


RESEARCH ARTICLE

Open Access



# Distance estimation with 2.5D anchors and its application to robot navigation

Hiroataka Hachiya<sup>1\*</sup> , Yuki Saito<sup>2</sup>, Kazuma Iteya<sup>1</sup>, Masaya Nomura<sup>1</sup> and Takayuki Nakamura<sup>1</sup>

## Abstract

Estimating the distance of a target object from a single image is a challenging task since a large variation in the object appearance makes the regression of the distance difficult. In this paper, to tackle such the challenge, we propose 2.5D anchors which provide the candidate of distances based on a perspective camera model. This candidate is expected to relax the difficulty of the regression model since only the residual from the candidate distance needs to be taken into account. We show the effectiveness of the regression with our proposed anchors, by comparing with ordinary regression methods and state-of-the-art 3D object detection methods, through Pascal 3D+ TV monitor and KITTI car experiments. In addition, we also show an example of practical uses of our proposed method in a real-time system, robot navigation, by integrating with ROS-based simultaneous localization and mapping.

**Keywords:** Deep learning, Monocular camera image, Distance estimation, Navigation

## Introduction

Detecting a target object from an image is an important task and recent deep learning based methods such as Faster R-CNN [1] and YOLO [2] have enormously advanced its performance and speed. However, the location of an object on an image plane provided by object detection methods would not be enough for a real application. Standard approaches to measuring the distance from a monocular camera are to use triangulation over a pair of images captured by the camera moving along with navigation robots [3]. This approach is cost-effective compared with the stereo camera; however, the movement to make the disparity would not be time effective. That is, as for tracking a target object, a robot may be required to detour to make the disparity for measuring the distance and thus it would delay the tracking.

Thus, measuring the distance from a single camera image would be expected in a real application. However, the regression of distance from a single image is prohibitively difficult due to the variation in the object appearance. Figures 1, 2 and 3 depict the examples of the relation between the appearance and distance using

KITTI dataset [4]. Ground truth (GT) bounding boxes (BBs) in Figs. 1 and 2 have similar shapes and sizes, however, the GT distances are largely different, i.e., 12.1 and 8.4 m (see values in green boxes). Meanwhile, the distances in Figs. 1 and 3 are almost same but the appearances of two cars are different. These examples imply that training regression models for the distance estimation would be difficult due to the diverse relationships between appearances, BBs, and distances.

In this paper, to tackle such the challenging task, we propose 2.5D anchors which provide the candidate of distances using *perspective camera model*. This candidate is expected to relax the difficulty of training regression model since only the small residual between GT and the candidate distances have to be taken into account. Using this proposed 2.5D anchor, called *perspective anchor*, we extend one of state-of-the-art object detection method, Faster R-CNN, and show its performance improvement by comparing with ordinary regression methods through experiments with Pascal 3D+ TV monitor dataset. In addition, we show that the performance of our proposed method is well comparable with the state-of-the-art 3D object detection methods [5, 6] over KITTI car dataset. Finally, we show an example of practical uses of our proposed method, on a real-time system, robot navigation, by integrating with simultaneous localization and mapping (SLAM).

\*Correspondence: hhachiya@sys.wakayama-u.ac.jp

<sup>1</sup> Faculty of System Engineering, Wakayama University, 930 Sakaedani, Wakayama-shi 640-8510, Japan

Full list of author information is available at the end of the article



**Fig. 1** Examples of the relation between the appearance of target objects (cars), and those distance in KITTI dataset. The green and red rectangles are the ground truth (GT) and estimated bounding boxes (by our proposed method) respectively. The values in green and red boxes are the GT and estimated distances respectively in meter. The distance of the car on the right hand side is 12.1 m



**Fig. 2** Examples of the relation between the appearance of target objects (cars), and those distance in KITTI dataset. The green and red rectangles are the ground truth and estimated bounding boxes (by our proposed method) respectively. The values in green and red boxes are the GT and estimated distances respectively in a meter. The distance of the car on the left-hand side is 8.4 m



**Fig. 3** Examples of the relation between the appearance of target objects (cars), and those distance in KITTI dataset. The green and red rectangles are the ground truth and estimated bounding boxes (by our proposed method) respectively. The values in green and red boxes are the GT and estimated distances respectively in a meter. The distance of the car on the left-hand side is 11.9 m

## Related works

Related to the distance measurement from a monocular image, 3D object detection methods have been actively studied recently [5–8]. There are mainly two types of approaches for 3D object detection, i.e., model-based and model-free approaches. The model-based approaches [7, 8] prepare a variety of 3D CAD models and make them fit to target objects on the image plane and infer its 3D position and pose. These model-based approaches provide high performance given appropriate 3D CAD models but are limited only to rigid objects, e.g., car and TV monitor—non-rigid object like a human could not be detected. Meanwhile, model-free approaches [5, 6] directly perform the regression of the dimension and orientation of 3D box using good initial deployment of candidate 3D

boxes through *subcategories* [6] and *MultiBin* [5]. These methods do not need CAD models and thus can be more flexibly applied to a variety of objects including a human and an animal. In these 3D box detection approaches, both 2D BB and 3D box are detected and the projection matrix from the 3D box to 2D BB is estimated to obtain the 3D position of the object. However, there are as many as 8 target variables, i.e., 4 for 2D BBs and 4 for 3D box dimension (height, width, and length) and orientation at Y-axis in camera coordinate. Annotating these 8 target variables of 2D BBs and 3D boxes could be expensive since the visual inspection by a human is necessary—especially annotating 3D boxes on a 2D image would be difficult due to unseen parts of the object.

Therefore, in this paper, we propose a *direct* distance estimation method by extending, a 2D object detection method, Faster R-CNN [1] to 2.5D object detection, e.g., 2D BB and distance.

## Faster R-CNN

In this section, we review Faster R-CNN. As shown in Fig. 4, Faster R-CNN [1] consists of four parts: pre-trained CNN (convolutional neural network), RP (region proposal) network, RoI pooling and FC (fully connected) network. For more details, in pre-trained CNN, given an input image, a variety of feature maps are extracted. In RP network, at each pixel of feature maps, the combination of 3-different-shape and 3-different-size 2D anchors are created as candidates of BBs. Then, the RP network selects 2D anchor potentially containing the target object, and regresses the residual of its position i.e.,  $\mathbf{a}_{2D}^i \equiv (x_{\min}^i, y_{\min}^i, x_{\max}^i, y_{\max}^i)^T$  to the target BB  $\mathbf{b}_{2D}^{*i} \equiv (x_{\min}^{*i}, y_{\min}^{*i}, x_{\max}^{*i}, y_{\max}^{*i})^T$ . Based on selected 2D anchors and RoI proposals, feature maps are cropped and converted to the same size feature map by RoI pooling. Finally, in FC layer, for each RoI map, the label of object is classified and BB is regressed again to refine its position.

