

Indoor Obstacle Discovery on Reflective Ground via Monocular Camera

Feng Xue · Yicong Chang · Tianxi Wang · Yu Zhou · Anlong Ming

Received: 16 March 2022 / Accepted: 30 September 2023

Abstract Visual obstacle discovery is a key step towards autonomous navigation of indoor mobile robots. Successful solutions have many applications in multiple scenes. One of the exceptions is the reflective ground. In this case, the reflections on the floor resemble the true world, which confuses the obstacle discovery and leaves navigation unsuccessful. We argue that the key to this problem lies in obtaining discriminative features for reflections and obstacles. Note that obstacle and reflection can be separated by the ground plane in 3D space. With this observation, we firstly introduce a pre-calibration based ground detection scheme that uses robot motion to predict the ground plane. Due to the immunity of robot motion to reflection, this scheme avoids failed ground detection caused by reflection. Given the detected ground, we design a ground-pixel parallax to describe the location of a pixel relative to the ground. Based on this, a unified appearance-geometry feature representation is proposed to describe objects inside rectangular boxes. Eventually, based on segmenting by detection framework, an appearance-geometry fusion regressor is designed to utilize the proposed feature to discover the obstacles. It also prevents our model from concentrating too much on parts of obstacles instead of whole obstacles. For evaluation, we introduce a new dataset for Obstacle on Reflective Ground

(ORG), which comprises 15 scenes with various ground reflections, a total of more than 200 image sequences and 3400 RGB images. The pixel-wise annotations of ground and obstacle provide a comparison to our method and other methods. By reducing the misdetection of the reflection, the proposed approach outperforms others. The source code and the dataset will be available at <https://github.com/XuefengBUP/IndoorObstacleDiscovery-RG>.

Keywords Reflective Ground · Obstacle Discovery · Homography

1 Introduction

In indoor environments, obstacles, e.g., charging cable, key chain, and wallet, etc., endanger mobile robots by tangling wheels or causing the robot to overturn. However, the 2D LiDAR commonly used for navigation only perceives obstacles with large heights, while obstacles lower than LiDAR are hard to be perceived. Hence, as a potentially effective way, cost-effective cameras are usually used in many researches [48,21,6] for discovering these hazards. While most existing approaches are mainly applied in environments with texture-less and non-reflective floor, and lack discussion on the indoor scene laying reflective floor. In practical scenarios, the floor with a mirror surface is prevalent and brings a challenge to obstacle discovery. In the Field of View (FOV) of a robot, the floor reflects real-world objects that we call unreal objects (UOs), e.g. the reflections of plant, gate, and person in Fig. 1. UOs generally have complex textures and surround obstacles, making them even harder to detect using classic depth sensors (see Appendix C) or eliminate with reflection removal algorithm (see Appendix B). This makes it challenging for discovery models to distinguish the obstacles from the floor, ultimately leads to confusion in robot navigation.

Corresponding author : Yu Zhou
F. Xue, Y. Chang and A. Ming
School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
E-mail: {xuefeng,yicongchang,mal}@bupt.edu.cn
Y. Zhou
School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China
E-mail: yuzhou@hust.edu.cn
Tianxi Wang
DiDi Autonomous Driving, Beijing 100094, China
E-mail: wangtianxi@didiglobal.com

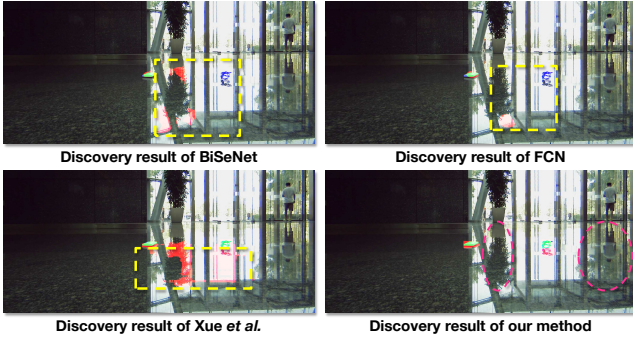


Fig. 1 Results of BiSeNet [47], FCN [40], Xue *et al.* [45] and our method on an exemplary scene of the proposed dataset. True positives are marked in green, red for false positives, blue for false negatives. Yellow boxes mark the mis-classified pixels. Magenta circles indicate the reflection.

As we know, there are currently no obstacle discovery methods investigating this task. Thus, in the following paragraphs, we discuss the feasibility of using the existing obstacle discovery methods on reflective ground. Overall, they can be mainly assigned to three groups.

Stereo-based conventional methods [36, 3] reconstruct 3D scenes by using a stereo camera, and classify the 3D points higher than the ground plane as obstacle. However, the accuracy of obstacle discovery highly depends on the quality of 3D reconstruction that is easily affected by the mirror reflection. Thus, obstacles cannot be distinguished from UOs by using the unreliable 3D information.

Monocular-based conventional methods [48, 6, 49, 23] first estimate a ground plane by registered feature point pairs [39, 5] between consecutive monocular images. Then, they judge whether a pixel is coplanar with the estimated ground. The pixels that are not coplanar with the ground indicate obstacles. However, the feature points of reflection dominate the ground area, and their 3D information is totally different from the ground plane, leading to the mis-detection of ground. For this reason, these solutions cannot be directly used to distinguish obstacles from UOs.

Learning-based methods [21, 45, 37, 11, 46, 42, 25] train models to classify pixels or region proposals [10, 22] in a single image into obstacle and non-obstacle. However, these methods do not capture the difference between reflections and the real world. Thus, in the face of complex, diverse and unseen ground textures, it is difficult for them to avoid mis-detecting the ground as obstacles. Fig. 1 depicts the prediction of two segmentation models and the baseline. These methods mistake the reflection of plant and gate for obstacles, and miss the real obstacles.

Although the use of homography for obstacle detection is not really new [49, 48, 6], we found that homography is good for expressing the difference between UOs and obstacles because different planes, which is rarely discussed in previous works. To fully leverage such a key

characteristic, we first follow [45] to gain the candidate proposals enclosing objects, and then construct a unified appearance-geometry feature representation to express and re-score the candidate proposals. In detail, the homography of ground is firstly estimated for feature construction. To avoid the failed ground detection [48, 49] on the reflective floor, a pre-calibration based ground detection scheme is introduced. This scheme uses robot motion, instead of registered feature point pairs [48], to figure the ground homography. Thus, it avoids failed detection caused by reflection. Secondly, we take the occlusion edge point [31] of the scene as the key point for feature extraction, and propose a ground-pixel parallax. It measures the homography difference between an occlusion edge point and the detected ground, thus is able to express whether this point is above or below the ground. Then, as a key discriminative feature, the parallax incorporates with appearance features [45] to form a unified appearance-geometry feature representation for region proposals. Finally, an appearance-geometry fusion model (AGFM) is designed to re-score all proposals. In AGFM, we carefully study the effects of fast-moving objects on features, and adaptively use the geometric and appearance features to eliminate the effects of fast-moving. Besides, to segment obstacles more accurately, AGFM utilizes a well-designed weight-decayed probability generation scheme to gain an obstacle-occupied probability map. It reduces weights of low-rank obstacle proposals to avoid concentrating too much on parts of obstacles.

To evaluate the effectiveness of our method, a dataset named Obstacle on Reflective Ground (ORG) is introduced, which is the first dataset focusing on discovering obstacles on reflective floor, as far as we know. Inspired by the obstacle discovery task in other scenarios [45, 11, 16], several segmentation methods [40, 4, 47, 35, 45] are employed to compare with our method. The experimental results prove that the presented approach significantly alleviates false positives and false negatives compared with other methods, and is robust against the noise of robot motion and motion blur.

To our knowledge, our method is the first stab to discover obstacle on reflective ground. The key insights lie in:

- A pre-calibration based ground detection scheme is introduced, which uses robot motion that is immune to reflection, thus avoiding detection failure.
- A ground-pixel parallax is introduced to form a unified appearance-geometry feature representation for region proposals together with appearance feature.
- An appearance-geometry fusion model (AGFM) is proposed to locate obstacle. It also avoids the performance degradation caused by fast-moving objects and proposals too concentrated on parts of obstacles.
- An Obstacle on Reflective Ground (ORG) dataset is proposed, which contains 15 challenging scenarios, and 200

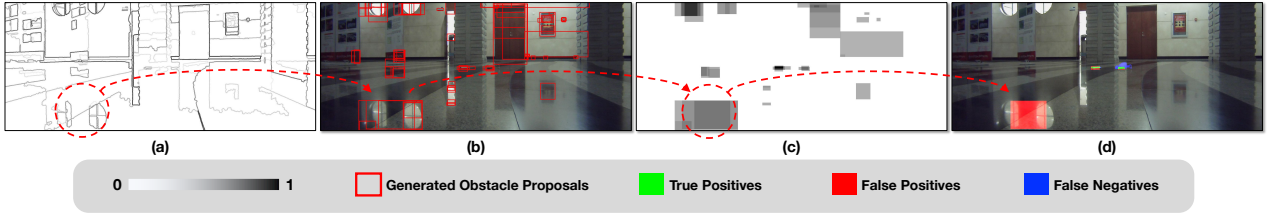


Fig. 2 The result obtained by previous method [45]. (a) the occlusion edge map. (b) the region proposals with high score (marked in red boxes). (c) the obstacle-occupied probability map constructed by these high-score boxes. (d) the final obstacle masks (using threshold 0.49 to segment the obstacles). In the grayscale images (a)(c), the value of each pixel ranges from 0 to 1. The darker it is, the closer it is to 1.

sequences. Compared to other segmentation models, our approach achieves a better performance on this dataset.

2 Related Work

Since our method is related to the general obstacle discovery [45, 46], occlusion edge, proposal extraction, and the ground homography. Brief introductions for them are given.

2.1 The Method Segmenting Obstacle by Detection

Xue et al. [45] propose to segment the obstacle by detection. In this pipeline, each region proposal obtained by [28] is expressed by a feature vector, which is used to train the random forest to detect the obstacle from proposals. Finally, all obstacle proposals are used to construct an obstacle-occupied probability map which is used to determine the obstacle pixels. Due to locating obstacles by the confidence of numerous proposals, instead of one proposal, this method achieves an approximate performance of the Convolutional Neural Networks (CNN) based methods [11, 37].

Unfortunately, the UOs are usually so complicated that the existing features cannot express the difference between them and obstacles, e.g. the occlusion edge in Fig.2(a). Hence, the UOs and obstacles are captured indiscriminately (see Fig.2(b)). Eventually, the UOs obtain high confidence and are mis-detected as obstacles, as shown in Fig.2(c)(d). This phenomenon accounts for a large proportion of indoor scenarios. Thus, our method aims to distinguish the proposals of UOs and obstacles by appearance-geometry features, boosting the performance of obstacle discovery.

2.2 Occlusion Edge and Region Proposal

Occlusion edge [15, 26, 32] locates the pixels indicating depth discontinuity between objects and background. Due to exploiting surface cue, it generally has a more robust confidence in the object contour than the typical edge cues [43, 1, 24]. By considering the occlusion edge points inside a bounding box, object-level proposal (OLP) [28] models

an occlusion-based objectness score to measure the probability that the bounding box contains object. The high-score bounding boxes are retained from densely sampled sliding windows, and are considered as the candidate proposals.

In this paper, we firstly employ [31] to detect the occlusion edge from the scene (see Fig.3 I), and then utilize OLP [28] to extract a set of candidate obstacle proposals (see Fig.3 II). The resulting occlusion edge and proposals are taken as the inputs of our method.

2.3 Homography of Planar Surface

With the representation in [48, 21, 49], for a set of the point pairs $\{x_i \leftrightarrow x'_i\}$ from two images, if the points $\{x_i\}$ are coplanar, there is a homography matrix $H \in \mathbb{R}^{3 \times 3}$:

$$x_i = Hx'_i \quad (1)$$

where $x_i, x'_i \in \mathbb{R}^{3 \times 1}$ are the homogeneous image coordinates. Since the matrix H has eight degrees of freedom [49, 12], a minimum of four non-degenerate point correspondences are required to determine H .

By searching for the coplanar points on the ground and registering them between two frames, several methods [6, 34] estimate the homography matrix to represent the ground plane. However, in the case of reflective ground, the complex UOs produce a lot of feature point pairs that are the outliers to the ground homography. Evidently, detecting the ground plane from point pairs with numerous outliers is difficult. Even, this phenomenon easily results in the virtual plane problem [48], namely, detecting a plane that does not correspond to the physical ground plane. In this paper, we utilize the robot motion, instead of the point pairs, to calculate the ground homography. Since the robot motion is independent of ground appearance, the proposed ground detector is immune to reflection.

