Monocular Based Navigation System for Autonomous Ground Robots Using Multiple Deep Learning Models

Machkour, Zakariae; Ortiz Arroyo, Daniel; Durdevic, Petar

Published in:
International Journal of Computational Intelligence Systems

DOI (link to publication from Publisher):
10.1007/s44196-023-00250-5

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (APA):
Machkour, Z., Ortiz Arroyo, D., & Durdevic, P. (2023). Monocular Based Navigation System for Autonomous Ground Robots Using Multiple Deep Learning Models. *International Journal of Computational Intelligence Systems*, *16*, 79-97. Article 79. https://doi.org/10.1007/s44196-023-00250-5

International Journal of Computational Intelligence Systems (2023) 16:79
https://doi.org/10.1007/s44196-023-00250-5

RESEARCH ARTICLE

Monocular Based Navigation System for Autonomous Ground Robots Using Multiple Deep Learning Models

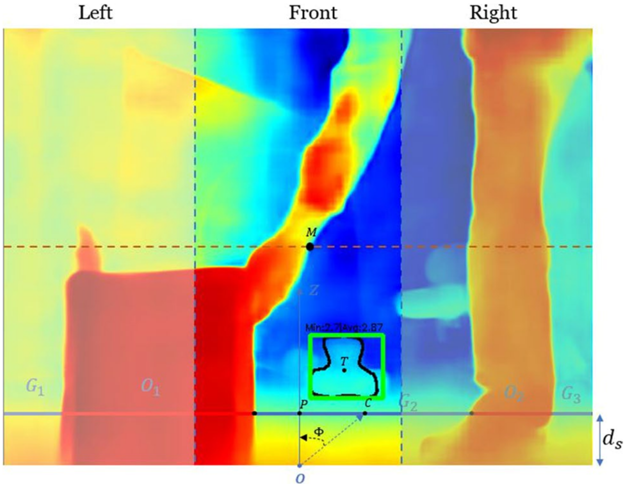
Zakariae Machkour[1] · Daniel Ortiz-Arroyo[1] · Petar Durdevic[1]

Received: 20 April 2022 / Accepted: 12 April 2023

Abstract

In recent years, the development of ground robots with human-like perception capabilities has led to the use of multiple sensors, including cameras, lidars, and radars, along with deep learning techniques for detecting and recognizing objects and estimating distances. This paper proposes a computer vision-based navigation system that integrates object detection, segmentation, and monocular depth estimation using deep neural networks to identify predefined target objects and navigate towards them with a single monocular camera as a sensor. Our experiments include different sensitivity analyses to evaluate the impact of monocular cues on distance estimation. We show that this system can provide a ground robot with the perception capabilities needed for autonomous navigation in unknown indoor environments without the need for prior mapping or external positioning systems. This technique provides an efficient and cost-effective means of navigation, overcoming the limitations of other navigation techniques such as GPS-based and SLAM-based navigation.

Graphical Abstract



Keywords Monocular depth · Object detection · Image segmentation · Sensitivity analysis · Obstacle avoidance · Mapless navigation

✉ Zakariae Machkour
zma@energy.aau.dk

Daniel Ortiz-Arroyo
doa@et.aau.dk

Petar Durdevic
pdl@et.aau.dk

[1] AAU Energy Department, Aalborg University, Niels Bohrs Vej 8, 6700 Esbjerg, Denmark

Abbreviations

BTS	From big To small
CRF	Conditional random field
ITC	Individual tree crown
LPG	Local planar guidance
mAP	Mean average precision
MDE	Monocular depth estimation

Published online: 10 May 2023

MSE	Mean square error
ODIS	Object detection and image segmentation
RMSE	Root mean square error
ROI	Region of interest

1 Introduction

Robotic navigation is a complex task that involves accurate perception of the robot's surroundings, trajectory planning, and actuator control. Navigation systems can be categorized into three main groups: map-based, map-building, and map-less systems [1, 2]. Map-based navigation systems rely on geometric models or topological maps of the environment to navigate [3], while map-building systems can build maps of the environment during runtime and localize the robot relative to the created map [4, 5]. However, visual SLAM technology is often unstable and prone to error, particularly in complex and dynamic environments that frequently change over time [6, 7]. Although deep learning techniques can enhance image recognition accuracy in complex scenes when combined with vSLAM, their performance in runtime is often poor, which limits their practical applications [7].