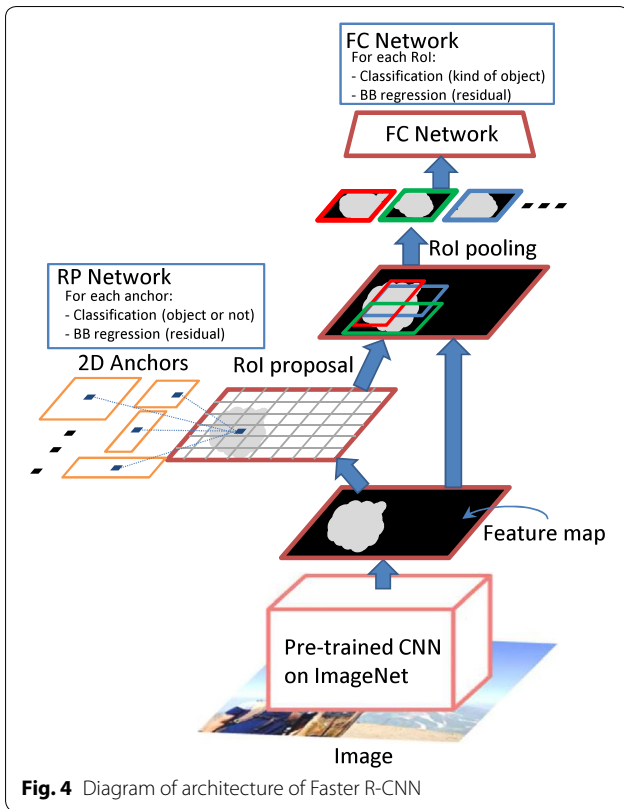
## 2D anchors

A noteworthy mechanism of Faster R-CNN is to use anchors as candidates of BBs—these anchors could cover a variety of BBs for multiple types of objects with various sizes and rotations. With such anchors, the regression problem of the BB can be simplified to the selection of most fitting 2D anchors and the regression of the residual between those 2D anchors and the GT BB.

## 2.5D bounding box estimation

In this section, we extend Faster R-CNN to estimate 2.5D BBs:

$$\mathbf{b}_{2.5D}^{*i} \equiv (x_{\min}^{*i}, y_{\min}^{*i}, x_{\max}^{*i}, y_{\max}^{*i}, z^{*i})^T \quad (1)$$

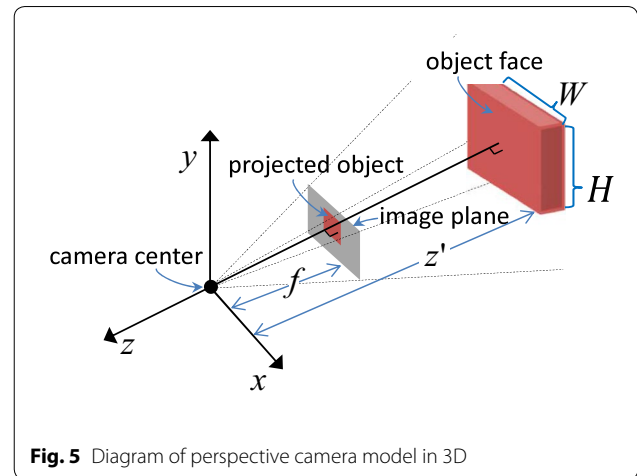


where the distance  $z^{*i}$  from a camera to the target object face is added to the original 2D BB  $\mathbf{b}_{2D}^{*i}$ . To estimate 2.5D BB, we assume to have the training data  $\mathcal{D}$  consisting of pairs of an image  $I^i$  and a 2.5D BB  $\mathbf{b}_{2.5D}^{*i}$  as follows:

$$\mathcal{D} \equiv \{I^i, \mathbf{b}_{2.5D}^{*i}\}_{i=1}^{N_{\text{train}}} \quad (2)$$

where  $N_{\text{train}}$  is the number of training data. This assumption on the data availability would be acceptable since if a laser sensor calibrated with a camera is available in the data collection phase, the distance  $z^{*i}$  can be systematically annotated by using scan data corresponding to the 2D BB annotated by a human as in KITTI dataset [4].

From this observation, one advantage of estimating 2.5D BBs over 3D Box approaches [5, 6] in terms of the distance estimation, is this feasibility of data annotation, i.e., 3D Box approaches need 8 different annotations for 2D BB, 3D box dimension, and orientation. But let us clear the difference in the target applications between 3D approaches and our 2.5D BB approach. The target applications of 3D approach are to localize a variety of target objects e.g., car, bicycle and pedestrian in an arbitrary 3D space and make a 3D visualization of objects like a birds eye's view using computer graphics (e.g., Fig. 4 in [6]). Meanwhile, our target application is a simple distance measurement of specific target objects, like cars



in roads or TV monitors in rooms. That is, it is assumed in our approach, that the target objects would have relatively small variance in size; for example, the case that target objects in a range from miniature cars and real cars would be out of scope.

To perform such 2.5D BB estimation in Faster R-CNN, we extend 2D anchor  $\mathbf{a}_{2D}^i$  to 2.5D anchor  $\mathbf{a}_{2.5D}^i = (\mathbf{a}_{2D}^{iT}, z^i)^T$  which additionally contains a distance candidate  $z^i$ . Similarly to the original Faster R-CNN, the key to success lies on a good design of anchors. To this purpose, we propose the *perspective anchor* based on the perspective camera model.

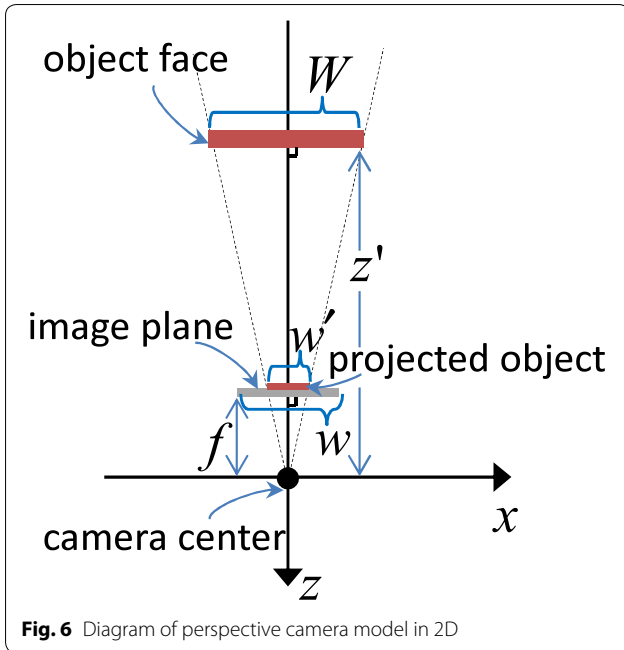
### Perspective camera model

We briefly review *perspective camera model*. As shown in Figs. 5 and 6, the perspective camera model here consists of *image plane*, *projected object*, and *object face* in the camera coordinate system. The distances from the image and the object face to the origin (i.e., camera center) are corresponding to the focal length  $f$  and object distance  $z'$  respectively. With *similar triangles rule*, the following equations are derived

$$w' = \frac{kfW}{z'} \quad (3)$$

$$h' = \frac{kfH}{z'} \quad (4)$$

here  $k$  is the camera parameter for converting the unit from *meter* to *pixel* in the image space.  $W$  and  $H$  are the width and height of the object face, and  $w'$  and  $h'$  are the width and height of its projection onto the image plane. Note that the height  $H$  and  $h'$  are not depicted at Fig. 6



since only widths  $w'$  and  $W$  in this figure are replaced with heights  $h'$  and  $H$ .