3 Method

3.1 Overview

Given an image sequence $\{I^1, I^2, \dots, I^t\}$ acquired by the robot, where $t \in \mathbb{N}^+$ is the current time, our approach takes

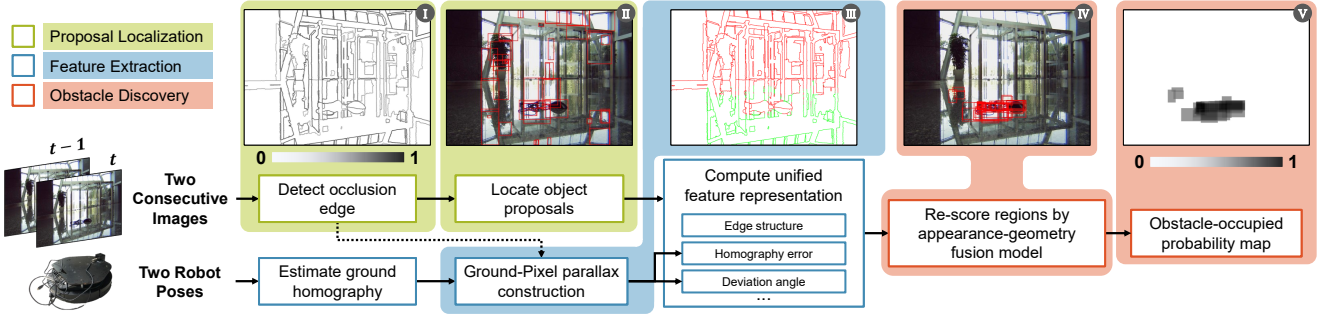


Fig. 3 The pipeline of our method. The inputs are consecutive RGB images and the robot pose corresponding to the two images. I-IV are the byproducts. V is the output, an obstacle-occupied probability map. In I and V, the value of each pixel ranges from 0 to 1. The darker it is, the closer it is to 1. In III, the points are divided into two types by setting threshold: the red points are above the ground, the green points are below the ground.

two frames I^t and I^{t-q} as inputs and generates a probability map P . Each element in the map P represents the presence of obstacles in image I^t and $q \in \mathbb{N}^+$ is the time interval. To simplify the representation, we default q to 1.

The pipeline is shown in Fig.3. We extract the occlusion edge map and the candidate proposals from image I^t , as mentioned in Sec. 2.2. To reduce the computation, we only preserve τ_e percent occlusion edge points with the top response. The candidates with the highest objectness score are denoted as $\mathcal{B} = \{b_j^t | j \in \{1, 2, \dots, J\}\}$, as the red rectangles shown in Fig. 3 II. However, through the projection from the 3D space to the 2D image plane, the detailed 3D information is lost in the 2D image. Thus, \mathcal{B} contains many false-positive obstacle proposals caused by the reflective ground. A key observation of our approach is that different planes in the 3D space satisfy different homography, thus, aiming to distinguish the real obstacles and the UOs, we leverage the discriminative characteristic of the homography, construct a unified appearance-geometry feature representation to express and re-score the obstacle proposals in \mathcal{B} .

More specifically, taking the occlusion edge points $\mathcal{P}^t = \{x_1^t, x_2^t, \dots, x_n^t\}$ as the key point for feature extraction, the Lucas-Kanade(LK) optical flow approach [27] is employed to track the occlusion edge point $x_i^t \in \mathcal{P}^t$ from image I^t to I^{t-1} , capturing the appearance corresponding point a_i^{t-1} on image I^{t-1} . In addition, we leverage the ground plane as the base reference plane, and compute the homography H of the ground plane (see Sec 3.2). By using H , we find the geometric corresponding point $g_i^{t-1} = Hx_i^t$ of $x_i^t \in \mathcal{P}^t$ in I^{t-1} . Furthermore, we define the parallax of the points by considering the relationship of a_i^{t-1} and g_i^{t-1} (see Sec 3.3). As shown in Fig. 4, such a parallax reflects a significant difference when the obstacle exists above or under the ground, and hence we leverage such a difference as a key discriminative feature to distinguish the UOs. Therefore, we gather the parallax values of the occlusion edge points inside a region proposal, and then express a proposal by jointly considering both the appearance cues and the geometric cues (see

Sec 3.4). An appearance-geometry fusion model (AGFM) is trained to re-score all region proposals (see Sec 3.5). Multiple proposals with the highest confidence are employed to form an obstacle-occupied probability map. Compared with the existing approaches, our approach takes the intrinsic geometric cue of the proposals into consideration and hence alleviates the false-positive obstacle proposals caused by the reflection of the ground efficiently.

3.2 Ground Plane Detection via Pre-calibration

The homography of the ground plane between both consecutive frames is a basic component for the geometric representation of proposal. However, the reflection obstructs the ground detection. To detect the ground without the effect from reflection, we first calibrate the ground parameter offline. In detail, since the camera is mounted on the robot platform that moves on a planar floor, the ground is indicated as a constant vector $\pi = \{\mathbf{n}^T, \mathbf{d}\}$, and the camera moves on a plane that satisfies $\mathbf{n}^T X + \mathbf{d} = 0$, where $\mathbf{n}^T \in \mathbb{R}^{3 \times 1}$ denotes the ground normal. The camera height \mathbf{d} can be measured directly, and the normal \mathbf{n}^T is calibrated as follows:

1. Take two images by the camera in two different poses of robot.
2. Manually mark a few (≥ 4) ground points pairs. $\{x_i\}$ and $\{x'_i\}$ denote the ground points of the two images, which is one-to-one correspondences: $\{x_i \leftrightarrow x'_i\}$.
3. Compute a homography H by these correspondences: $x'_i = Hx_i$.
4. Decompose the homography H by the decomposing method [29] to obtain a proper ground normal \mathbf{n}^T .

Based on the pre-calibrated ground π and the robot pose, the ground homography can be determined online. Specifically, we firstly utilize the odometer-camera calibration approach [9, 14] to obtain the transformation matrix from the robot's wheel odometer to the camera. Then, given the robot poses from the odometer at time t and $t-1$, the poses

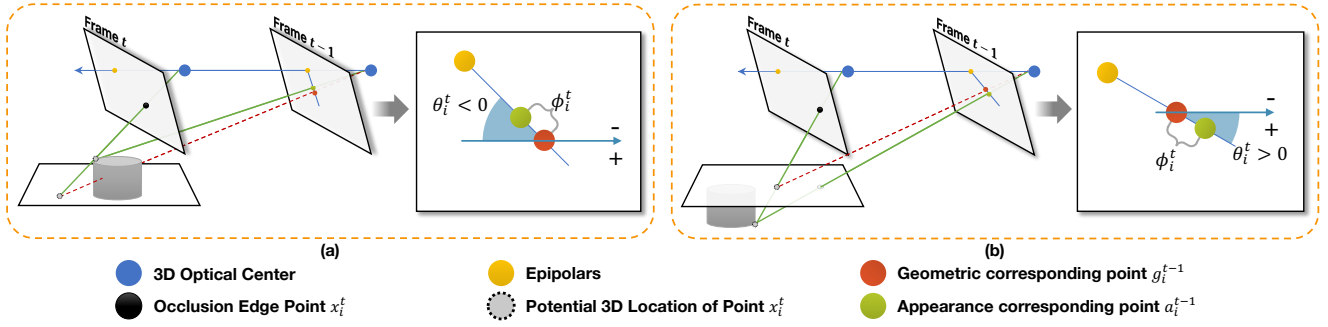


Fig. 4 (a) the case that the observed point is above the ground. (b) the case that the observed point is below the ground. For each case, the points are zoomed in to the right-side images to obtain a clear view.

of the camera at time t and $t - 1$ are figured by transformation, which are denoted as T^t and T^{t-1} , where $T^t = [R^t|C^t]$ ($R^t \in \mathbb{R}^{3 \times 3}$, $C^t \in \mathbb{R}^{3 \times 1}$). Referring to [49, 12], the homography of ground π can be directly calculated by the formulation in [12]:

$$H = K(\Delta R - \Delta C \mathbf{n}^T / d) K^{-1} \quad (2)$$

where $K \in \mathbb{R}^{3 \times 3}$ is the intrinsic matrix of the camera, and $\Delta R = R^{t-1}(R^t)^{-1}$ is the relative rotation of the camera from T^t to T^{t-1} , $\Delta C = C^{t-1} - C^t$ corresponds to the relative translation. Since the ground vector \mathbf{n}^T is pre-calibrated, and the camera motion $[\Delta R|\Delta C]$ is independent of the pattern inside the ground area, Equation 2 avoids false ground detection when working on reflection floor. Thus, the proposed scheme is suitable for reflective ground.

3.3 Ground-Pixel Parallax of Occlusion Edge Point

By modeling the contrastive property between point and ground plane, we propose ground-pixel parallax to judge whether an occlusion edge point is real or reflection. With the representation in Sec. 3.1, the observed occlusion edge points are denoted as $\mathcal{P}^t = \{x_1^t, x_2^t, \dots, x_n^t\}$. For each point inside set \mathcal{P}^t , the point's appearance and geometry corresponding points in frame $t - 1$ are denoted as a_i^{t-1} and g_i^{t-1} , as shown by the green and red circles in Fig.4. In addition, we denote the epipoles of frames I^t, I^{t-1} as e^t and e^{t-1} , namely, the yellow circles in Fig.4. Since g_i^{t-1} is determined by the ground homography H , it can be considered as the projection of a 3D ground point. Hence, the parallax of point x_i^t to the ground is defined as the difference between the appearance corresponding point a_i^{t-1} and the geometry corresponding point g_i^{t-1} :

$$\mathbf{p}_i^t = a_i^{t-1} - g_i^{t-1} \quad (3)$$

According to [12], since the three points g_i^{t-1} , e^{t-1} , and a_i^{t-1} are collinear, the vector \mathbf{p}_i^t can also be extended as:

$$\mathbf{p}_i^t = a_i^{t-1} - g_i^{t-1} = \rho(g_i^{t-1} - e^{t-1}) \quad (4)$$

where ρ is a scalar representing the deviation relative to the ground. It indicates which side of ground the real 3D location of point x_i^t is in the following way:

- If $\rho = 0$, the real 3D location is coplanar with the ground.
- If $\rho < 0$, the real 3D location is above the ground, as depicted in Fig.4 (a).
- If $\rho > 0$, the real 3D location is under the ground, as depicted in Fig.4 (b).

Observably, vector \mathbf{p}_i^t reveals that point x_i^t is above or below the ground. The proofs of Equation 4 and detailed technical discussion are included in the supplementary materials.

3.4 Appearance-Geometry Feature Representation

For each proposal b_j^t in \mathcal{B} , we propose a unified appearance-geometry feature representation, which has two parts, as shown in Table 1. Inspired by another obstacle detector [45], the first part contains edge cue, pseudodistance, objectness score, and color cue. For the second part, we gather the parallaxes of occlusion edge points inside the proposal as feature. As the ground-pixel parallax \mathbf{p}_i^t cannot be normalized and is easily affected by failed optical flow tracking, we decompose it into two features: homography error and deviation angle.

Region-level Homography Error: To express the relative distance of a proposal b_j^t to the ground, we define the homography error of occlusion edge points x_i^t enclosed by b_j^t : $\phi_i^t = \|a_i^{t-1} - g_i^{t-1}\|$, as shown in Fig.4. According to [21, 48, 49], it depicts the 2D deviation from occlusion edge point x_i^t to the ground. For proposal b_j^t , the region-level homography error is defined as the mean error of occlusion edge points x_i^t enclosed by proposal b_j^t :

$$\Phi_j^t = 1/N_j^t \sum_{x_i^t \in b_j^t} \phi_i^t \quad (5)$$

where N_j^t is the number of occlusion edge points in the proposal b_j^t . The region-level homography error indicates how far the proposal b_j^t is from the ground.

Table 1 The appearance-geometry feature representation of region proposal.

