data from FY2016, and hence, contains a small number of images of the objects on a path newly added as search area this fiscal year. Therefore, we decided to include not only the incorrectly recognized images but also all the extracted images into the train set. In addition, because the images extracted every 0.1 s looked similar to each other, learning about local characteristics was repeatedly performed, causing over-learning. Of the images, 10% were randomly chosen from the extracted images to create image data with virtual intervals of 1 s. **Table 3** shows the details of the train set.

Next, the following data augmentation was made. Each one of the image edges was randomly trimmed by several pixels and the brightness was changed by gamma correction. It was expected that the trimming could make the images robust against parallel translation and the change in brightness could reduce the influence of sunlight.

Lastly, in the previous train set, the ratio of the images of objects other than the search targets to the images of the search targets was large, and hence, learning of the characteristics of the search targets and that of the objects other than the search targets may not be performed under similar conditions. Therefore, we applied data augmentation to the images of the search targets to make their number in the train set equal to that of the objects other than the search targets.

5.3. Identification Performance After Improvement

To investigate the effects of the above three improvements, learning was conducted under different conditions of the following four patterns, and the resulting performance was analyzed.

1. No data augmentation.
2. Data augmentation only for search targets.
3. Augmentation of all data.
4. Data augmentation for search targets after augmentation of all data.

In the patterns 2 and 4, the data augmentation is performed to make the number of the images of the search targets equal to that of the objects other than the search targets.

Table 4 shows details of the data and result of the object identification. Images extracted on November 6, 2016 not contained in the train set were used for the evaluation. With the increased data set, one can see that every pattern significantly improved the precision and recall of the CNN. In the data augmentation of patterns 2 to 4, the recognition accuracy increased, indicating a positive effect of the data augmentation.

Figure 10 shows the percentage of the number of times that the CNN recognized an object as search target under the condition of the pattern 4. One can see from **Fig. 10** that the improved CNN tended to repeatedly recognize the search targets as search target and recognize the objects other than the search targets incorrectly in some cases, but in most cases correctly. Therefore, if a threshold is set to a value between 0.1 and 0.6, the objects that the CNN erroneously recognized as search targets could be correctly recognized after repeated judgments. In the same way, an appropriate threshold can be set for patterns 1 to 4, and then the search targets could be identified without incorrect recognition in the playback of the log data of the final run.

5.4. Discussion

The accuracy of the CNN was improved by 0.23% in pattern 1, although the number of training images decreased compared with that before the improvement of the CNN. In particular, the precision was improved considerably, by 34.8%, and the false positives, cause for incorrect recognition, significantly decreased. It was therefore found that not the number but the diversity of the images in the train set could affect the performance of the CNN.

If we compare pattern 1 with pattern 2, or pattern 3 with pattern 4, the increase in the number of the training images increased the recall but decreased the precision. This could occur because data augmentation only for the search targets made the characteristics of the search targets more general for learning, and thus, made it easier to identify various objects as search targets. From the above, one can conclude that, to some extent, recall and precision are in a trade-off relation, and data augmentation does not

Table 3. Description of improved train set.

Date	Number of target images	Number of non-target images	Weather	Precipitation [mm]	Sunlight [MJm^{-2}]
2015/09/26	71	9070	Fine	–	1.42
2015/11/03	544	1134	Fine	–	1.90
2015/11/06	58	14237	Fine	–	1.92
2015/11/07	5	3091	Fine	–	0.84
2015/11/08	31	5348	Rain	2.0	0.09
2016/09/22	111	5195	Rain	0.5	0.33
2016/10/17	4	8379	Rain	0.0	0.24
2016/11/04	13	19271	Fine	–	2.07
2016/11/05	30	28	Fine	–	2.07
Total	867	88385	–	–	–

Table 4. Classification result after improvements.

Classifier	Data augmentation (not target)	Data augmentation (target)	Number of training images	Accuracy	precision	Recall
2016 model	–	–	173752	99.61	58.3	78.3
Pattern 1	–	–	89,252	99.84	93.1	74.1
Pattern 2	–	102 times	176,819	99.85	86.5	82.1
Pattern 3	5 times	5 times	446,260	99.85	98.4	71.7
Pattern 4	5 times	510 times	884,095	99.86	98.0	74.0

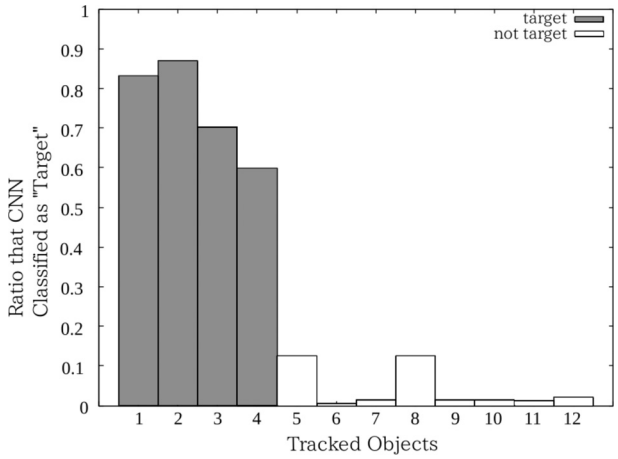

Fig. 10. Ratio that CNN classified objects as search targets (after improvement). Gray shows data of the search targets and white shows data of other objects. The difference between the search targets and objects other than the search targets is large and a threshold in between 0.1 and 0.6 can be designated.