Let us multiply this width  $w'$  and height  $h'$  to express the object distance  $z'$  as the ratio between two areas  $WH$  and  $w'h'$  as

$$w'h' = \left(\frac{kf}{z'}\right)^2 WH \tag{5}$$

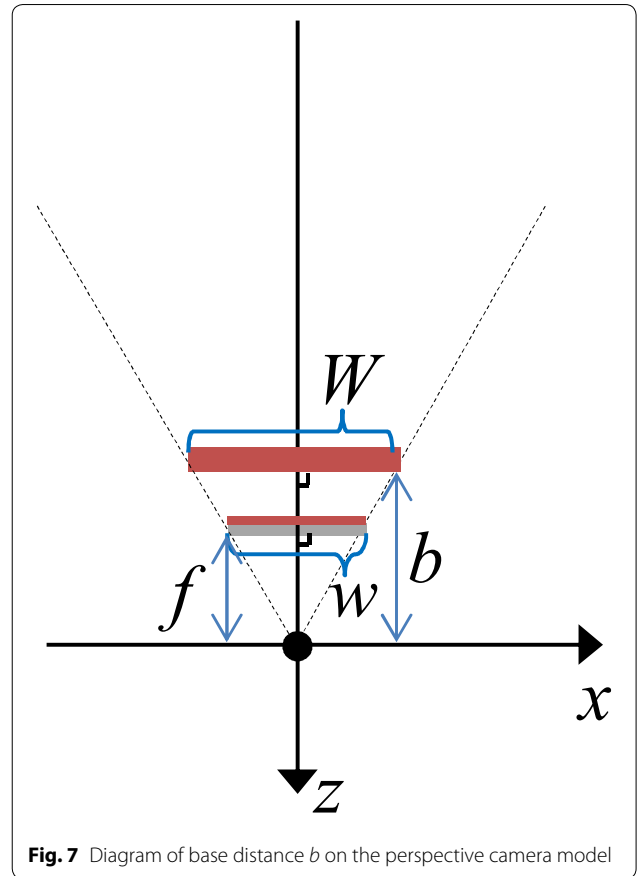
$$z' = kf \sqrt{\frac{WH}{w'h'}} \tag{6}$$

This indicates that given two areas of the object face  $WH$  and the projected object  $w'h'$  (e.g., BB), we can systematically calculate the distance  $z'$  of the target object. However, the width  $W$  and height  $H$  of the object face cannot be observed and is changeable depending on the orientation of the target object against the camera.

**Base distance**

To omit such unobservable value  $W$  and  $H$  from Eq. 6, we introduce the base distance  $b$  which is a small distance  $z'$  when the target object is close to the camera so that the area of projected object  $w'h'$  is also close to the area of image plane  $wh$ , i.e.,  $w'h' \approx wh$ , as shown in Fig. 7.

$$z' = kf \sqrt{\frac{WH}{w'h'}} \approx kf \sqrt{\frac{WH}{wh}} \equiv b \tag{7}$$



Using this base distance  $b$ , we express the distance  $z'$  of the target object without  $W$  and  $H$  by dividing Eq. 6 by Eq. 7 as follows:

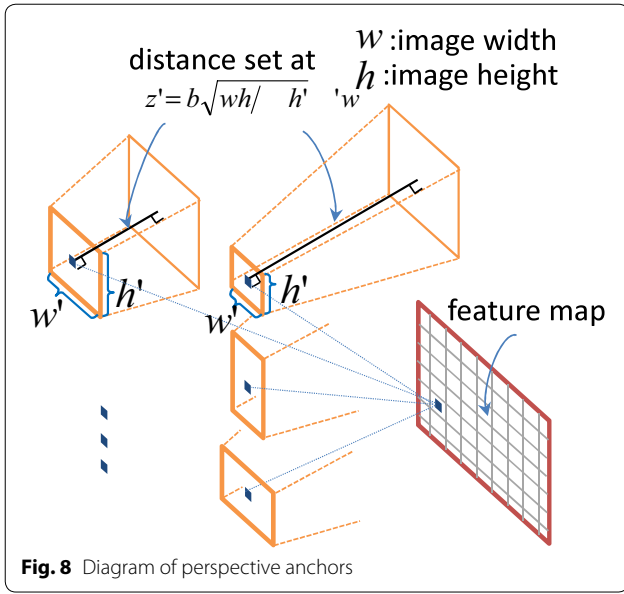
$$\frac{z'}{b} = \sqrt{\frac{wh}{w'h'}} \tag{8}$$

$$z' = b \sqrt{\frac{wh}{w'h'}} \tag{9}$$

That is, the distance  $z'$  is now expressed as the inverse of the projected object area  $w'h'$ , multiplied by the constants of the image plane area  $wh$  and the base distance  $b$  which can be set based on training data  $\mathcal{D}$ .

**Perspective anchor**

Although Eq. 9 may not hold in reality since the gap between  $w'h' \approx wh$  would be large, it must be helpful for *setting the candidate* of 2.5D anchors. From this idea, we generate 2.5D anchors as shown in Fig. 8 where at each pixel of the feature map, different shapes and sizes of 2D anchors are allocated and the depth of each anchor is set by Eq. 9. That is,  $z^i$  of 2.5D anchor  $\mathbf{a}_{2.5D}^i$  is calculated



given the image size  $wh$ , the anchor plane size  $w'h'$  and the base distance  $b$  is a hyperparameter.

The advantage of the perspective anchor with the base distance  $b$  lies in twofold: (i) the distance candidate  $z^i$  can be calculated only from the ratio between the size of the corresponding 2D anchor  $\mathbf{a}_{2D}^i$  and the size of the input image; (ii) the base distance  $b$  could be tuned automatically from training data. In addition, in the case that there is a large variance in the target object or there are multiple target object categories, our proposed 2.5D anchors can be flexibly extended by preparing multiple 2.5D anchors for each 2D anchor with different base distances. For example, if there are two object categories *car* and *tv monitor*, we can generate two 2.5D anchors as follows:

$$\mathbf{a}_{2.5D}^j = (\mathbf{a}_{2D}^{jT}, z_{car}^j)^T \quad (10)$$

$$\mathbf{a}_{2.5D}^{j+N_a} = (\mathbf{a}_{2D}^{jT}, z_{tv}^j)^T \quad (11)$$

where  $N_a$  is the number of 2D anchors.

### Setting of base distance

There is one hyper parameter in our proposed 2.5D anchors, base distance  $b$  to be tuned. Since the base distance  $b$  is a small distance when the target object is close enough to the camera, one heuristic approach would be to set  $b$  at  $\alpha$ -percentile of GT distances  $\{z^{*,i}\}_{i=1}^{N_{train}}$  in training data  $D$  as follows:

$$b = z_{sorted}^{*, \lfloor \frac{N_{train} \times \alpha}{100} \rfloor} \quad (12)$$

Here,  $z_{sorted}^*$  is the sorted distance in descending order.  $\alpha$  is usually set at small values like 3, 5 and 10.

### Training with perspective anchors

For each pair  $I^i$  and  $\mathbf{b}_{2.5D}^i$  of training data  $\mathcal{D}$ , we select positive and negative 2.5D anchors based on the following condition:

$$\mathbf{IoU}(\mathbf{a}_{2D}^j, \mathbf{b}_{2D}^{*i}) \geq \tau_{IoU} \wedge \mathbf{E}_{dist}(\mathbf{a}_{2.5D}^j, \mathbf{b}_{2.5D}^{*i}) \leq \tau_{dist} \quad (13)$$

$j = 1, 2, \dots, N_{anchor}$

where  $\mathbf{IoU}(\mathbf{a}_{2D}^j, \mathbf{b}_{2D}^{*i})$  is *Intersection over Union* (IoU) between a 2D anchor  $\mathbf{a}_{2D}^j$  and GT BB  $\mathbf{b}_{2D}^{*i}$ , and  $\mathbf{E}_{dist}(\mathbf{a}_{2.5D}^j, \mathbf{b}_{2.5D}^{*i})$  is *relative distance error* between the depth of 2.5D anchor  $\mathbf{a}_{2.5D}^j$  and the one of GT 2.5D BB  $\mathbf{b}_{2.5D}^{*i}$  defined as

$$\mathbf{E}_{dist}(\mathbf{a}_{2.5D}^j, \mathbf{b}_{2.5D}^{*i}) = \frac{|z - z^*|}{\max(z, z^*)} \quad (14)$$

$\tau_{IoU}$  and  $\tau_{dist}$  are the threshold of IoU and distance respectively, and  $N_{anchor}$  is the number of 2.5D anchors, e.g., 5940 for 3-shape and 3-size of 2.5D anchors.