Part	Category	Feature name
Appearance	Edge cue	max edge response proportion of most response average edge response average edge response in inner ring
	Pseudo distance	normalized area aspect ratio X,Y coordinate of the region center width, height
	Objectness	occlusion-based objectness
	Color	color deviations in the H,S,V channel color contrasts of the H,S,V channel
Geometry	Parallax	homography error deviation angle

Region-level Deviation Angle: To express which side of the plane π the observed 3D point is, we define the deviation angle as the angle of point a_i^{t-1} in the polar coordinate system centered on point g_i^{t-1} , which is formulated as:

$$\theta_i^t = \gamma \arcsin\left(\frac{[a_i^{t-1}]_2 - [g_i^{t-1}]_2}{\phi_i^t}\right) \quad (6)$$

where $[\cdot]_2$ denotes taking y-value of a pixel. γ is a parameter, which is $+1$ when the robot moves forward, conversely -1 . The region-level deviation angle of proposal b_j^t is stated as:

$$\Theta_j^t = 1/N_j^t \sum_{x_i^t \in b_j^t} \theta_i^t \quad (7)$$

Although it seems feasible to use $[a_i^{t-1}]_2 - [g_i^{t-1}]_2$ directly to distinguish the proposals of obstacles and UOs, the deviation angle θ_i^t is more suitable. The reason is that θ_i^t has a fixed value range and is not easily affected by sharply changing noise points, which is further proved by the experiments in Sec 4.5.1.

Region-level Feature Confidence: However, in a hands-on environment, robots might move a lot suddenly, causing the objects move fast between frames. Due to the high sensitivity of the optical flow tracking to fast moving, the point x_i^{t-1} is located inaccurately at times. Thus, feature confidence is necessary. Specifically, the forward-backward error [18], indicating the distance between the original point and its position after the forward and backward optical flow tracking, is employed to calculate feature confidence. Assuming that edge point \mathbf{x}_i^t is the backward optical flow point of a_i^{t-1} from images I^{t-1} to I^t , the forward-backward error of point x_i^t is formulated as the Euclidean distance from \mathbf{x}_i^t to x_i^t : $\lambda_i^t = \|\mathbf{x}_i^t - \mathbf{x}_i^t\|$. Thus, the feature confidence of proposal b_j^t is stated as:

$$\Lambda_j^t = 1/\sqrt{(1/N_j^t \sum_{x_i^t \in b_j^t} \lambda_i^t)} \quad (8)$$

If the point a_i^{t-1} calculated by tracking x_i^t is inaccurate, the backward-tracking point \mathbf{x}_i^t would be far away from the original point x_i^t , which leads to a large distance λ_i^t . In this case, the confidences of the proposals containing point x_i^t are decreased.

3.5 Appearance-Geometry Fusion Model

To locate the obstacles, we re-score the proposals by a joint model, i.e., appearance-geometry fusion model (AGFM), that consists of two parts: The first appearance-geometry regressor (AGR) distinguishes UO and obstacle which fully uses the proposed features. The second appearance regressor (AR) handles proposals affected by fast motion.

3.5.1 Model Structure

Our AGFM employs a random forest structure [7] to achieve an accuracy-efficiency trade-off. In detail, the regressor $\text{AGR} = \{f_k^{ag} | k = 1, \dots, K^{ag}\}$ consists of K^{ag} decision trees, and each one contains several internal nodes and leaf nodes. With the same structure, the regressor AR contains K^a trees, and f_k^a for each tree in AR. In the training phase, the extracted proposals are taken as training samples. Each internal node selects a feature and splits the samples into two parts, each leaf node stores the mean label of proposals reaching on it during training. In the inference phase, by inputting the proposals to root nodes of these trees, each proposal is separated by internal nodes, and passed to the left or right pathway until a leaf node is reached.

3.5.2 Training Data

For our AGFM, all the samples, i.e., the proposals $\{b_j^t\}$, are predefined as (i) floor or (ii) background. The proposal whose more than 40% pixels are occupied by floor or obstacle is marked as floor, otherwise background. Similar to [45], only the proposals of the floor are selected for training. In terms of the features, as depicted in the last section, each training sample of AGR is represented by the unified features, that is, a sample b_j^t corresponds to 19-dimensional feature vector $v_j^t \in \mathbb{R}^{19}$, as shown in Table 1. The detailed formulation of each feature channel can be found in Appendix D. In contrast to AGR, the sample of AR is represented by the first 17 dimensions of v_j^t , i.e., denoted as $\bar{v}_j^t \in \mathbb{R}^{17}$. With the training samples mentioned above, AGR and AR are trained to regress the Intersection over Union (IoU) between a proposal and obstacle segmentation.

3.5.3 Prediction

Following the inference process of decision tree [7], each decision tree inputs the 19-dimensional feature vector v_j^t ,

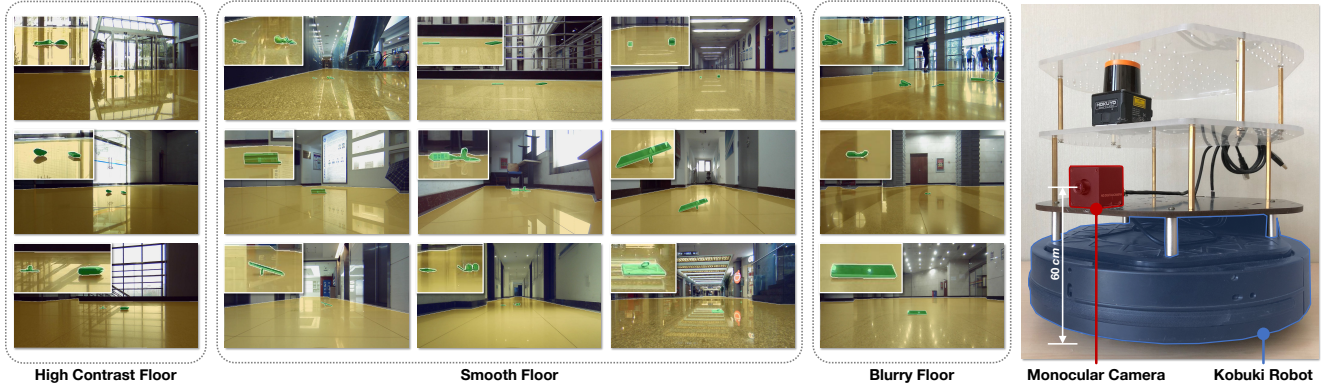


Fig. 5 Exemplar images and pixel-level annotations taken from the proposed dataset. The floor is marked in yellow, and green for obstacle. The images are zoomed in to clearly show these obstacles. The right-side image exhibits our platform, a Kobuki robot that provides the odometer data.

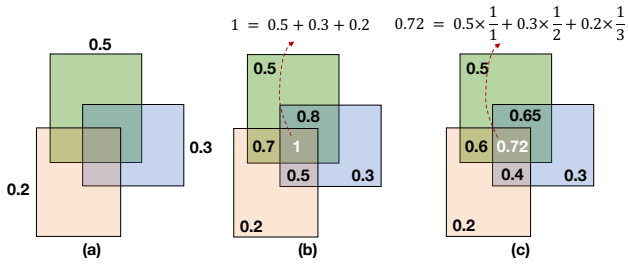


Fig. 6 The illustration of weight-decayed probability generation. (a) three bounding boxes and their confidences. (b) generated probabilities without decayed weights. (c) proposed generation scheme.

and outputs the predicted IoU between proposal b_j^t and obstacle. The outputs of the two regressors are stated as:

$$\begin{aligned} \text{AGR}(b_j^t) &= 1/K^{ag} \sum_{i=1}^{K^{ag}} f_i^{ag}(v_j^t) \\ \text{AR}(b_j^t) &= 1/K^a \sum_{i=1}^{K^a} f_i^a(\bar{v}_j^t) \end{aligned} \quad (9)$$

Since some objects move fast in the view of robot, the failed optical flow tracking leads to a low feature confidence Λ_j^t . Thus, confidence Λ_j^t indicates if an object moves fast in the field view of robot. Based on this, the entire prediction is formulated as follows:

$$F(b_j^t) = \begin{cases} \text{AGR}(b_j^t), & \text{if } \Lambda_j^t < \tau_{gc} \\ \text{AR}(b_j^t), & \text{otherwise} \end{cases} \quad (10)$$

where τ_{gc} represents the confidence threshold used to select between AR or AGR for scoring bounding boxes. The geometry features express the difference between obstacle and UO, which accordingly helps AGFM to limit the score of the UO. Furthermore, since the appearance features are independent of the fast motion, AR makes up for the inaccuracy of AGR in fast motion.

3.5.4 Weight-decayed Scheme for Obstacle-occupied Map

After scoring all proposals by AGFM, the scores of top τ_b proposals, i.e., $\{F(b_j^t) | b_j^t \in \hat{\mathcal{B}}\}$, are accumulated to con-

struct an obstacle-occupied probability map, where $\hat{\mathcal{B}}$ denotes the set of top τ_b proposals. However, the previous methods [45,46] often obtain a large number of low-score proposals in a similar location, even if using Non-Maximum Suppression (NMS) to remove many low-score proposals. This issue makes the probability map concentrates too much on parts of obstacles, instead of whole obstacles.

To segment obstacles as completely as possible, we propose a weight-decayed generation scheme for obstacle-occupied probability map. In more detail, assuming that the proposals covering a certain pixel p are sorted in descending order of score, and the numerical order of the j -th proposal is represented as r_j^p , each proposal has a weight that is an reciprocal of order r_j^p . The obstacle-occupied probability of pixel p is formulated as:

$$P(\text{pixel}(p)) = \frac{1}{N^P} \sum_{b_j^t \in \hat{\mathcal{B}}} \mathbf{1}(p, b_j^t) \times F(b_j^t) \times \frac{1}{r_j^p} \quad (11)$$

where $\text{pixel}(p)$ denotes the coordinate of pixel p . $\frac{1}{N^P}$ denotes the normalization term. $\mathbf{1}(p, b_j^t)$ is an indicator. If $p \in b_j^t$, then $\mathbf{1}(p, b_j^t) = 1$. Otherwise, $\mathbf{1}(p, b_j^t) = 0$. As shown in Fig. 6 (b)(c), when pixel p is enclosed by proposal b_j^t , the proposal's score $F(b_j^t)$ is added to this pixel. Finally, each pixel in this map P indicates the probability to be obstacle. By setting a parameter to segment the map into a mask, the obstacle is eventually located by the segmented mask. Since the low-score proposal has lower weight, the proposed scheme avoids an extremely high response in parts of obstacles caused by a large number of proposals with similar locations, as shown in Fig. 6 (c).

4 Evaluation

4.1 Obstacle on Reflective Ground (ORG) Dataset

To evaluate the proposed method, a novel dataset for Obstacle on Reflective Ground (ORG) is proposed, which consists of 15 different indoor scenes and 42 types of obstacles, as

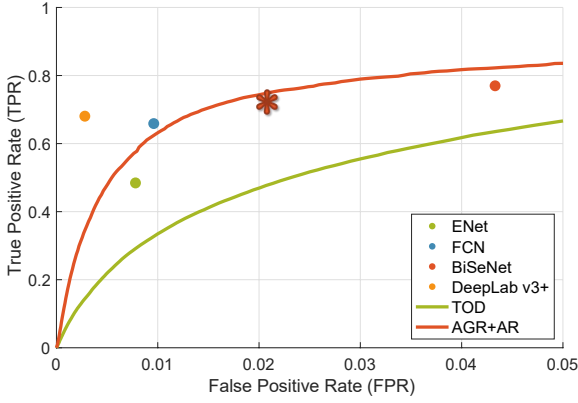


Fig. 7 The pixel-level ROC of different methods. The performance when FPR is 2% is used to compare with other methods in the instance-level performance comparison, which is indicated by \star .

shown in Fig.5. The scenes of this dataset are mainly divided into three types: high contrast floor, smooth floor, and blurry floor. In these scenes, the ground of each scene has varying UOs to evaluate the robustness of the algorithm. Even, low illumination, different patterned floor, and small obstacle also appear in this dataset. In terms of the obstacle, the obstacles with different sizes and materials are contained in the dataset, which have low height and therefore fail to be perceived by 2D LiDAR.

The ORG dataset contains 223 monocular video sequences in total. In each sequence, several additional information is provided, i.e., a set of pose information given by the wheel odometer, the pixel-level annotation of the ground and the obstacles. Each video contains about 10 to 20 frames annotated, and each object has a different category ID. In terms of the dataset structure, the dataset is split into a Train set (117 videos/1711 images) and a Test set (106 videos/1709 images), each of which contains completely different scenes. As the platform shown in Fig. 5, the monocular camera is fixed with a height of 60 cm from the ground and features a focal length of 2.8 mm, a spatial resolution of 1920×1080, and a pixel size of 8 bit. The Kobuki robot provides the wheel odometer for our method.