In contrast, map-less systems, which operate without a map of the environment, generate motion commands for robot navigation based on sensor data. This approach eliminates the cost of storing and maintaining a map, resulting in low computational complexity and fast response times. Additionally, map-less systems are useful in situations where it is impractical to build and maintain a map, such as when the robot operates for a short period or must navigate an unknown, dynamic, and complex environment [8]. Depth estimation, object detection [9], and a control system are typically used to enable the robot to detect obstacles and their distance during navigation.

Various proximity sensors, such as LiDAR [10], RADAR [11], or ultrasonic [12], have been used to measure depth data and enhance perception capabilities in robotic systems. Nonetheless, the employment of depth estimation techniques utilizing cameras, such as Mono or Stereo, has emerged as a cost-effective alternative for depth sensing [13].

Although stereo cameras can provide the required depth information [14], they are frequently orders of magnitude more expensive than a monocular camera [15].

With the fast advancement of deep neural networks, monocular depth estimation (MDE) based on deep learning has received increased attention in recent years. Surveys on the topic such as [16–18] describe a diversity of methods to train deep neural networks that are capable of estimating depth in a 2D image.

MDE problem is handled in a supervised way [19] with depth information obtained from labeled data or in an unsupervised fashion [20] using unlabeled stereo images or video datasets. Examples of training and benchmarking datasets used in depth estimation are NYUv2 [21] for indoors, and KITTI [22] for outdoors scenes.

Unlike stereo vision methods, MDE algorithms can't use epipolar geometry, but instead, they rely primarily on pictorial cues [23] that indicate depth within a single image, such as perspective, occlusion, object size, or texture gradients.

Eigen et al. [19] demonstrated one of the earliest examples of MDE by utilizing a CNN to predict depth maps directly from images, thereby inspiring numerous DNN-based solutions in this field. Xu et al. [24] introduced an attention-based CRF to regulate feature fusion across scales, while DORN [25] discretized depth and used ordinal regression. DenseDepth [26] utilized a DenseNet-169 that was pre-trained on ImageNet as an encoder in conjunction with a simple decoder that did not include batch normalization to enhance the quality and resolution of depth maps. Using DenseNet [27] and ResNext-101 [28] as encoders and Dense ASPP as a dense multi-scale feature learning module, Lee et al. [29] developed BTS. During the decoder stage, they used LPG layers instead of upsampling skip connections. BTS achieved a high level of accuracy (88.5% on NYUv2).

These state-of-the-art MDE methods prioritize accuracy, but their large number of parameters make them impractical for resource-constrained platforms. FastDepth [30] addresses this issue by introducing a low latency depth estimation network for embedded systems, but with reduced accuracy (77.1% on NYUv2).

MDE can be combined with object detection and image segmentation to allow robots to interact more with their environment. Tesla's Autopilot network [31] is a notable example of such a combination, utilizing 48 neural networks to perform semantic segmentation, object detection, and monocular depth estimation that take 70,000 GPU hours to train. One-stage approaches like Yolo [32] and SSD [33] have high inference speed, while two-stage approaches like Fast-RCNN [34], Faster-RCNN [35], and MaskR-CNN [36] have high accuracy [37]. Mask-RCNN has been used successfully in various applications, such as identifying damage in building structures with 95.13% accuracy [38] and outlining individual tree crowns in a tropical forest with up to 98% precision [39, 40].

The aim of our research is to develop a monocular vision-based approach that can estimate the distance to a target object and enable navigation towards it through the joint use of monocular depth estimation, object detection, and image segmentation. Our method involved utilizing Mask-RCNN for object detection and image segmentation and BTS as an MDE network which has achieved high accuracy on the indoor NYUv2 dataset. We conducted several experiments in our university lab to evaluate the accuracy of distance estimation under various conditions, such as occlusion, grayscale, blur, and high brightness.