That is, if 2.5D anchors are overlapping GT 2.5D BB to some extent, those anchors are selected as *positive* data and otherwise are selected as *negative* data. With selected positive 2.5D anchors, we train RP network using following loss-function<sup>1</sup>

$$L^m(RPN_{cls}, RPN_{reg}) \equiv \frac{1}{N_{cls}} \sum_{j=1} L_{cls}(p_{a_{2.5D}^j}^*, RPN_{cls}(\mathbf{a}_{2.5D}^j)) + \frac{1}{N_{reg}} \sum_{j=1} p_{a_{2.5D}^j}^* L_{reg}(\mathbf{b}_{2.5D}^{*i}, RPN_{bbox}(\mathbf{a}_{2.5D}^j)) \quad (15)$$

where  $N_{cls}$  and  $N_{reg}$  are the number of anchors for the binary classification (object or non-object) and the regression of 2.5D anchors for the  $i$ -th pair of  $I^i$  and  $\mathbf{b}_{2.5D}^{*i}$  respectively.  $p_{a_{2.5D}^j}^*$  is the true probability of being classi-

fied as object or non-object, i.e.,  $p_{a_{2.5D}^j}^* = 1$  for the anchor

satisfying the condition of Eq. 13 and 0 otherwise. Then, the softmax loss  $L_{cls}$  between  $p_{a_{2.5D}^j}^*$  is calculated. In addition,

$L_{reg}$  is the smooth L1 loss between the true 2.5D BB  $\mathbf{b}_{2.5D}^{*i}$  and the estimated 2.5D BB  $\widehat{\mathbf{b}}_{2.5D}$  for each anchor  $\mathbf{a}_{2.5D}^j$ . Using those loss functions, RP network for classification  $RPN_{cls}(\mathbf{a}_{2.5D}^j)$  and RP network for regression

<sup>1</sup> The definition of loss function of FC network is almost same as Eq. 15 and thus is omitted.

$RPN_{\text{bbox}}(\mathbf{a}_{2.5D}^j)$  consisting of convolution and ReLU layers are trained.

### Evaluation

In this section, we evaluate the performance of our proposed Faster R-CNN with 2.5D anchors using PASCAL 3D+ TV monitor [9] and KITTI car [4] datasets. We implement our proposed method by extending the *py-faster-rcnn* codes provided by github [10]. Our code will be also available on github <https://github.com/hirotaka-hachiya>.

### Evaluation metric

To evaluate the performance of the distance estimation without the influence of BB detector, we introduce *precision of distance* (PD) defined as follows:

$$\mathcal{S}_{\text{IoU}} \equiv \{i \mid \text{IoU}(\widehat{\mathbf{b}}_{2.5D}^i, \mathbf{b}_{2.5D}^{i,*}) \geq \tau_{\text{IoU}}, i = 1, 2, \dots, N_{\text{train}}\} \quad (16)$$

$$\text{PD} \equiv \frac{\#\{i \mid \mathbf{E}_{\text{dist}}(\widehat{\mathbf{b}}_{2.5D}^i, \mathbf{b}_{2.5D}^{i,*}) \leq \tau_{\text{dist}}, i \in \mathcal{S}_{\text{IoU}}\}}{\#\mathcal{S}_{\text{IoU}}} \quad (17)$$

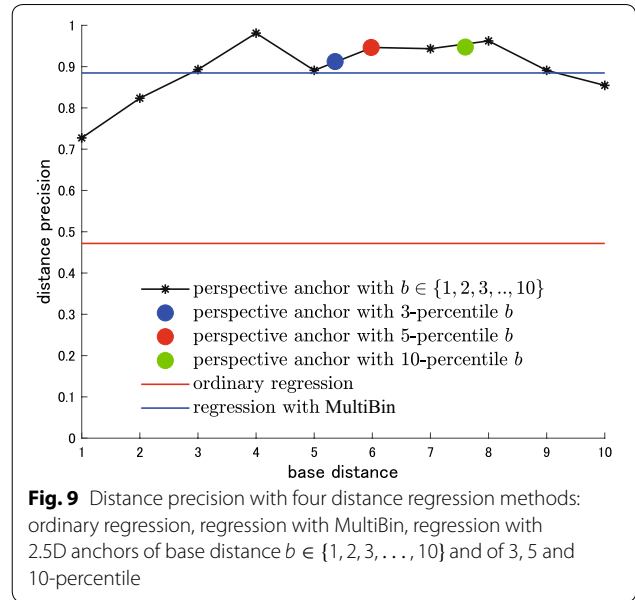
where  $\#\mathcal{S}$  is the number of element in a set  $\mathcal{S}$ . That is, PD is the ratio of correctly estimated distances in the set of corrected estimated 2D BBs. In following evaluations, the threshold values are set at  $\tau_{\text{IoU}} = 0.5$  and  $\tau_{\text{dist}} = 0.25$ .

### Evaluation on Pascal 3D+ TV monitor

We utilize the Pascal 3D+ dataset to evaluate the performance of distance estimation. Pascal 3D+ dataset provides 12 rigid categories with distance and we use only TV monitor category for our evaluation purpose. The reason why we chose the TV monitor from 12 categories is that there is a large variation in shape, size and pose as shown in Figs. 10, 11 and 12, and thus the regression of the distance would be reasonably difficult. There are 595 images in “tvmonitor\_pascal” and we randomly split the data to 90% (535 images) for training and validation data  $\mathcal{D}$ , and 10% (60 images) for testing data.

We compare four distance regression methods as follows

- Distance regression given 2D region proposals (ordinary regression)—the regression of the distance is performed in fully connected (FC) network for each



**Fig. 9** Distance precision with four distance regression methods: ordinary regression, regression with MultiBin, regression with 2.5D anchors of base distance  $b \in \{1, 2, 3, \dots, 10\}$  and of 3, 5 and 10-percentile

selected 2D BB by region proposal (RP) network<sup>2</sup> (see the network architecture described in Fig. 4).

- Distance regression with 2D region proposals and *MultiBin*—following Eq. 5 in the paper [5], the regression of residuals from the mean distance computed from training data  $\mathcal{D}$  is performed in FC network for each selected 2D BB by the pretrained RP network.
- Regression using 2.5D anchors with fixed base distance  $b$  set at each of  $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ —as for 2D anchors, 3-shape and 3-scale (8, 16 and 32) anchors are used [1].
- Regression using 2.5D anchors with the base distance  $b$  set by each of  $\{3, 5, 10\}$ -percentile of the GT distances in training data  $\mathcal{D}$  (see Eq. 12)—as for 2D anchors, 3-shape and 3-scale (8, 16 and 32) anchors are used [1].