4.2 Metrics

To evaluate our method fairly, a pixel-level metric and an instance-level metric are employed to quantitatively analyze the performance of all methods, respectively.

Pixel-level Metric: Referring to other discovery methods [45,36,37], we employ the Pixel-level Receiver-Operator-Characteristic (ROC) curve that measures the True-Positive Rate (TPR) under different False-Positive Rate (FPR):

$$TPR = \frac{TP}{GT_{obs}}, \quad FPR = \frac{FP}{GT_{ground}} \quad (12)$$

Table 2 Instance-level results of all methods, The thresholds are set according to FPR of 2%. Bold numbers indicate the 1-st results, and underlined numbers for the 2-nd results.

Method	ITPR↑ /% (Instance)	MIFP↓ (Instance)	TPR↑ /% (Pixel)	FPR↓ /% (Pixel)
FCN [40]	54.29	1.15	48.43	<u>0.78</u>
ENet [35]	51.43	55.22	64.88	0.96
BiSeNet [47]	<u>64.85</u>	5.37	76.98	4.33
DeepLab v3+ [4]	52.02	2.48	68.05	0.28
TOD [45]	61.72	3.46	47.00	2.01
AGR+AR	77.46	<u>1.87</u>	<u>74.23</u>	1.98

where TP denotes the number of obstacle pixels which are correctly discovered, and FP corresponds to the number of floor pixels which are predicted as obstacle. GT_{obs} is the total number of pixels inside the obstacle proposal, and GT_{ground} is the total number of pixels labeled as the floor.

Instance-level Metric: Referring to [36,37], since the pixel-level metric suffers from the bias toward object instances that cover large areas in the images, we employ the instance-level metric to evaluate our method, namely Instance-level True-Positive Rate (ITPR) and Mean Instance-level False Positives (MIFP). Inspired by [36,37], each consecutive area in the generated segmentation is an instance. An obstacle instance is marked correctly detected if more than 50% of the pixels of the instance is obstacle. An instance is considered as incorrectly detected if its overlap with the floor area is larger than 50%. Based on these definitions, ITPR is defined as the fraction of obstacle instances in the ground truth map which are correctly detected. MIFP is defined as the mean incorrectly detected instances per frame.

$$ITPR = \frac{iTP}{N_{obs}}, \quad MIFP = \frac{iFP}{N_{img}} \quad (13)$$

where iTP denotes the total number of correctly detected obstacles, and iFP corresponds to the total number of false-positive instances. N_{obs} is the total number of obstacle instances, and N_{img} is the number of frames in the test set.

4.3 Experimental Setup

Several variants are compared to illustrate the effectiveness of our method. AGR denotes the geometry-appearance fusion regressor F^{ag} , and AR denotes the appearance based regressor F^a . AGR+AR denotes our method that uses AR to handle the proposal with fast moving. Note that the tree numbers K^a and K^{ag} are set to 50. In addition, τ_e is set to 0.8, τ_{gc} is set to 0.1, and τ_b is set to 50. When discovering the obstacles in image I^t , q is set to ensure that the distance the robot moves $\|\Delta C\|$ is larger than 20 cm.

Furthermore, due to lacking the source code of obstacle discovery methods [36,37,11], we cannot compare our

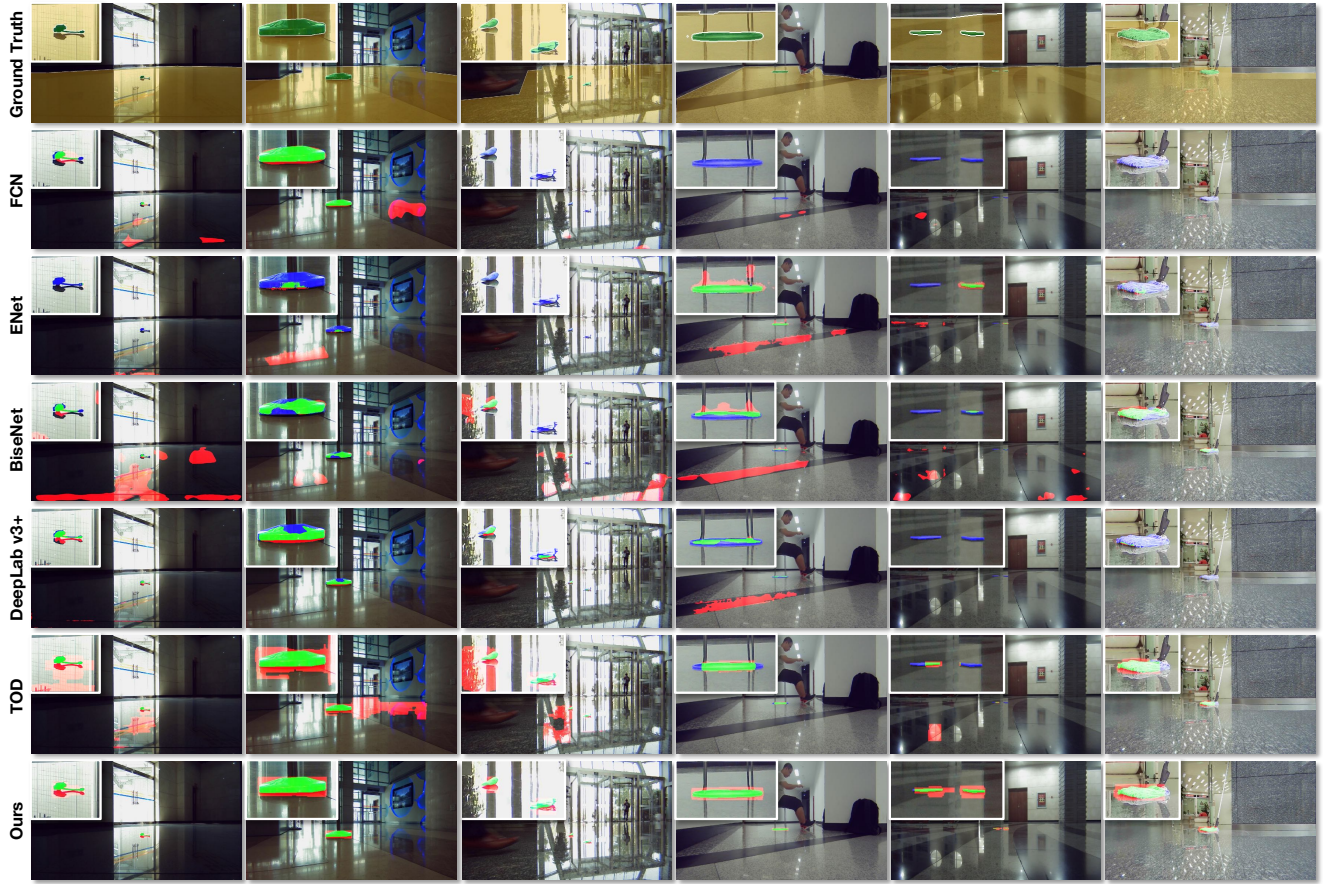


Fig. 8 Qualitative results for obstacle discovery in the indoor environment. In the ground truth, the floor is marked in yellow, and green for the obstacle. The images could be zoomed in to clearly observe these tiny obstacles. In the results, the true-positive pixel of obstacles are marked in green, blue for false-negative pixels, and red for false-positive pixels.

method with them. To make a full comparison with the state-of-the-art methods, we are inspired by the existing CNN-based obstacle discovery methods [37, 11, 16, 30, 41], and employ several classic semantic segmentation networks to discover obstacles, namely FCN [40], ENet [35], BiSeNet [47], and Deeplab v3+ [4]. For these networks, the parameters of the optimizer are set to the same as those proposed in their papers. Note that ENet uses Adam [19] as the optimizer, while other methods take stochastic gradient descent (SGD) [20]. To adapt these networks to our hardware environment, we conduct several modifications when training these networks. For clarity, ResNet-50 [13] is employed as the backbone for Deeplab v3+. The batch sizes of FCN and Deeplab v3+ are set to 6 and 4, respectively. During training, we randomly crop the image into a fixed size as input. Specifically, the resolution for training FCN or ENet is 960×540 , a half resolution of the original image, while BiSeNet and Deeplab v3+ take the crop size 1024×1024 and 513×513 respectively. For all the methods, we use full resolution image as input during testing. All networks are trained on two NVIDIA GeForce GTX 1080 Ti GPUs. In

addition, the base method, namely segmenting by detection framework [45], is denoted as TOD in the experiment.

4.4 Result

4.4.1 Quantitative Result

Fig.7 depicts the pixel-level ROC of different methods. One can see that AGR+AR outperforms TOD by a large margin. Especially, by jointly utilizing AGR and AR, the FPR of our method is 0.85% when the TPR is 60%, a drop of about 76.94% (from 3.69% to 0.85%) compared to TOD, which proves the effectiveness of our method in avoiding the mis-detection of the UOs. Besides, our method even outperforms ENet [35] and BiSeNet [47] and achieves a comparative pixel-level performance to Deeplab v3+ [4] and FCN [40]. Note that, since the proposed method segments obstacles by the detection, most of the mis-detected pixels are in a rectangular area that includes the irregular-shape obstacle. More details can be found in the instance-level result and the qualitative results.

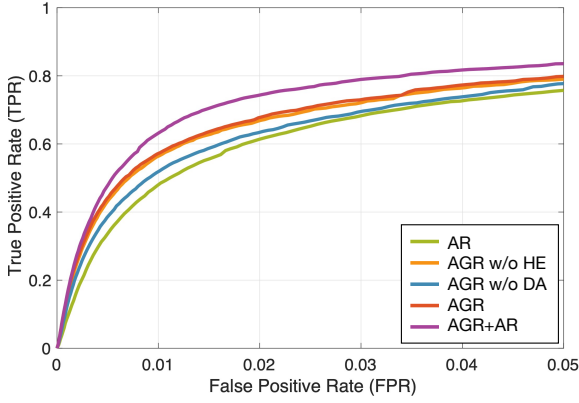


Fig. 9 The pixel-level ROC of models using different features.

Table 3 Instance-level results of models using different features. Bold numbers indicate the 1st results, and underlined numbers for the 2nd results. w/o means ‘without’, HE for homography error, DA for deviation angle.

Method	ITPR↑ /% (Instance)	MIFP↓ (Instance)	TPR↑ /% (Pixel)	FPR↓ /% (Pixel)
AR	64.47	1.86	61.39	2.00
AGR w/o HE	76.46	2.07	66.92	2.00
AGR w/o DA	73.24	1.91	63.28	2.00
AGR	<u>76.83</u>	1.94	<u>67.67</u>	<u>1.99</u>
AGR+AR	77.46	<u>1.87</u>	74.23	1.98

Table 2 shows the instance-level results of different methods. For TOD and our method, the ITPR, MIFP, and TPR are determined by setting the FPR to 2%. Observably, our method achieves 15.74% ITPR improvement over TOD and obtains 1.59 fewer false detection instances per picture than TOD. Besides, ENet [35] performs poor at TPR, ITPR, and MIFP. BiSeNet [47] obtains high TPR and ITPR at the cost of poor FPR and MIFP. The reason is that the lightweight spatial branch of BiSeNet [47] fails to suppress the high-frequency visual information and instead retains it as noise, while the context branch is unable to eliminate the noise when fusing two branches’ features. As a result, more false positives are generated in high-contrast areas, as observed in Fig. 8. On the contrary, DeepLab v3+[4] and FCN [40] achieve better FPR and MIFP but poor TPR and ITPR. Compared to these CNN-based methods, although the FPR of our method is not the lowest one, our method achieves the best performance of ITPR, and the second-best performance of MIFP. Thus, our method achieves a better trade-off. Besides, this comparison also indicates that most of the false-positive pixels of our model is adjacent to the pixels of obstacle, not scattered to the area of UOs, which is consistent with the visualization in Fig. 8. Thus, our method avoids the wrong detection of UOs better than all others.