Our study proposes a novel approach for robot navigation that utilizes monocular depth estimation (MDE) and takes into consideration the relative positions of objects in the scene, rather than relying solely on accurate depth estimates. The sensitivity analysis conducted on BTS indicates that the pre-trained model's accuracy is unsatisfactory when applied to indoor datasets other than NYU. To make scene analysis more intuitive, we partition the depth map into regions. To navigate autonomously through obstacles in the scene, we integrate our navigation algorithm with a PID controller.

The remainder of this paper is organized as follows. Section 2 presents a summary of the related works and our main contributions. Distance estimation and sensitivity analysis, are described in Sect. 3. The proposed mapless based navigation system is introduced in Sect. 4. Finally, we discuss the results and explore future directions in Sects. 5 and 6.

2 Related Works

In the following sections, we delve into recent studies that investigate the use of depth information to enhance object detection and image segmentation, in addition to exploring the potential advantages of cross-task features that can augment all three tasks. Furthermore, we analyze cutting-edge examples of mapless navigation systems.

2.1 Integration of Depth Estimation, Object Detection, and Image Segmentation: Recent Studies

Combining depth estimation with object detection and image segmentation in one sole application has been studied in the recent literature.

For instance, the authors in [41] propose a deep framework which jointly detects vehicles and estimates the distance. The proposed multi-tasks framework consists of a depth estimation module, a multi-modal Siamese network for object detection, and a depth-guided distance estimator. The Siamese network leverages the geometric features provided by the depth information, in addition to RGB image, to improve the detection performance. The depth ROI (region of interest) features are combined with the bounding box coordinates and fed to a Distance Regression Neural Network to estimate the distance.

Using Multiplixed images, the work in [42] can perform simultaneously positions and depth estimation of the detected objects. The proposed approach uses a disparity detector network which is based on SSD, and the shape of the anchor is defined by a k-mean algorithm proposed by YOLO [24]. The problem with this approach is that it is based on multiplixed images which requires a special image device and not just an ordinary camera.

Another example reported in [13], where a distance estimation system is based on Multi Hidden-Layer Neural Network, named DisNet, is used to learn and predict the distance between the object and the camera sensor. The DisNet was trained using a supervised learning technique where the input features were manually calculated parameters of the object bounding boxes resulted from the YOLOv3 [43] object classifier and outputs were the 3D laser scanner measurements of the distances to objects in the recorded scene.

Further, a recent method predicts the Time-to-Collision from a monocular video camera embedded in a smartglasses device [44]. It consists of a static data extractor module using a convolutional neural network to predict the obstacle position and distance from the obstacle, and a dynamic data extractor module that stacks the obstacle data from multiple frames and predicts the Time-to-Collision with a fully connected neural network. YOLOv3 and FastDepth [30] networks were used for object detection and depth estimation, respectively. By applying the bounding box on the depth map, the minimum depth value within this 2D area is considered as the obstacle distance. However, using the minimum depth value within the bounding box would not be correct in case the obstacle is partly occluded by another object. In this situation, the minimum depth value would correspond to the foreground object (nearest) and not the obstacle (in the background), as shown in Fig. 1. Our proposed method solves this problem by using image segmentation instead of object detection.

2.2 Methods Leveraging Depth Information for Object Detection and Image Segmentation

Other methods [45–47] employed rich depth information to support the object detection and image segmentation tasks.

Cao et al. [45], used the estimated depth information as a cue to improve the performance of both object detection and segmentation. For depth estimation, the authors used a DCNF (Deep Convolution Neural Field) model [48]. The network was trained on RGB-D dataset and it consists of three parts: (i) a unary part that takes an input RGB image after being segmented into n patches called superpixels, and outputs a corresponding n-dimensional vector of regressed depth values. (ii) The pairwise part takes similarity vectors for each pair of neighboring superpixels and feeds them to a fully-connected layer. (iii) The contentious CRF (Conditional Random Field) [49] loss layer takes both outputs from unary and pairwise parts and minimize the negative log-likelihood. Then, the estimated depth map is used to improve the object detection and segmentation. For object detection, two models R-CNN [50] and Fast R-CNN [34] were used to learn depth features for each object proposal. In R-CNN based object detection model, the RGB features and depth features are concatenated and