Figure 9 and Table 1 depict the precision of distance for four regression methods. Figure 9 shows that ordinary regression (see red line) fails to predict distances for given 2D BBs, indicating that the distance regression for TV monitors with various appearance is prohibitively difficult. This difficulty would be mitigated by introducing *MultiBin* which split the target distance value to the half at the mean computed over training data [5] as the PD is improved to 0.88 (see blue line in Fig. 9 and Table 1). However, only the half-split is not helpful enough to

<sup>2</sup> RP network is pretrained with 2D BB  $\{b_{2D}^{*i}\}_{i=1}^{N_{\text{train}}}$  in the training data  $\mathcal{D}$  and fixed when training the distance regression.



**Fig. 10** Examples of results in Pascal3D+ TV monitor for a small TV monitor. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit of distance is based on CAD models



**Fig. 11** Examples of results in Pascal3D+ TV monitor for a medium size TV monitor. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit of distance is based on CAD models

achieve higher precision in comparison with our proposed methods. That is, regression with 2.5D anchors given good base distance  $b$  outperforms both ordinary regression and regression with MultiBin (see black line in Fig. 9). This indicates that with a tuned base distance  $b$  can be generated for the accurate distance regression. In addition, the setting of base distance  $b$  based on  $\alpha$ -percentile distance is good heuristic as those performances (see red, blue and green circles in Fig. 9 and Table 1) are well comparable with the best performance. This implies that our proposed 2.5D anchors are not so sensitive to the base distance and then users can easily tune using  $\alpha$ -percentile.

### Overlap-level of 2.5D anchors

For further analysis of the usefulness of perspective 2.5D anchors, we evaluate how largely 2.5D anchors overlap GT 2.5D BB. Table 2 depicts average  $\text{IoU}$ , average  $\mathbf{E}_{\text{dist}}$  and the PD of the closest 2.5D anchor to each of GT 2.5D BB  $\{b_{2D}^{*i}\}_{i=1}^{N_{\text{train}}}$ . That is, the closest 2.5D anchor here is the one holding the maximum  $\text{IoU}$  and the minimum  $\mathbf{E}_{\text{dist}}$  to each of  $\{b_{2D}^{*i}\}_{i=1}^{N_{\text{train}}}$ . This table shows that our perspective anchors have lower  $\mathbf{E}_{\text{dist}}$  and higher PD than 2.5D anchors with the  $\alpha$ -percentile depth. For example, PD of perspective anchors with  $\alpha = 3$  or  $\alpha = 5$  are about 0.5, meaning that the half of GT distances could be estimated by only 2.5D anchors without regression. This indicates that perspective anchors can provide good references for the distance regression to treat appearance variations. We note that the PD is improved to from 0.5 to 0.95 by the distance regression as shown in Fig. 9 and Table 1.

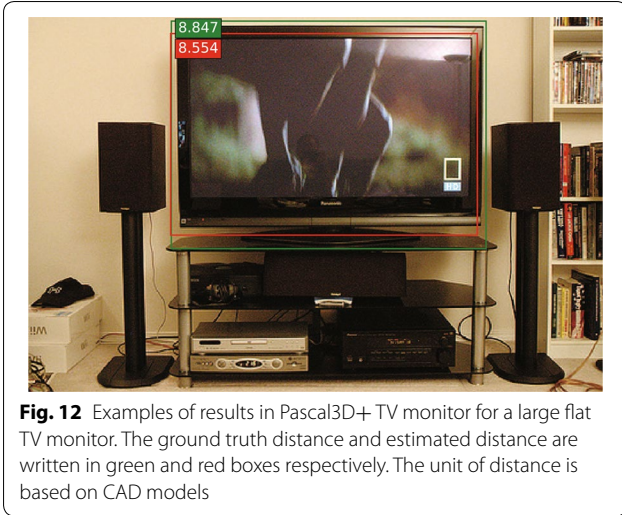
Figures 10, 11 and 12 depict examples of object detection and distance measurement of TV monitors in the case of 3-percentile  $b$ . This shows that small, medium and large sized TV monitors can be detected and those distances are estimated accurately (GT and estimation are depicted in green and red respectively). Note that although the distance value in Pascal 3D+ is not absolute, it is not problematic for the evaluation purpose.

### Evaluation on KITTI car

As mentioned in the section of Related Works, 3D box object detection methods using subcategories and MultiBin [5, 6] can also estimate the object distance (actually 3D location). These methods have the following procedure to estimate the 3D location as follows:

- 2D BB and the dimension and orientation of 3D box of the object are detected by the regression with subcategories or MultiBin.
- Using the estimated corners of 2D and 3D boxes, the projection matrix (including translation and rotation) between 2D and 3D boxes are estimated by solving the optimization problem.

Here, we evaluate the performance of our method on 3D localization problem in comparison with the state-of-the-art methods [5, 6]. To this purpose, we utilize KITTI dataset [4] with the training-test data indices provided by the paper [6]—7481 images in KITTI are split to 3682 for training and 3799 for testing. As for the evaluation metric, we use the average distance error between the estimated 3D location of objects and its GT following the paper [5]—to suppress the influence of the performance of 2D BB detection, only the detection satisfying  $\text{IoU} \geq 0.7$  are counted on the error. We call this error, *average location error* (ALC).



**Fig. 12** Examples of results in Pascal3D+ TV monitor for a large flat TV monitor. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit of distance is based on CAD models

**Table 1** Average precision (AP) over foreground thresholds  $\{0, 0.1, 0.2, \dots, 0.9, 1\}$  [16], precision of distance (PD) and total average precision (AP  $\times$  PD) for TV monitor in Pascal3D+

Regression method	AP	PD	AP $\times$ PD
With region proposals	0.77	0.47	0.36
With region proposals and MultiBin	0.78	0.88	0.69
With perspective anchors $\alpha = 3$	0.78	0.91	0.71
With perspective anchors $\alpha = 5$	0.77	0.95	0.73
With perspective anchors $\alpha = 10$	0.79	0.95	0.75

**Table 2** Average IoU, precision of distance (PD) and relative distance error of the closest 2.5D anchors to GTs (BB and distance) in Pascal3D+ TV dataset

2.5D anchor	Avg. IoU	Avg. E <sub>dist</sub>	PD (without regression)
Fixed depth, $z = 0$	0.45	1.0	0.0
Fixed depth, $z = \bar{z}$	0.45	0.39	0.29
Perspective anchor, $\alpha = 3$	0.45	0.26	0.50
Perspective anchor, $\alpha = 5$	0.45	0.27	0.49
Perspective anchor, $\alpha = 10$	0.45	0.33	0.35

To compute ALC for our method, we estimate  $x_{3D}$  and  $y_{3D}$  on the 3D camera coordinate system based on the perspective camera model and our estimated distance  $\hat{z}$  (see Fig. 6) as follows:

$$x_{3D} = \frac{kf}{\hat{z}} \left( \frac{\hat{x}_{\max} - \hat{x}_{\min}}{2} + \hat{x}_{\min} \right) \tag{18}$$

$$y_{3D} = \frac{kf}{\hat{z}} \left( \frac{\hat{y}_{\max} - \hat{y}_{\min}}{2} + \hat{y}_{\min} \right), \tag{19}$$

where  $k$  and  $f$  are calculated from KITTI GT data in advance.

Figure 13 depicts ALC of our proposed method with 1-percentile (black) in comparison with 3D BB detection approaches [5, 6]. As for 2D anchors in our method, 3-shape  $\times$  6-scale (4, 6, 8, 10, 16 and 32) anchors are used. This figure shows that our proposed method estimates the 3D location of the object quite accurately, i.e., the ALC is small in a range from 1.3 to 3.8 m. The performance of our method is well comparable with the state-of-the-art methods even in 3D localization problem, with the advantage of the low number of annotated target variables.