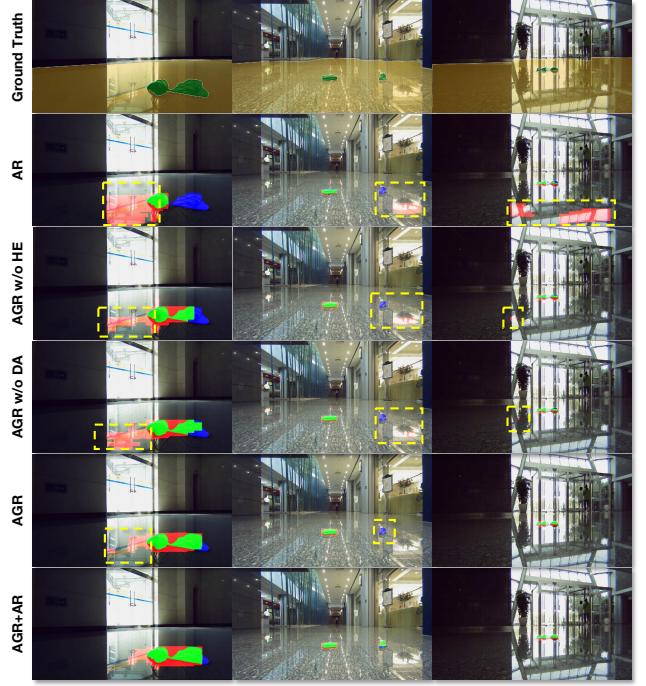


Fig. 10 The visualized result of different variants. In the ground truth, the floor is marked in yellow, and green for the obstacle. In the results, the true-positive pixel of obstacles are marked in green, blue for false-negative pixels, and red for false-positive pixels. The yellow dotted boxes enclose the wrong prediction.

4.4.2 Qualitative Result

Fig.8 depicts the qualitative result of different methods in indoor scenarios with different UO in the Test subset. Note that, we take the obstacle segmentation of TOD and our method by fixing the FPR to 2%, which is consistent with the setting of Table 2. In the visualization of results, the green pixels denote the true-positive pixels of obstacle, the red pixels for the false-positive pixels, and the blue pixels for the false-negative pixels. Besides, in the ground truth, the green pixels and the yellow pixels indicate obstacle and floor, respectively. The area near the obstacle is zoomed in for clear observation.

The first column shows a bunch of data lines on a smooth floor, where the reflection causes an extremely large color difference. It can be seen that all other methods are confused by the reflective door. The CNN-based segmentation methods suffer from a lot of false-positive pixels, and TOD is also degraded due to the wrong segmentation of the reflective door handle. Although our method generates a few false positives nearby the obstacle, no false positive exists in the area of UOs. The reason is that the proposed features well represent the obstacle and the reflection. The second column shows a guardrail base in the lobby. The floor reflects some fuzzy highlights due to its matte surface. FCN wrongly takes the fake TV as the obstacle. ENet, BiSeNet,

Table 4 Instance-level results when using various τ_b in generating probability map. The thresholds are set according to FPR of 2%. Bold numbers indicate the 1-st results, and underlined numbers for the 2-nd results.

Method	ITPR \uparrow /% (Instance)	MIFP \downarrow (Instance)	TPR \uparrow /% (Pixel)	FPR \downarrow /% (Pixel)
Original [45]	76.23	2.46	65.52	2.00
Top 10	71.47	1.72	70.30	2.00
Top 50	77.46	<u>1.87</u>	<u>74.23</u>	1.98
Top 100	78.37	1.97	74.36	<u>1.99</u>
Top 200	<u>78.00</u>	2.05	73.10	<u>1.99</u>
Top 500	76.55	2.19	71.35	2.02

and TOD not only generate many false-positive pixels, but also fail to segment the real obstacle. In contrast, our method corrects the misclassification of TV and highlight reflection, and segments the complete obstacle. The third column depicts a tube of hand cream and a pair of glasses on the floor of the office building. Due to the extremely smooth surface, the floor reflects all the scenes in the real world, making it hard to distinguish the real obstacles. FCN and ENet fail to discover the two obstacles. BiSeNet discovers one of the obstacles, but suffer from the wrong classified pixels inside the area of reflection. DeepLab v3+ successfully discovers them all, but misses several obstacle pixels. TOD is unable to distinguish the reflection and the real obstacle. Compared to them, our method segments all the obstacles and avoids false-positive pixels in the area of reflection. The fourth column shows a round stool on the ground with multiple textures. The four thin legs and the base of this stool can hardly be captured by the LiDAR of the robot. The reflection is not obvious, but the different texture obviously confuses all the CNN-based methods. TOD performs poorly due to the inaccurate segmentation of the obstacle, but our method avoids all the problems. The fifth column shows two cellphones. The floor is full of reflections and multiple textures. And the dim illumination makes it harder to recognize the cellphones on the ground. Intuitively, FCN, ENet, BiSeNet, and TOD cannot discover all the cellphones, while failing to avoid the false positives. DeepLab v3+ avoids the wrong classification to the obstacle pixel, but is unable to discover the two obstacles. By contrast, our method avoids the false positives, and discovers the two cellphones. The last column shows a mop leaning against the wall. Due to the similar appearance with the ground, almost all the methods cannot discover it totally. However, our method can capture this mop completely.

In general, it can be seen from all the above results that our method successfully avoids the confusion brought by the ground reflection, and discovers the obstacles better than other methods. It is noteworthy that our method obtains a few false-positive obstacle pixels in the rectangle region en-

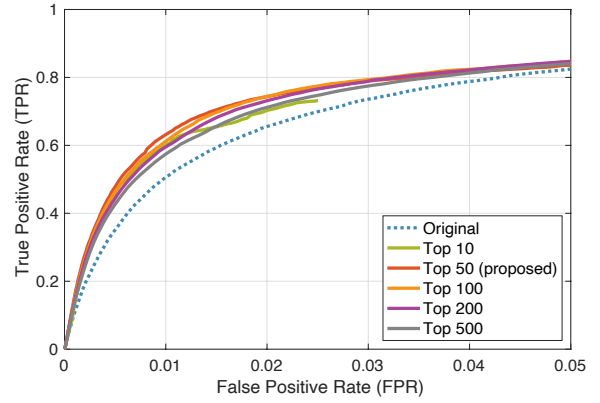


Fig. 11 The pixel-level ROC of variants with different numbers of top proposals in generating the probability map.

closing the obstacle. The reason is that we segment the obstacle by detection, the region proposal inevitably covers these false-positive pixels. Fortunately, due to connection with the obstacles, these few false-positive pixels do not affect the robot navigation at all. Moreover, this observation also exposes the highest ITPR and low MIFP mentioned in the last subsection.

4.5 Analysis and Discussions

4.5.1 Effectiveness of appearance-geometry feature

In this paper, we propose a unified appearance-geometry feature representation for bounding box proposals. The effectiveness of the proposed feature is evaluated in the comparisons in Fig. 9 and Table 3. Intuitively, AR only employs the appearance features, and suffers from the low ITPR and TPR. The reason is that the lack of geometry feature leads to a reduced ability to distinguish obstacles from UOs. By additionally using the homography error (HE), the TPR is further improved by 1.89%, and 8.77% for ITPR. By additionally using the deviation angle (DA), the TPR is further improved by 5.53%, and 11.99% for ITPR. The two results demonstrate the effectiveness of homography error and deviation angle, respectively. By jointly using the two features, the improvement of TPR is 6.28% (comparing AGR and AR), and that of ITPR for 12.36%. Besides, the model using AGR and AR, namely the proposed model, achieves the best performance, 6.56% TPR higher than the second-best result, and 0.63% ITPR higher than the second best result. The reason is that, by using AR to avoid the misclassification of proposals with low confidence, the proposed model better utilizes the appearance-geometry feature representation.

Fig. 10 shows the visualized results of the mentioned variants. The first column shows a wired mouse on the reflective ground with extreme illumination change. Due to the insufficient expression of feature to reflection, AR, AGR

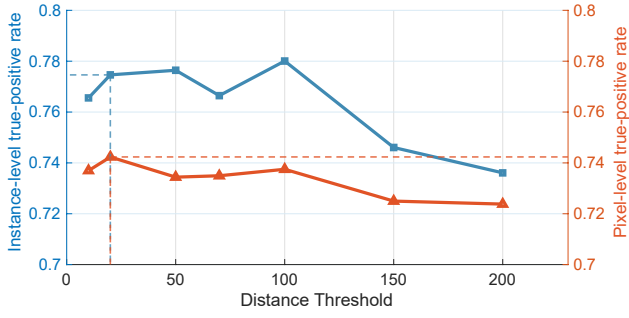


Fig. 12 The pixel-level and instance-level results of variants with different distance threshold when determining q . We select the results when FPR is fixed to 2%.

w/o DA, and AGR w/o HE wrongly segment the reflection. AGR performs better than the prior variants, and AGR+AR has the best result. The second column shows a watch and a wallet on the floor of the mall. The size of the watch is small, which makes almost all the variants fail to capture it. AGR+AR not only segments the watch, but also avoids the false positives on the floor. The third column depicts a mouse on the ground with complex reflection. AR is completely incapable of avoiding false detection of reflection. AGRs with only HE or DA also cannot completely avoid false detection. AGR and AGR+AR perform well in this scene.

4.5.2 Different settings of weighted-decay probability map

In this paper, the weighted-decay probability generation scheme is proposed to avoid that the final segmentation concentrates too much on parts of obstacles. To validate its effectiveness, the original generation scheme in [45, 46] is employed to compare with the proposed method. It firstly uses NMS to remove redundant proposals, then accumulates the top 50% proposals to generate the probability map. Besides, we take five number of top proposals in the generation process to find the best parameter.

Fig. 11 and Table 4 show the results when using different parameters in generating the probability map. Firstly, by comparing the original generation method and the proposed one, it can be seen that our method fully outperforms the original generation method. Secondly, as the number of top proposals increases, the pixel-level accuracy first improves and then decreases, and the instance-level accuracy follows the same way. Intuitively, top 50, top 100, and top 200 achieve better performances than others. Eventually, we consider that the optimal number of top proposals is 50, which is brought by two reasons:

- It performs best when FPR ranges from 0% to 2%, which makes our model obtains fewer wrong segmentation pixels.

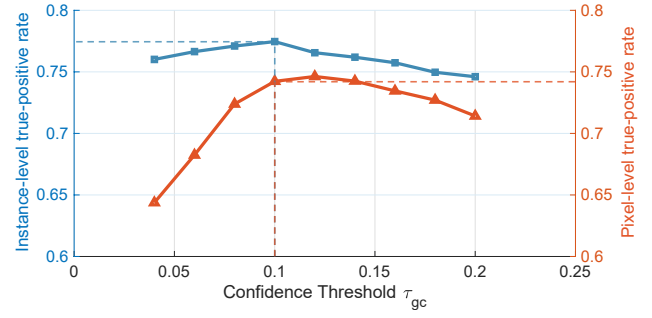


Fig. 13 The pixel-level and instance-level results of variants with different τ_{gc} . We select the results when FPR is fixed to 2%.

- Although its instance-level accuracy is slightly poorer than top 100 and top 200, it obtains fewer false-positive instances.

4.5.3 Different settings of distance threshold

In our method, the parameter q , which represents the frame interval of two consecutive images, is set to ensure that the distance the robot moves is larger than a certain distance threshold. Fig. 12 displays the pixel-level and instance-level results of the model with different distance thresholds. The results illustrate that the TPR and ITPR decrease significantly when the threshold is lower than 20 or larger than 100. The reason is that excessive movement would result in a large change in the camera view. Furthermore, there are no significant changes in accuracy when the threshold ranges from 20 to 100, and the model that uses a threshold of 20 achieves the best TPR and the third best ITPR.

4.5.4 Different settings of confidence threshold τ_{gc}

The variable τ_{gc} represents the confidence threshold used to select between AR or AGR for scoring bounding boxes. Fig. 13 illustrates the pixel-level ROC curve and instance-level results obtained by varying τ_{gc} from 0.04 to 0.20. Notably, setting τ_{gc} to 0.1 yields the highest ITPR, lowest MIFP, and third-highest TPR. Furthermore, when τ_{gc} is set below 0.08, TPR and ITPR decrease significantly due to insufficient usage of AGR, and when τ_{gc} exceeds 0.12, TPR and ITPR continue to decrease due to the limited representation ability of the low-confidence geometry feature.