Fig. 11. Examples of objects that CNN wrongly recognized as search targets even after the improvement.

always contribute to the improvement in recognition accuracy; thus, an appropriate pattern of the train set data augmentation has to be chosen.

However, the increase rate in recall was smaller than the increase rate in precision for every pattern. This could happen because the given data comprised only eight days and the number of the search targets was only 32. The recall could be increased by using more search target images.

Lastly, samples of the images that the CNN erroneously recognized as search targets in pattern 4, which had the highest recognition accuracy, are shown in **Fig. 11**. Objects in 12 out of 13 false positive images were passers-by, and of those 7 out of the 12 wore clothes of similar color to those the search targets wore (two central images in **Fig. 11**). Those in the remaining 5 of the 12 wore clothes of apparently different color from those the search targets wore, but were incorrectly recognized (right side in **Fig. 11**).

Incorrect recognition of objects that had no similarity to the search targets indicates vulnerability of the CNN against unknown data. The disadvantage of the CNN that it tends to erroneously recognize images not included in

the train set, and geometrical patterns for high confidence has been reported [15–17]. Because, if extracted on a different day or time, the images of passers-by are not the same as those in the train set, they are likely to be incorrectly recognized.

A CNN extracts and learns characteristics from the train set. However, in the search target recognition of this experiment, the classes other than the search targets are diverse and there are no characteristics typical to objects, other than those of the search targets. Therefore, a CNN learning of the characteristics is practically impossible. In this case, the learning operation is easily made with the training images but not with unknown data. Therefore, the CNN, even that improved in this study, may still incorrectly recognize objects in a completely different situation, for example if a different path is used in Tsukuba Challenge of next year.

It was already confirmed that, if objects categorized as false positives were added in the train set, the occurrence of incorrect recognition of the objects became less frequent, increasing the overall accuracy of the recognition. Therefore, the reliability of the system could be enhanced by adding more training images to the train set. However, it is obvious that a larger train set results in a higher achievement, and hence, we cannot conclude that more robust characteristics can now be captured. The accuracy improved by adding false positive images may reach a plateau if the addition is repeated. A detailed investigation should be done in the future.

6. Conclusion

This study was intended at achieving high-speed, highly accurate object recognition. In this paper, we explained the autonomous navigation of the developed robot and CNN-based object recognition method, and presented the obtained results at Tsukuba Challenge 2016, in which we participated for verification of the applicability of the developed system.

Real-time recognition was realized without using a GPU, by making the size of the input images smaller and limiting the recognition frequency of the same object.

The followings conclusions were drawn from the recognition improvement:

- A higher degree of accuracy is achieved by improving the train set quality than by improving the CNN model.
- A higher degree of accuracy is achieved by providing a large number of similar images to the train set than by providing it with different images.

During the period from FY2015 to FY2016, we changed the CNN model and increased the train set, but the changes were not sufficient to remove incorrect recognition. The analysis of the result of the final run showed that the CNN recognition accuracy increased from 99.61% to 99.86% by increasing the types of images in the train set and by data augmentation. It was also confirmed that all the search targets could be successfully identified, without incorrect recognition, by using, in addition, the tracking information. The recognition capability of the CNN largely depends on the quality of the train set, and objects not included in train set may be wrongly recognized. In the future, we will investigate the versatility of our developed identifier at next year's Tsukuba Challenge.

References:

[1] S. Akimoto, T. Takahashi, M. Suzuki, Y. Arai, and S. Aoyagi, "Human Detection by Fourier Descriptors and Fuzzy Color Histograms with Fuzzy c-Means Method," J. of Robotics and Mechatronics, Vol.28, No.4, pp. 491-499, 2016.

[2] M. Nomatsu, Y. Suganuma, Y. Yui, and Y. Uchimura, "Development of an Autonomous Mobile Robot with Self-Localization and Searching Target in a Real Environment," J. of Robotics and Mechatronics, Vol.27, No.4, pp. 356-364, 2015.

[3] J. Eguchi and K. Ozaki, "Development of the Autonomous Mobile Robot for Target-Searching in Urban Areas in the Tsukuba Challenge 2013," J. of Robotics and Mechatronics, Vol.26, No.2, pp. 166-176, 2014.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems, Vol.25, pp. 1097-1105, 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Network," Advances in Neural Information Processing Systems, Vol.28. pp. 91-99, 2015.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 779-788, 2016.

[8] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," ICRA Workshop on Open Source Software, 2009.

[9] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," Proc. of the 22nd ACM Int. Conf. on Multimedia, pp. 675-678, 2014.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1-9, 2015.