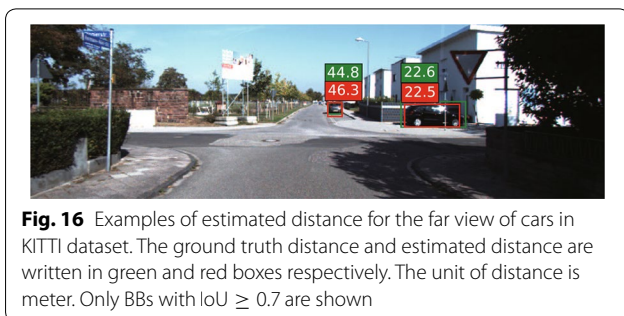
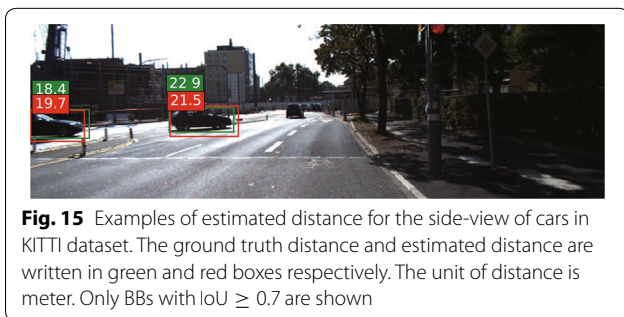
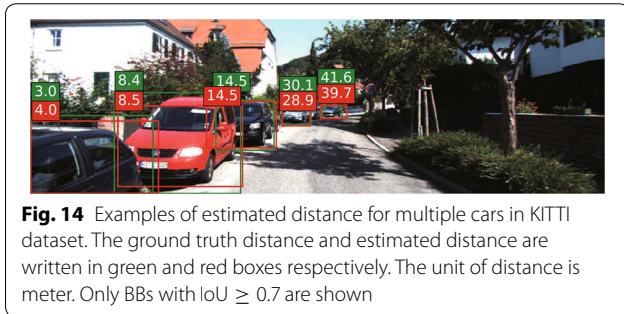
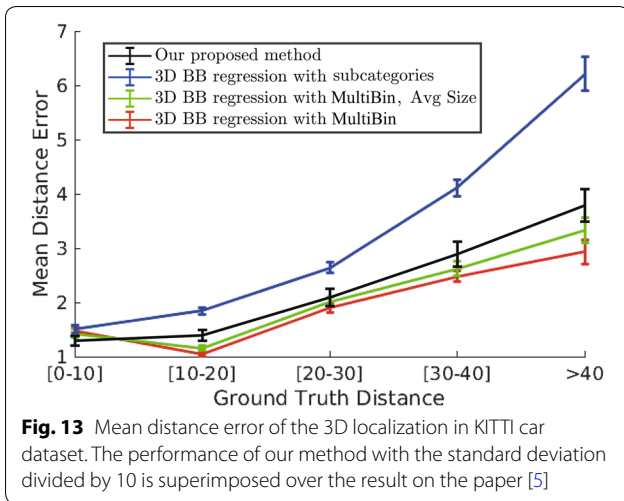
More in detail, in 3D box detection approaches [5, 6], since 2D BB and 3D box are detected, there are totally 8 annotated target variables, i.e., 4 for 2D BB, 4 for 3D box dimension and orientation, *height*, *width*, *length* and *orientation* at Y-axis in camera coordinate. Meanwhile, our proposed method estimates only 2D BB and distance, and there are only 5 target variables, i.e.,  $(x_{\min}^i, y_{\min}^i, x_{\max}^i, y_{\max}^i, z)$ .

This difference would be a great advantage for our method when considering real applications since annotating 3D box by a human is prohibitively expensive. That is, for each target object in many images, a 2D BB and a 3D box need to be annotated manually by a human with a careful consideration of the correct orientation and dimension of the object [4]. Meanwhile, in our method, if a laser sensor calibrated with cameras is available in the data collection phase like KITTI dataset, the distance annotation could be systematically performed given a 2D BB annotated by a human, e.g., by taking the average of corresponding distances measured by the laser sensor. We note that if a laser sensor is not available in the data collection phase, the distance can be annotated later from a single image using the CAD model of the target object, as shown in “Application to navigation” section.

Figures 14, 15, and 16 depict examples of estimated BB and distance for the side, front and far views of cars. These figures show that the estimated BBs and distances in red are reasonably close to GT in green in various view angles.

Overall, the experimental results with Pascal 3D+ TV monitor and KITTI car datasets show that the proposed method, 2.5D anchors is a promising approach for distance regression and even for 3D localization from the single camera image.





### Application to navigation

We apply our perspective anchors for a real robot navigation task in *Tsukuba challenge* [11], navigating to a person wearing a specific cloth. Our proposed method can be applied to such task by converting the estimated BB and distance to the location on a map maintained in SLAM system built with *slam\_gmapping stack* and *navigation stack* of *robot operating system* (ROS).

### Data and annotation

We collect 1006 images ( $744 \times 480$ -pixel) using three cameras [12, 13] along the course of the Tsukuba challenge in two different times—there are totally 6 data groups i.e., 3 cameras at 2 different times. We use Matlab codes provided by Pascal 3D+ [14] to annotate the 2D BB and the distance. More concretely, for the annotation of distance, we prepare CAD model for the signboard next to persons as depicted in the top left of Fig. 17. Then, we calculate the distance between camera and BB including both signboard and person by fitting the signboard CAD model to the 2D image as shown in the bottom of Fig. 17.

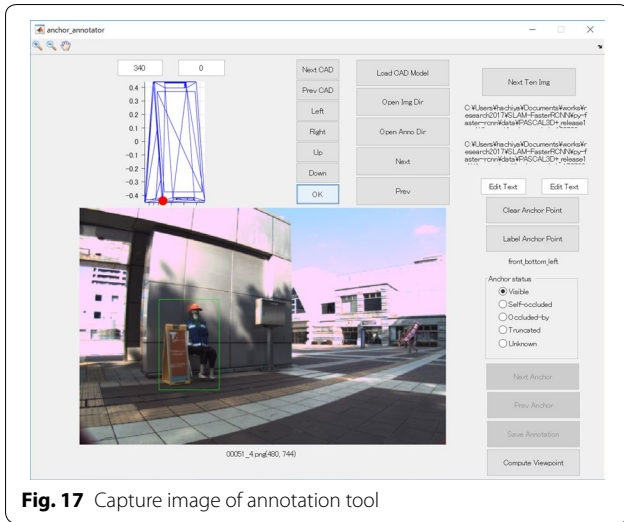
### Evaluation on distance measurement

We use the same setting and evaluation metric as Pascal 3D+ experiments with real average distance errors. We note that distance annotation computed by the annotation tool is of the unit in CAD but we can convert to the unit of the meter using a standard calculation based on camera geometry. Table 3 shows that our proposed method detects the target object and estimates its distance quite accurately. More specifically, average distance error (ADE) is only about 0.5 m although ordinary regression suffers from variation in appearance and has a large error, i.e., 1.92 m. This indicates that the distance regression for this task is also extremely difficult since the appearance of target objects changes largely due to the change of camera angle with moving robot. Related to this point, Figs. 18, 19 and 20 depict examples of estimated BB and distance for the side, front and far views. These figures show that the estimated BBs and distances in red are reasonably close to ground truth in green in various view angles.

In addition, Table 4 shows that the average process time of our proposed method is reasonably fast as 42.6 ms on a desktop PC with GTX980 and 314 ms on Jetson TX1. Note that as for the pre-trained CNN in Faster R-CNN, we employ *Zeiler Fergus* (ZF) net which is lighter and faster.