4.5.5 Robustness to motion blur noise

In actual application, when the robot moves on the ground, the mounted camera is inevitably impacted by the fast motion and the violent shaking. In these cases, the long exposure time of the camera causes the blurred image, which presents the obstacle discovery method with challenges. To verify the robustness of our method to the motion blur, the

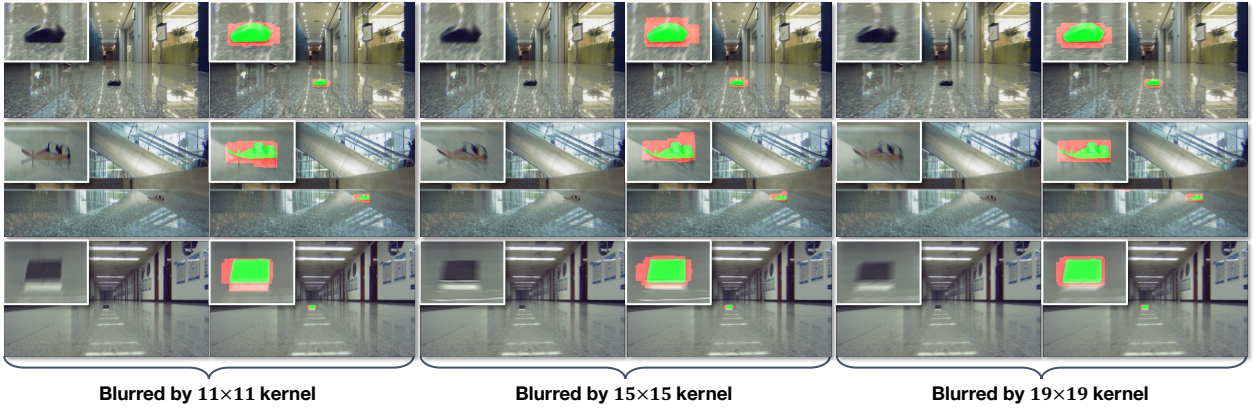


Fig. 14 The visualized results of variants adding different levels of the motion blur. In the results, the true-positive pixel of obstacles are marked in green, blue for false-negative pixels, and red for false-positive pixels.

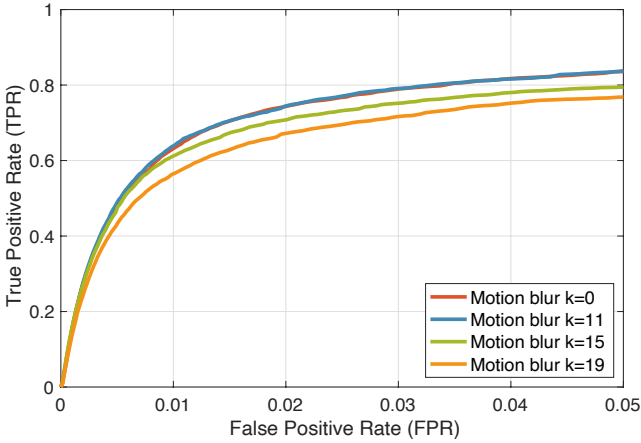


Fig. 15 The pixel-level ROC of variants impacted by different level of motion blur.

random motion blurring scheme [16] is employed to introduce the noise in the testing phase, which uses the point spread function [17] to simulate the motion blur of image. More specifically, we utilize a blurring filter with $k \times k$ size and θ^* direction to blur the image before being fed into our method, where k determines the intensity of motion blur. According to the resolution of the proposed dataset and our experience, most of the blur kernels are within 15 pixels in size. Thus, the kernel size used for the testing is 11, 15, and 19 pixels, and the blur direction ranges from 0° to $\pm 180^\circ$ randomly. Note that, our model is trained on the image without blurring.

Fig. 15 and Table 5 illustrate the result of our method affected by the motion blur. Intuitively, when the size of the blurring filter is 11×11 , the pixel-level accuracy is barely affected, and the instance-level accuracy is reduced by 8.62%. The reason is that several small obstacles are smaller than the blurring kernel, and thus their visual information is lost. The failure to discover these extremely tiny obstacles has a small impact on pixel-level accuracy, but has a great impact on instance-level accuracy. When the size of the blurring fil-

Table 5 Instance-level result of models impacted by different levels of motion blur. The thresholds are set according to FPR of 2%. Bold numbers indicate the 1-st results, and underlined numbers for the 2-nd results.

Kernel size for motion blur	ITPR \uparrow /% (Instance)	MIFP \downarrow (Instance)	TPR \uparrow /% (Pixel)	FPR \downarrow /% (Pixel)
$k = 0$	77.46	<u>1.87</u>	74.23	1.98
$k = 11$	<u>68.84</u>	1.97	<u>74.37</u>	<u>1.99</u>
$k = 15$	64.04	1.97	70.81	2.01
$k = 19$	57.32	1.85	67.21	2.00

ter is enlarged to 15×15 , both the instance-level and pixel-level accuracy are further reduced by around 4%. When the intensity of blurring is maximized, the accuracy is further reduced. The above comparison proves that although our method suffers from the noise given by motion blur, the accuracy is still comparable to other methods, which proves the robustness of our method.

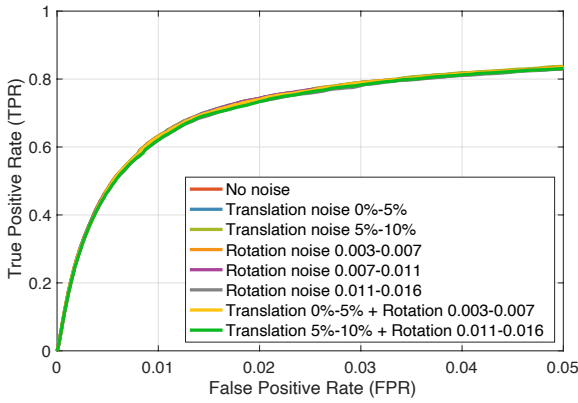
Fig. 14 shows three scenes blurred by different kernels. The first row depicts a wireless mouse. It can be seen that, since it is close to our robot, this mouse can be discovered well in all intensities of motion blur. The second row shows a pair of glasses, it is farther from the robot than the mouse in the first row. Observably, even if some false-positive pixels are generated when using 19×19 kernel, the pair of glasses can be completely segmented. The last row shows a pad, which is far from the robot. Intuitively, the visual information is largely blurred. Our method still successfully discover this obstacle.

4.5.6 Robustness to robot motion noise

Since our method exploits the robot motion to figure the motion of the mounted camera, theoretically, the accuracy of the robot motion determines the performance of our method. Hence, we add the noise of different intensities on the robot

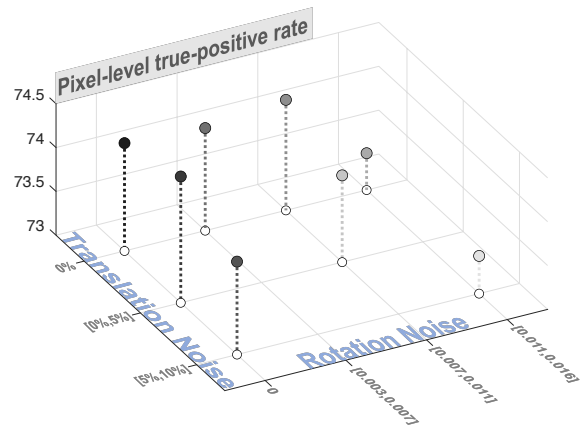
Table 6 Instance-level results when adding noise to the robot motion. The thresholds are set according to FPR of 2%.

Variants	Translation Noise	Rotation Noise	ITPR \uparrow /% (Instance)	MIFP \downarrow /% (Instance)	TPR \uparrow /% (Pixel)	FPR \downarrow /% (Pixel)
1	-	-	77.46	1.87	74.23	1.98
2	$[-5\%, 0] \cup [0, 5\%]$	-	77.69	1.89	74.44	2.02
3	$[-10\%, -5\%] \cup [5\%, 10\%]$	-	77.01	1.88	74.06	1.99
4	-	$[-0.007, -0.003] \cup [0.003, 0.007]$	77.51	1.87	74.17	2.01
5	-	$[-0.011, -0.007] \cup [0.007, 0.011]$	77.46	1.86	74.26	2.00
6	-	$[-0.016, -0.011] \cup [0.011, 0.016]$	77.10	1.85	73.42	2.00
7	$[-5\%, 0] \cup [0, 5\%]$	$[-0.011, -0.007] \cup [0.007, 0.011]$	77.41	1.87	73.99	2.00
8	$[-10\%, -5\%] \cup [5\%, 10\%]$	$[-0.016, -0.011] \cup [0.011, 0.016]$	76.64	1.85	73.43	2.01

**Fig. 16** The pixel-level ROC when adding different levels of the motion noise.

motion to validate the robustness of our method. In more detail, the robot motion can be decomposed into translation and rotation. Assuming that the time difference between the wheel odometer and the camera is less than 10 ms, the robot moves at a speed of 200 mm per second, and the maximum angular velocity is 90 degrees per second, the maximum angular error caused by the unsynchronized sensor is $90^\circ/s \times 0.01s = 0.9^\circ \approx 0.016rad$. Hence, we add the random noise of two intensities on the translation of robot, i.e., $[-5\%, 0] \cup [0, 5\%]$ and $[-10\%, -5\%] \cup [5\%, 10\%]$. In addition, we add the noise of three intensities on the rotation of robot, i.e., $[-0.007, -0.003] \cup [0.003, 0.007]$, $[-0.011, -0.007] \cup [0.007, 0.011]$, and $[-0.016, -0.011] \cup [0.011, 0.016]$. Finally, we add the maximum translation noise and rotation noise on the motion of the robot.

Table 6 and Fig. 16 show the effect of noise on the performance, and Fig. 17 presents the TPR in a three-dimensional coordinate system. Observably, as the noise added to the translation increases from 0 to 10%, both TPR and ITPR decrease slightly. Similarly, the TPR is also slightly down when rotation noise is lower than 0.011. The performance degradation is so insignificant that our model is barely affected. Finally, when both types of noises are increased simultaneously, with magnitudes greater than 5% and 0.011,

**Fig. 17** Pixel-level true-positive rate (TPR) of models disturbed by translation and rotation noise. The gray strength and the height of the points indicate the TPR of the variants.

respectively, both TPR and ITPR experience a more severe drop although the ITPR is only reduced by as much as 0.82% and 0.81% for the pixel-level accuracy. Overall, we infer that the confidence interval for translation noise is $[-5\%, 5\%]$, and the confidence interval for rotation noise is $[-0.011, 0.011]$.

4.5.7 Inference time analysis

Our method is implemented in MATLAB and runs on a PC with 16GB memory and an AMD Ryzen 2700 CPU. The current implemented version of our method fails to run in real-time with an input resolution of 1920×1080 . But we believe that the proposed method can achieve real-time performance in the C++ implementation with parallel computing. The inference time of each component is shown in Table 7. Specifically, our method consists of five parts, edge detection, proposal extraction, feature extraction, AGFM, and obstacle occupied map generation.

The first two parts employ the original occlusion edge detection [31] and object-level proposal [28], which account for the majority of time consumption. In the third part, the

Table 7 Inference time of our method and the basic methods.

Modules	Time of basic method	Basic method	Time
Edge Detection	3.411 s	[31]	3.411 s
Proposal Extraction	2.826 s	[28]	2.826 s
Feature Extraction	2.339 s	[45]	3.507 s
AGFM	0.053 s	[7]	0.058 s
Probability Map	0.007 s	[45]	0.016 s
ALL	8.636 s	-	9.818 s

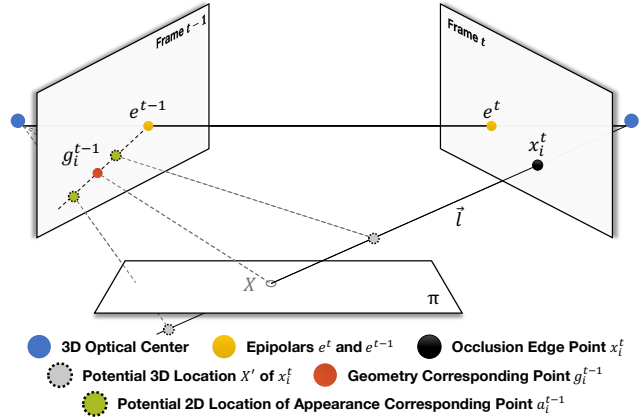
extraction of appearance features takes 2.339 seconds, while the extraction of geometry features takes 1.168 seconds. Finally, the random forest and obstacle occupied probability map generation take 0.058 seconds and 0.016 seconds, respectively. In summary, the first three parts take up most of the overall time cost, namely, 9.744 seconds, while the basic algorithms [31, 28, 45] is the core factor.

In principle, the proposed features can be calculated using integral images and can be computed in parallel with a time complexity of $\mathcal{O}(n)$, while part appearance features that cannot be computed using integral images have a time complexity of $\mathcal{O}(nl)$, where n denotes the number of proposal and l for pixel number of a proposal. Therefore, we infer that the reason for the current slower speed is the lack of optimized compilation processes and parallel computing.