[11] J. Kikuchi, H. Date, S. Ohkawa, Y. Takita, and K. Kobayashi, "Detection of Person in a Seat Using 3D LiDAR," The 31st Annual Conf. of Robotics Society of Japan, 1E3-03, 2013.

[12] Z. Zhang, "A Flexible New Technique for Camera Calibration," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.22, No.11, pp. 1330-1334, 2000.

[13] R. Mitsudome, H. Date, and T. Tsubouchi, "Fast CNN Image Recognition Using Multi-Layered Laser Scanner," The 34st Annual Conf. of Robotics Society of Japan, 2Z1-04, 2016.

[14] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 3642-3649, 2012.

[15] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks," The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 2574-2582, 2016.

[16] A. Nguyen, J. Yosinski, and J. Clune, "Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images," The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 427-436, 2015.

[17] A. Bendale and T. Boult, "Towards Open World Recognition," IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), pp. 1563-1572, 2016.

Name:
Ryohsuke Mitsudome

Affiliation:
Master Student, Department of Intelligent Interaction Technologies, University of Tsukuba

Address:
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan

Brief Biographical History:
2016- Master Student, University of Tsukuba

Membership in Academic Societies:
• The Robotics Society of Japan (RSJ)

Name:
Hisashi Date

Affiliation:
Associate Professor, Faculty of Engineering, Information and Systems, University of Tsukuba

Address:
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan

Brief Biographical History:
2003-2013 Research Associate, Department of Computer Science, National Defense Academy of Japan
2005-2006 Visiting Associate, Department of Mechanical Engineering, California Institute of Technology
2013-2015 Lecturer, Department of Computer Science, National Defense Academy of Japan
2015- Associate Professor, Faculty of Engineering, Information and Systems, University of Tsukuba

Main Works:
• "Recognition Method Applied to Smart Dump 9 Using Multi-Beam 3D LiDAR for the Tsukuba Challenge," J. of Robotics and Mechatronics, Vol.28, No.4, pp. 451-460, Oct. 2016.
• "High Precision Localization of Mobile Robot Using LIDAR Intensity of Surface," Trans. of the Japan Society of Mechanical Engineers, Series C, Vol.79, No.806, pp. 3389-3398, Oct. 2013.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)
• The Japan Society of Mechanical Engineering (JSME)
• The Robotics Society of Japan (RSJ)

Name:
Azumi Suzuki

Affiliation:
Personnel Department, Honda Motor Co., Ltd

Address:
661-13 Horinochi, Hachioji-shi, Tokyo 192-0355, Japan

Brief Biographical History:
2017- Honda Motor Co., Ltd.

Name:
Takashi Tsubouchi

Affiliation:
Professor, Faculty of Engineering, Information and Systems, University of Tsukuba

Address:
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan

Brief Biographical History:
1988 Received Ph.D. from Department of Electronics and Information Engineering, University of Tsukuba
1989-1993 Assistant Professor, Utsunomiya University and The University of Tokyo
1994-2006 Lecturer and Associate Professor, University of Tsukuba
2006- Professor, University of Tsukuba

Main Works:
• H. Kawanishi, Y. Hara, T. Tsubouchi et al., "Calibration of Lens Distortion for Super-Wide-Angle Stereo Vision," 2015 IEEE Int. Conf. on Automation Science and Engineering, pp. 843-848, Aug. 2015.
• Y. Hara, S. Bando, T. Tsubouchi, and A. Oshima, "6DOF Iterative Closest Point Matching Considering A Priori with Maximum A Posteriori Estimation," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2013), pp. 4172-4179, Nov. 2013.
• S. Bando, T. Tsubouchi, and S. Yuta, "Scan matching method using projection in dominant direction of indoor environment," Advanced Robotics, Vol.28, No.18, pp. 1243-1251, 2014.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)
• The Robotics Society of Japan (RSJ)
• The Japan Society of Mechanical Engineers (JSME)

Name:
Akihisa Ohya

Affiliation:
Professor, Division of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba

Address:
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan

Brief Biographical History:
1992-1994 Research Associate, University of Tsukuba
1994-2000 Assistant Professor, University of Tsukuba
1995-1997 Visiting Scholar, Purdue University
2000-2012 Associate Professor, University of Tsukuba
2000-2003 Researcher, PRESTO, JST
2012- Professor, University of Tsukuba

Main Works:
• "Inside Vehicle Inspection System Utilizing a Mobile Robot with LRF Sensor" J. of Robotics and Mechatronics, Vol.26, No.1, pp. 59-67, Feb. 2014.

Membership in Academic Societies:
• The Institute of Electrical and Electronic Engineers (IEEE) Robotics and Automation Society
• The Robotics Society of Japan (RSJ)
• The Society of Instrument and Control Engineers (SICE)
• The Japan Society of Mechanical Engineering (JSME)
• The Institute of Electronics, Information and Communication Engineers (IEICE)
• The Acoustical Society of Japan (ASJ)
• The Japan Society of Ultrasonics in Medicine (JSUM)