### Coordinate system transformation

To apply our proposed method to a robot navigation system implemented by ROS *navigation stack*, we created two original ROS nodes: *Faster R-CNN node* and



**Fig. 17** Capture image of annotation tool

**Table 3 Average precision (AP) over foreground thresholds  $\{0, 0.1, 0.2, \dots, 0.9, 1\}$  [16], precision of distance (PD), total average precision (AP  $\times$  PD) and average distance error (ADE) for Tsukuba dataset**

Regression method	AP	PD	AP $\times$ PD	ADE [m]
With region proposals	0.89	0.5	0.50	1.92
With region proposals and MultiBin	0.90	0.94	0.85	0.65
With perspective anchors $\alpha = 3$	0.90	0.99	0.90	0.5
With perspective anchors $\alpha = 5$	0.90	0.99	0.90	0.55
With perspective anchors $\alpha = 10$	0.90	0.99	0.90	0.52



**Fig. 18** Examples of estimated distance for the side-view of the target object in Tsukuba Challenge. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit is in meter

*Coordinate transform node* as shown in Fig. 21. The *Faster R-CNN node* is to run our proposed method on Jetson TX1 and to publish the information of detected BB and its distance. Meanwhile, the *coordinate transform*

*node* is to transform coordinate systems from images to SLAM map through the camera coordinate system. More in detail, firstly, the node converts image coordinate into camera coordinate, i.e., the centroid of BB  $x'$  into  $X$  in Fig. 22, using the following equations:

$$\theta = \frac{x'}{w} \phi, \quad X = z' \tan(\theta) \tag{20}$$

where  $\phi$  is the field-of-view of camera and  $\theta$  is the angle of the centroid of BB from the origin of camera coordinate system. Secondly, the node transforms the camera coordinate,  $(X, 0, z')$  to the SLAM map coordinate system  $(x_{map}, y_{map}, 0)$  (see Fig. 23) using standard rotation-translation matrices. We note that the camera coordinate  $Y \approx 0$  is assumed since the height of camera and the centroid of the target object (i.e, sitting person) is almost same.

Finally, the *navigation nodes* (see Fig. 21), i.e, *navigation stack*, set the subscribed map coordinate of the target object as the goal and navigate the robot along the map.

**Evaluation on navigation**

To evaluate the effectiveness of our navigation system, we conduct experiments using a real mobile robot, called *mercury* [15], equipped with a 2D LiDAR, wheel encoders, an IMU, cameras and Jetson TX1 as shown in Fig. 24. Then, we consider the following navigation scenario:

- Mobile robot starts moving from a specific start-point where the target can be found (see Fig. 25).
- After a while, a pedestrian (disturbance) stands in front of the robot for short or long time to hide the target object from the robot.
- Elapsed time from moving and reaching to the target object within 1 m is measured for the comparison.
- The translation and rotation velocity is set at 0.5 m/s and 90°/s.

Table 5 depicts elapsed time for both short and long disturbance. This table shows that our navigation system guides successfully the robot to the target object in short time even if the target object is lost by the disturbance—the robot does not need to keep eyes on the target since the position of the target is registered as the goal on *navigation stack*. Here are videos demonstrating our proposed our navigation system:

- <https://youtu.be/pPkgonE6tRM>
- <https://youtu.be/CC5hISe19R8>

for both disturbance cases.



**Fig. 19** Examples of estimated distance for the front-view of the target object in Tsukuba Challenge. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit is in meter



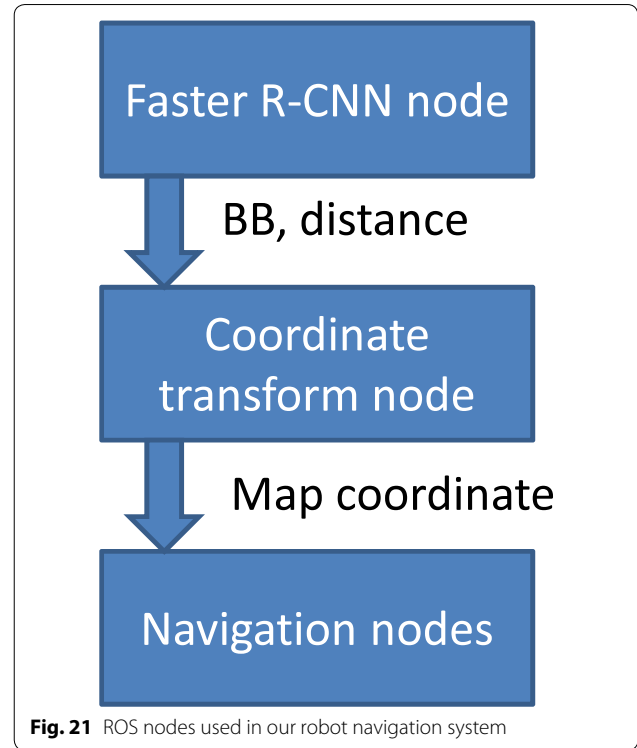
**Fig. 20** Examples of estimated distance for the far-view of the target object in Tsukuba Challenge. The ground truth distance and estimated distance are written in green and red boxes respectively. The unit is in meter

**Table 4 Process time in person detection and its distance measurement**

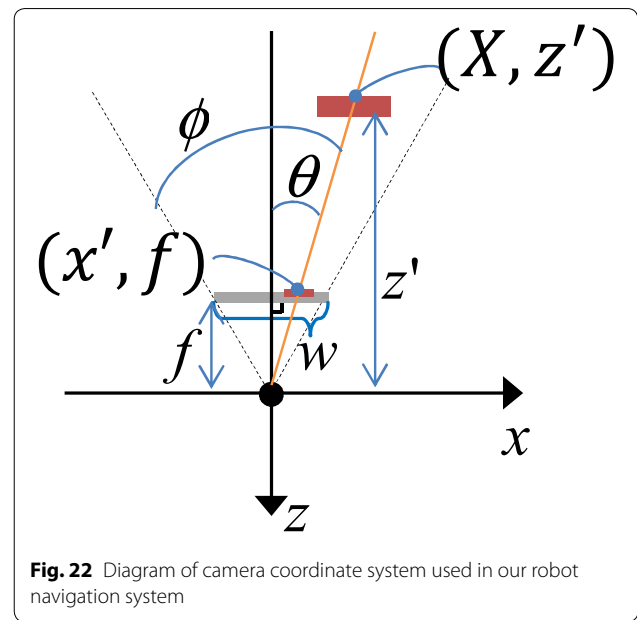
Method	Time (GTX980)	Time (Jetson TX1)
2.5D anchors with 3-percentile	42.6 ms	314 ms

**Conclusion**

In this paper, we have proposed 2.5D anchors (called *perspective anchors*), designed based on the perspective camera model, which are suitable for both bounding box and distance estimation in Faster R-CNN. Through the experiments with Pascal 3D+ TV monitor and KITTI car datasets, we have shown the effectiveness of our proposed method in the distance estimation and even in the 3D localization. In addition, we have demonstrated an example of practical uses of our proposed method in a