5 Conclusion and Future Work

In this paper, a novel method is proposed to discover the obstacles on the reflective ground. To construct the feature representation that reveals the difference between obstacles and UOs, we first propose a ground detection scheme with pre-calibration, and introduce the ground-pixel parallax to represent the location of an occlusion edge point relative to the ground. Subsequently, by aggregating the parallax and the appearance cue, we propose a unified appearance-geometry feature representation for object bounding box proposal. Then, an appearance-geometry fusion model is proposed to locate the obstacle, meanwhile, avoids concentrating too much on parts of obstacles. Finally, a novel ORG dataset is proposed to evaluate our method, which is the first dataset focusing on the obstacle discovery on reflective ground.

This paper specializes in the problem of discovering obstacle on reflective ground. In the future, we are going to extend our method to address more challenges, such as small obstacle discovery, mirror obstacle discovery, and dynamic obstacle discovery, by a unified discovery framework. In this unified framework, monocular depth prediction [38, 2, 44] and dynamic objects tracking [50, 51] will be combined to further safeguard robot navigation. Additionally, to accelerate the running speed, we will adopt deep feature extraction

**Fig. 18** The ground-pixel parallax in two-view geometry.

and region proposal methods in future work, and implement parallel computation of features using C++.

Data Availability Statement

The code of the MATLAB implementation and datasets generated during the current study are available in the GitHub repository, <https://github.com/XuefengBUPT/IndoorObstacleDiscovery-RG>.

Acknowledgements This work was supported by the National Natural Science Foundation of China 62176098, 61703049, the Natural Science Foundation of Hubei Province of China under Grant 2019CFA022, the national key R&D program intergovernmental international science and technology innovation cooperation project under Grant No. 2021YFE0101600, the Beijing University of Posts and Telecommunications (BUPT) Excellent Ph.D. Students Foundation under Grant CX2020114.

Appendix

A Principle of Ground-pixel Parallax

In this section, we state the correctness of Equation 4 in the main manuscript, and prove that this equation determines the relationship between an observed point and the ground.

First of all, we give the proof of Equation 4 in the main manuscript. Supposing I^t and I^{t-1} denote two consecutive images from robot's view, π denotes the ground plane, and e^t, e^{t-1} denote the epipoles on the two images, which respectively are two intersection points of the two images and a line, i.e., the line connecting two camera optical centers, as the yellow points in Fig.18. $x_i^t = \{u_i^t, v_i^t, 1\}$ denotes an occlusion edge points of image I^t in its homogeneous form, and it is also the 2D projection of a 3D point X' on image I^t . A ray emitted from I^t 's optical center, denoted as \vec{l} , penetrates X' and x_i^t , and intersects the ground π at a 3D point X . In addition, with the representation of the main

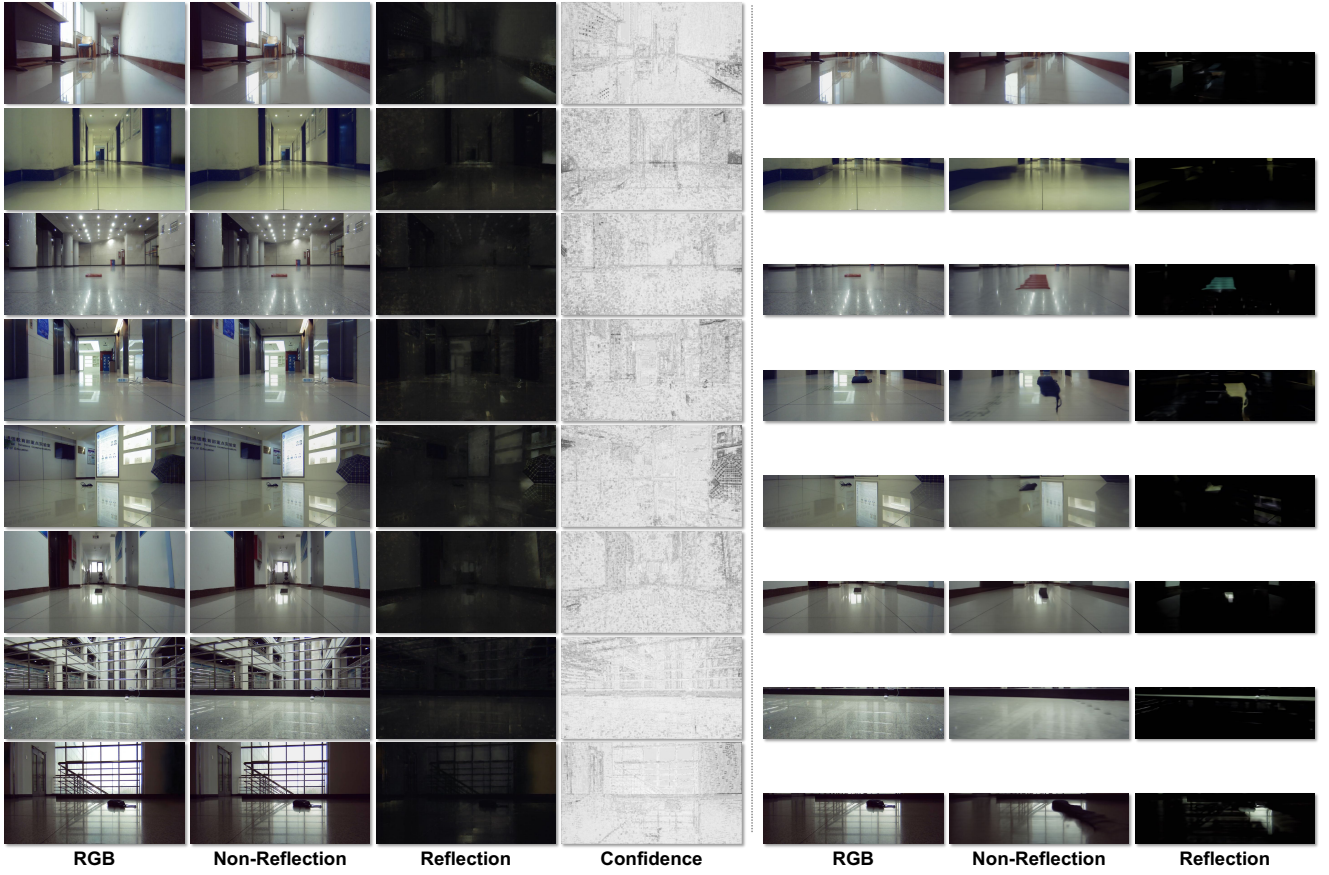


Fig. 19 Example results of the single frame reflection removal algorithm [8] and the multi-frame reflection removal algorithm [33] on multiple training scenes.

manuscript, the geometric and appearance corresponding points are denoted as g_i^{t-1} and a_i^{t-1} . According to Epipolar Constraints, the 3D line \vec{l} can be projected to image I^{t-1} to form a projected 2D line, denoted as $\vec{g}_i^{t-1}e^{t-1}$. Since the 3D point X' is on the 3D line \vec{l} , its projection on image I^{t-1} , namely a_i^{t-1} , is on the projected 2D line $\vec{g}_i^{t-1}e^{t-1}$. Hence, these 2D points a_i^{t-1} , g_i^{t-1} , and e^{t-1} are collinear. Based on this, 2D point a_i^{t-1} can be stated as:

$$a_i^{t-1} = g_i^{t-1} + \rho(g_i^{t-1} - e^{t-1}) \quad (14)$$

where ρ is a scalar. This equation can be reformulated as Equation 4 in the main manuscript.

Then, we discuss the reason that Equation 4 in the main manuscript can be used to distinguish the points above the ground and below the ground. Noteworthily, since g_i^{t-1} uniquely represents the projection of 3D ground point X , g_i^{t-1} can be considered as the dividing point, and all points on both sides of this 2D line are partitioned into two spaces, above the ground and below the ground, as shown in Fig.18. Hence, the space of point x_i^t can be determined by comparing a_i^{t-1} and g_i^{t-1} .

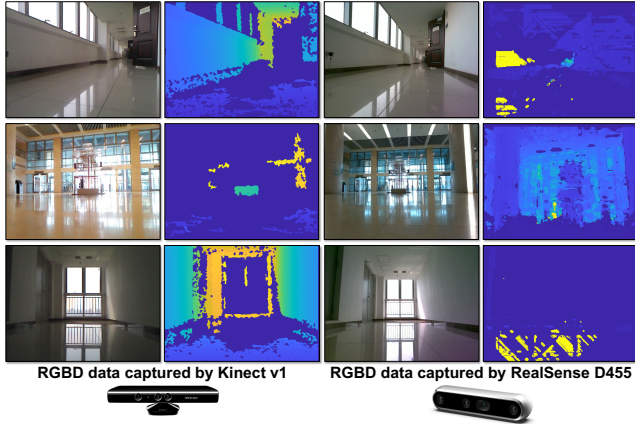
B Feasibility of Reflection Removal Methods

In scenes of reflection ground, it is intuitive to incorporate reflection removal approaches into the feature extraction part. Therefore, we employ the state-of-the-art single frame reflection removal algorithm [8] (ICCV 2021) and multi-frame reflection removal algorithm [33] (ECCV 2022), both of which have publicly available code. Their results on our dataset are visualized in Fig. 19. Note that since the real-world interfaces the reconstruction of multi-frame-based methods, we only conduct the reflection removal in the bottom half of the image, i.e., in the ground area.

Observably, both reflection removal methods are ineffective in removing reflections in our benchmark scenarios. In fact, they even damage the information of obstacles that needed to be detected. The single-frame-based method [8] produced confidence maps that failed to highlight the reflection, and the non-reflection images were almost identical to the original RGB images. Despite being free from the interface of the real world, the multi-frame-based method [33] still fails to eliminate reflections. The reason is that *both algorithms require strong textures of main object for adequate reconstruction, but the ground texture is too weak to be per-*

Table 8 Formulation of each feature representing a bounding box.

Category	Feature of bounding box	Formulation	index
<i>Edge cue</i>	Max edge response	$\max(\{x_i^t x_i^t \in \mathbf{b}\})$	1
	Proportion of most response	$1/N_j^t \sum_{x_i^t \in \mathbf{b}} (x_i^t = \arg \max_r \sum_{x_i^t \in \mathbf{b}} [x_i^t = r])$	2
	Average edge response	$1/N_j^t \sum_{x_i^t \in \mathbf{b}} x_i^t$	3
	Average edge response in inner ring	$1/\hat{N}_j^t \sum_{x_i^t \in \tilde{\mathbf{b}}} x_i^t$, where $\tilde{\mathbf{b}} = (u + w/4, v + h/4, w/2, h/2)$	4
<i>Pseudo distance</i>	Normalized area	$(w \times h) / (W \times H)$	5
	Aspect ratio of box	w/h	6
	X coordinate of the box center	$u + w/2$	7
	Y coordinate of the box center	$v + h/2$	8
	Width of box	w	9
	Height of box	h	10
<i>Objectness</i>	Occlusion-based objectness score	Referring to [28]	11
<i>Color</i>	Color standard deviation in the H channel	$\sqrt{1/hw \sum_{p \in \mathbf{b}} (\mathcal{H}(p) - 1/hw \sum_{p \in \mathbf{b}} \mathcal{H}(p))^2}$	12
	Color standard deviation in the S channel	$\sqrt{1/hw \sum_{p \in \mathbf{b}} (\mathcal{S}(p) - 1/hw \sum_{p \in \mathbf{b}} \mathcal{S}(p))^2}$	13
	Color standard deviation in the V channel	$\sqrt{1/hw \sum_{p \in \mathbf{b}} (\mathcal{V}(p) - 1/hw \sum_{p \in \mathbf{b}} \mathcal{V}(p))^2}$	14
	Color contrast in the H channel	$1 - \frac{hist_b^H \cdot hist_b^H}{\ hist_b^H\ _2 \ hist_b^H\ _2}$, where $\hat{\mathbf{b}} = (u - w/4, v - h/4, 2w, 2h)$	15
	Color contrast in the S channel	$1 - \frac{hist_b^S \cdot hist_b^S}{\ hist_b^S\ _2 \ hist_b^S\ _2}$, where $\hat{\mathbf{b}} = (u - w/4, v - h/4, 2w, 2h)$	16
	Color contrast in the V channel	$1 - \frac{hist_b^V \cdot hist_b^V}{\ hist_b^V\ _2 \ hist_b^V\ _2}$, where $\hat{\mathbf{b}} = (u - w/4, v - h/4, 2w, 2h)$	17
<i>Parallax</i>	Homography error	Referring to Sec. 3.4	18
	Deviation angle	Referring to Sec. 3.4	19

**Fig. 20** RGBD data captured by Kinect v1 and Realsense D455.

ceived. In contrast, the reflection has a stronger texture than ground, making it appear as the main object. Overall, existing reflection removal algorithms cannot be directly used in the scene with reflective ground, and even damage obstacle information.

C Feasibility of Depth Sensors

In recent years, multi-modal sensors have been increasingly popular in autonomous driving. Thus, we evaluate the us-

ability of radar or depth cameras in reflective ground environments. To this end, we collect depth data of several reflective scenes by two classical sensors, i.e., the structured light camera (Kinect v1, released in 2010, priced at \$150), the stereo camera (RealSense D455, released in 2019, priced at \$249). The exemplar RGBD data is visualized in Fig. 20. Intuitively, the depth data obtained by these cameras in reflective environments is of such low quality that it cannot be applied in reflective scenes. Specifically, the structured light camera generates many void areas on the ground plane, while the stereo camera matches corresponding pixels erroneously between the cameras and results in completely incorrect depth data. Clearly, both types of cameras are unsuitable for reflective ground scenes.

Furthermore, we conduct capability tests of laser sensor using a single-beam 360-degree LiDAR, which is illustrated in Fig. 21. The results illustrate that the single-beam LiDAR is not affected by reflections in all angles, which means that 3D LiDAR can obtain reliable depth information in reflective scenes. Unfortunately, although 3D LiDAR almost avoid the issue brought by reflective ground, it is too expensive to be deployed on a robot compared other depth sensor.

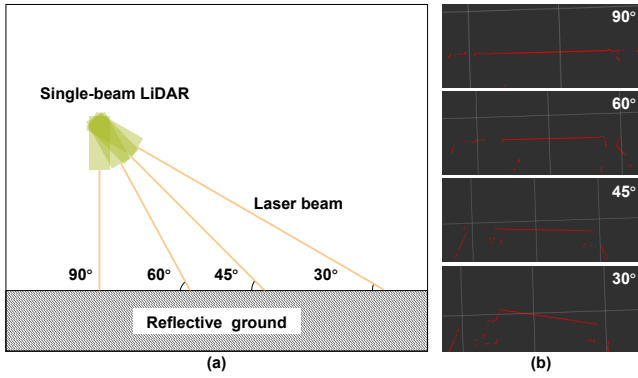


Fig. 21 Capability tests of laser sensor. (a) Side view of scanning ground with a single-beam LiDAR. (b) Laser scan obtained in different angle.

D Detailed Formulation of Feature Vector

To clearly represent the feature used in this paper, Table 8 shows the formulation of each feature channel. Note that, according to Sec. 3.4, b_j^t denotes the j -th bounding boxes in the t -th image, and its feature vector is denoted as v_j^t , which consists 19 channels grouped into five categories. To simplify the notation, we use b to represent the bounding box b_j^t , specified by its top-left pixel coordinates (u, v) and its width and height (w, h) . In Table 8, the notation \tilde{b} refers to the inner ring of the bounding box b , while \hat{b} represents the outer ring.

In the 2-nd channel, the notation $[\cdot]$ represents an indicator function that outputs 1 if the input is correct and 0 otherwise. In the 5-th channel, (W, H) denotes the width and height of the input image. For the 12-th - 17-th channels, the variables \mathcal{H} , \mathcal{S} , and \mathcal{V} correspond to the input image's HSV channels, and $\mathcal{H}(p)$ denotes the p -th pixel in the channel \mathcal{H} . Note that, the color contrast formulation involves the normalized histogram of box b 's H channel, represented as $hist_b^{\mathcal{H}}$, which is discretized into 18 bins denoted by $\{h_k^{\mathcal{H}}, k \in [1, 18]\}$, where k ranges from 1 to 18. The value of $h_k^{\mathcal{H}}$ is obtained as the sum over all pixels p in box b such that $h_k^{\mathcal{H}} = \sum_{p \in b} \left[\left\lfloor \frac{\mathcal{H}(p)}{360/18} \right\rfloor = k \right]$. Additionally, similar histograms $hist_b^{\mathcal{S}}$, $hist_b^{\mathcal{V}}$, $hist_b^{\mathcal{H}}$, $hist_b^{\mathcal{S}}$, $hist_b^{\mathcal{V}}$ are computed in the same way.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **33**(5), 898–916 (2011)
2. Bian, J.W., Zhan, H., Wang, N., Li, Z., Zhang, L., Shen, C., Cheng, M.M., Reid, I.: Unsupervised scale-consistent depth learning from video. *International Journal of Computer Vision (IJCV)* **129**(9), 2548–2564 (2021)
3. Broggi, A., Buzzoni, M., Felisa, M., Zani, P.: Stereo obstacle detection in challenging environments: The VIAC experience. In:

- IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2011)
4. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: *European Conference on Computer Vision (ECCV)* (2018)
5. Chen, T., Vemuri, B.C., Rangarajan, A., Eisenschenk, S.J.: Group-wise point-set registration using a novel cdf-based havrda-charvat divergence. *International Journal of Computer Vision (IJCV)* **86**(1), 111 (2009)
6. Conrad, D., DeSouza, G.N.: Homography-based ground plane detection for mobile robot navigation using a modified em algorithm. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2010)
7. Criminisi, A., Shotton, J., Konukoglu, E.: *Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning*. Now Publishers Inc (2012)
8. Dong, Z., Xu, K., Yang, Y., Bao, H., Xu, W., Lau, R.W.: Location-aware single image reflection removal. In: *IEEE/CVF International Conference on Computer Vision (ICCV)* (2021)
9. Dongfu, Z., Zheng, C.: The code of CamOdomCalibraTool. <https://github.com/MegviiRobot/CamOdomCalibraTool>
10. Ghodrati, A., Diba, A., Pedersoli, M., Tuytelaars, T., Van Gool, L.: Deepproposals: Hunting objects and actions by cascading deep convolutional layers. *International Journal of Computer Vision (IJCV)* **124**(2), 115–131 (2017)
11. Gupta, K., Javed, S.A., Gandhi, V., Krishna, K.M.: Mergenet: A deep net architecture for small obstacle discovery. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2018)
12. Hartley, R., Zisserman, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press (2003)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016)
14. Heng, L., Bo, L., Pollefeys, M.: Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2013)
15. Hoiem, D., Efros, A.A., Hebert, M.: Recovering occlusion boundaries from an image. *International Journal of Computer Vision (IJCV)* **91**(3), 328–346 (2011)
16. Hua, M., Nan, Y., Lian, S.: Small obstacle avoidance based on rgb-d semantic segmentation. In: *IEEE International Conference on Computer Vision Workshop (ICCVW)* (2019)
17. Jia, J.: Single image motion deblurring using transparency. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2007)
18. Kalal, Z., Mikolajczyk, K., Matas, J.: Forward-backward error: Automatic detection of tracking failures. In: *IEEE International Conference on Pattern Recognition (ICPR)* (2010)
19. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: *International Conference on Learning Representations (ICLR)* (2015)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems (NIPS)* (2012)
21. Kumar, S., Karthik, M.S., Krishna, K.M.: Markov random field based small obstacle discovery over images. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2014)
22. Li, H., Liu, Y., Ouyang, W., Wang, X.: Zoom out-and-in network with map attention decision for region proposal and object detection. *International Journal of Computer Vision (IJCV)* **127**(3), 225–238 (2019)

23. Lin, C., Jiang, S., Yueh-Ju Pu, Song, K.: Robust ground plane detection for obstacle avoidance of mobile robots using a monocular camera. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2010)
24. Lindeberg, T.: Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision (IJCV)* **30**(2), 117–156 (1998)
25. Lis, K., Nakka, K.K., Fua, P., Salzmann, M.: Detecting the unexpected via image resynthesis. In: *IEEE/CVF International Conference on Computer Vision (ICCV)* (2019)
26. Lu, R., Xue, F., Zhou, M., Ming, A., Zhou, Y.: Occlusion-shared and feature-separated network for occlusion relationship reasoning. In: *IEEE/CVF International Conference on Computer Vision (ICCV)* (2019)
27. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: *International Joint Conference on Artificial Intelligence (IJCAI)* (1981)
28. Ma, J., Ming, A., Huang, Z., Wang, X., Zhou, Y.: Object-level proposals. In: *IEEE International Conference on Computer Vision (ICCV)* (2017)
29. Malis, E., Vargas, M.: Deeper understanding of the homography decomposition for vision-based control. *Research Report RR-6303, INRIA* (2007)
30. Mancini, M., Costante, G., Valigi, P., Ciarfuglia, T.A.: J-MOD2: Joint monocular obstacle detection and depth estimation. *IEEE Robotics and Automation Letters (RA-L)* **3**(3), 1–1 (2018)
31. Ming, A., Wu, T., Ma, J., Sun, F., Zhou, Y.: Monocular depth-ordering reasoning with occlusion edge detection and couple layers inference. In: *IEEE Intelligent Systems (IS)*, vol. 31, pp. 54–65 (2016)
32. Ming, A., Xun, B., Ni, J., Gao, M., Zhou, Y.: Learning discriminative occlusion feature for depth ordering inference on monocular image. In: *IEEE International Conference on Image Processing (ICIP)* (2015)
33. Nam, S., Brubaker, M.A., Brown, M.S.: Neural image representations for multi-image fusion and layer separation. In: S. Avidan, G. Brostow, M. Cissé, G.M. Farinella, T. Hassner (eds.) *Europe Conference on Computer Vision (ECCV)* (2022)
34. Panahandeh, G., Jansson, M.: Vision-aided inertial navigation based on ground plane feature detection. *IEEE/ASME Transactions on Mechatronics (TMECH)* **19**(4), 1206–1215 (2014)
35. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: A deep neural network architecture for real-time semantic segmentation. In: *International Conference on Learning Representations (ICLR)* (2017)
36. Pinggera, P., Ramos, S., Gehrig, S., Franke, U., Rother, C., Mester, R.: Lost and found: detecting small road hazards for self-driving vehicles. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016)
37. Ramos, S., Gehrig, S., Pinggera, P., Franke, U., Rother, C.: Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. In: *IEEE Intelligent Vehicles Symposium (IV)* (2017)
38. Saxena, A., Chung, S.H., Ng, A.Y.: 3-d depth reconstruction from a single still image. *International Journal of Computer Vision (IJCV)* **76**(1), 53–69 (2008)
39. Sharp, G., Lee, S., Wehe, D.: ICP registration using invariant features. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **24**(1), 90–102 (2002)
40. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **39**(4), 640–651 (2017)
41. Singh, A., Kamireddypalli, A., Gandhi, V., Krishna, K.M.: Lidar guided small obstacle segmentation. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020)
42. Sun, L., Yang, K., Hu, X., Hu, W., Wang, K.: Real-time fusion network for rgb-d semantic segmentation incorporating unexpected obstacle detection for road-driving images. *IEEE Robotics and Automation Letters (RA-L)* (2020)
43. Xie, S., Tu, Z.: Holistically-nested edge detection. *International Journal of Computer Vision (IJCV)* **125**(1), 3–18 (2017)
44. Xue, F., Cao, J., Zhou, Y., Sheng, F., Wang, Y., Ming, A.: Boundary-induced and scene-aggregated network for monocular depth prediction. *Pattern Recognition (PR)* **115**, 107901 (2021)
45. Xue, F., Ming, A., Zhou, M., Zhou, Y.: A novel multi-layer framework for tiny obstacle discovery. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2019)
46. Xue, F., Ming, A., Zhou, Y.: Tiny obstacle discovery by occlusion-aware multilayer regression. *IEEE Transactions on Image Processing (TIP)* **29**, 9373–9386 (2020)
47. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: *European Conference on Computer Vision (ECCV)* (2018)
48. Zhou, J., Li, B.: Homography-based ground detection for a mobile robot platform using a single camera. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2006)
49. Zhou, J., Li, B.: Robust ground plane detection with normalized homography in monocular sequences from a robot platform. In: *IEEE International Conference on Image Processing (ICIP)* (2006)
50. Zhou, M., Ma, J., Ming, A., Zhou, Y.: Objectness-aware tracking via double-layer model. In: *IEEE International Conference on Image Processing (ICIP)* (2018)
51. Zhou, Y., Bai, X., Liu, W., Latecki, L.: Similarity fusion for visual tracking. *International Journal of Computer Vision (IJCV)* **118**(3), 337–363 (2016)