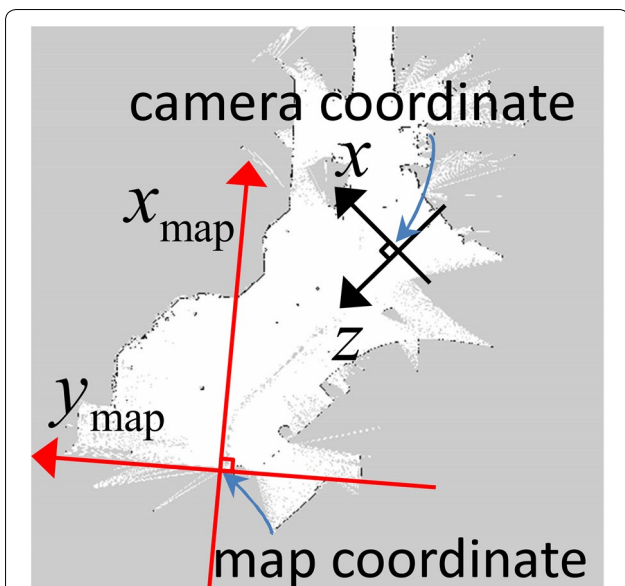
**Fig. 21** ROS nodes used in our robot navigation system



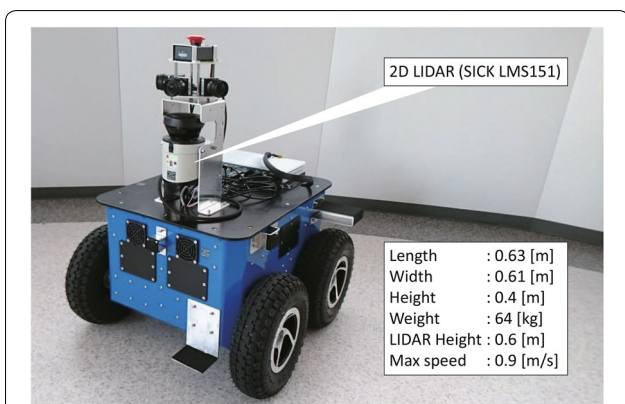
**Fig. 22** Diagram of camera coordinate system used in our robot navigation system

real-time system, robot navigation, by ROS-based simultaneous localization and mapping (SLAM).

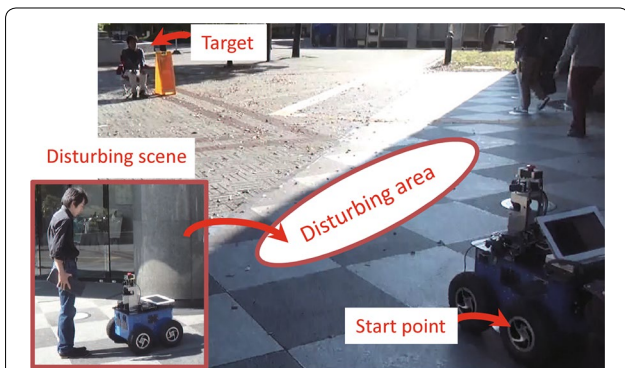
In this paper, we consider estimating the distance of a specific target-object category, i.e., TV monitors, cars or humans. However, in a real application such as an autonomous driving system, multiple target objects, e.g., pedestrian and car need to be treated at the same time.



**Fig. 23** Grid-based map with coordinate systems used in our robot navigation system



**Fig. 24** Mobile robot used in our robot navigation system



**Fig. 25** Experimental setting with navigation target and obstacle

**Table 5** Elapsed time of navigation

Method	Short disturb.	Long disturb.
Proposed navigation	16.4 s	20.7 s

Thus, the task of multiple 3D object distance measurement would be our future work. Although we believe that our method could be flexibly extended to such case by setting multiple base distances to each 2D anchor, further research is needed to investigate an efficient way of the multi-class distance regression problem.

In addition, in this paper, we extend one of the state-of-the-art object detection method, Faster R-CNN [1]. Recently, there are advanced object detection methods, e.g., YOLO2 [2], which provide a better and faster performance. Thus, extending such advanced method with the concept of our proposed 2.5D anchor will be also a future work.

**Authors' contributions**

HH and YS conceived of the presented idea, designed the presented algorithm, and carried out the implementation of the algorithm. KI and MN implemented the system of presented robot navigation. HH, KI and MN carried out experiments. HH developed the theoretical formalism, carried out the analysis of the experimental results, and wrote the manuscript with the support of YS and TN. All authors read and approved the final manuscript.

**Author details**

<sup>1</sup> Faculty of System Engineering, Wakayama University, 930 Sakaedani, Wakayama-shi 640-8510, Japan. <sup>2</sup> Department of Statistical Science, Graduate University for Advanced Studies, 10-3 Midori-cho, Tachikawa, Tokyo 190-8562, Japan.

**Acknowledgements**

This work was supported by JSPS KAKENHI Grant Number JP17H06871. We appreciate Revast Co., Ltd and Dr. Yuta Kanuki for providing us the mobile robot *mercury* and images captured in *Tsukuba challenge*.

**Competing interests**

The authors declare that they have no competing interests.

**Publisher's Note**

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 30 January 2018 Accepted: 28 August 2018

Published online: 10 September 2018

**References**

- Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems (NIPS)
- Joseph R, Ali F (2016) Yolo9000: better, faster, stronger. arXiv preprint [arXiv :1612.08242](https://arxiv.org/abs/1612.08242)
- Mur-Artal R, Tardós JD (2015) ORB-SLAM: a versatile and accurate monocular slam system. *IEEE Trans Robot* 31(5):1147–1163
- Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? The KITTI vision benchmark suite. In: Proceedings of 2012 IEEE international conference on computer vision and pattern recognition (CVPR2012)

5. Mousavian A, Anguelov D, Flynn J (2017) 3D bounding box estimation using deep learning and geometry. In: Proceedings of 2017 IEEE international conference on computer vision and pattern recognition (CVPR2017)
6. Xiang Y, Choi W, Lin Y, Savarese S (2017) Subcategory-aware convolutional neural networks for object proposals and detection. In: Proceedings of 2017 IEEE winter conference on applications of computer vision (WACV2017)
7. Chabot F, Chaouch M, Rabarisoa J, Teuliere C, Chateau T (2017) Deep MANTA: a coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In: Proceedings of 2017 IEEE international conference on computer vision and pattern recognition (CVPR2017)
8. Xiang Y, Choi W, Lin Y, Savarese S (2015) Data-driven 3D voxel patterns for object category recognition. In: Proceedings of 2015 IEEE international conference on computer vision and pattern recognition (CVPR2015), pp 1903–1911
9. Xiang Y, Mottaghi R, Savarese S (2014) Beyond PASCAL: a benchmark for 3D object detection in the wild. In: Proceedings of 2014 IEEE winter conference on applications of computer vision (WACV2017)
10. Girshick R. Faster R-CNN (Python implementation). <https://github.com/rbgirshick/py-faster-rcnn>
11. Website of Tsukuba Challenge (2017) <http://www.tsukubachallenge.jp>
12. Website of ARGO CORPORATION. <https://www.argocorp.com/cam/usb2/tis/DxK22xUC03.html>
13. Website of TAMRON. [http://www.tamron.biz/data/ipcctv/cctv\\_ir/13fm28ir.html](http://www.tamron.biz/data/ipcctv/cctv_ir/13fm28ir.html)
14. Xiang Y. Pose\\_Dataset. [https://github.com/yuxng/Pose\\_Dataset](https://github.com/yuxng/Pose_Dataset)
15. Website of Revast. <http://revast.co.jp>
16. Everingham M, Gool LV, Williams CKI, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. *Int J Comput Vision (IJCV)* 88:303–338

**Submit your manuscript to a SpringerOpen<sup>®</sup> journal and benefit from:**

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

---

Submit your next manuscript at ▶ [springeropen.com](http://springeropen.com)

---