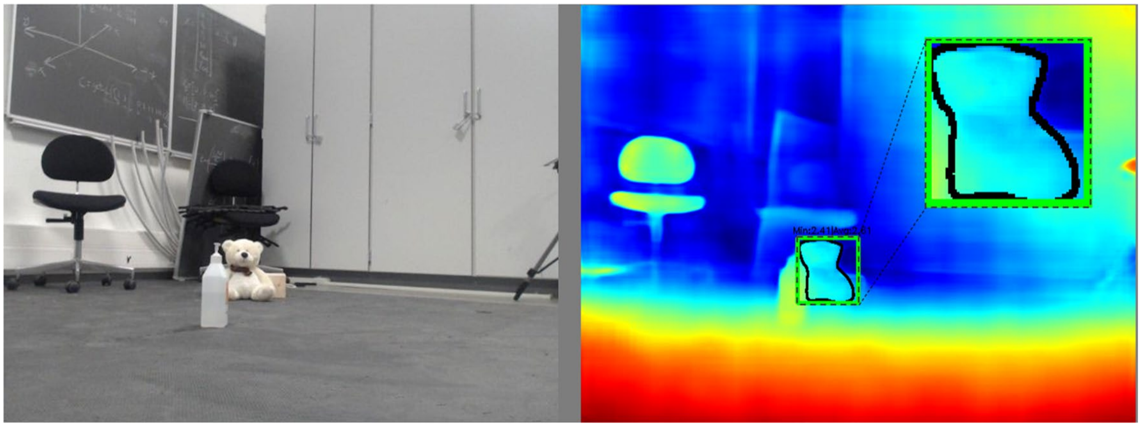


Fig. 1 Example of overlapping scenario where the target object is partially occluded

used to train a binary SVM for each class as object detector. Instead of extracting RGB and depth features for each object proposal (which is considered as resource consuming), RGB-D Fast R-CNN object detector takes the entire RGB and depth images and generates convolutional spatial maps using fully CNN. Different datasets (NYUv2, VOC 2007,) were used to test the RGB-D detectors, and the results showed an improvement in mAP (mean average precision) when both RGB and depth features were combined. For RGB-D semantic segmentation, two models are build one with feature concatenation and the other one using multi-task joint training. In RGB-D segmentation with feature concatenation, the idea is similar to RGB-D object detection where the RGB and depth features are concatenated and only one loss function is used during the training. Instead, RGB-D segmentation by multi-task joint training uses two loss functions and two separate networks for RGB and depth features extraction. VOC2012 dataset was used to test the RGB-D segmentation methods, and all results (in terms of maximum mean IoU) with depth features outperform the RGB only results.

The authors in [46] fuse instance segmentation and 3D object detection using depth information. The proposed framework discretizes the objects' depth into depth categories and feeds them simultaneously to a 3D branch and a mask branch to predict instance-level and pixel-level depth categories, respectively. The proposed method was 18 times faster than Mask-RCNN and 1.8 times faster than a two-stage instance segmentation method LklNet [51].

Kang et al. [47] have presented a depth adaptive deep neural network that adjusts the receptive field for semantic segmentation. The proposed network is based on a depth-adaptive multiscale convolutional layer including an adaptive perception neuron and the in-layer multiscale neuron where the distance of each point from the camera is used to determine the size of each filter of each neuron.

2.3 Synergies Between Depth Estimation, Object Detection, and Segmentation Tasks

Other studies have shown that depth estimation, object detection, and segmentation tasks might also share cross-task features that could benefit each other.

In their work, Lin et al. [52] developed a hybrid CNN (HybridNet) that integrates semantic segmentation and depth estimation. The proposed approach was evaluated for two different architectures; one unifies the two tasks by sharing the feature extraction network for both tasks. The commonly extracted features are used separately for each task. This method makes the training easy since most parameters are shared for both tasks. The other architecture considers a feature-extraction network for both tasks and a global depth network used only for the depth estimation. The extracted features are concatenated with the Global Depth Net to benefit from object features, and rich depth boundaries learned in the semantic segmentation and depth estimation tasks.

Wang et al. [53] propose two models built from RefineDet [54] and MonoDepth [55], to obtain both object detection and depth estimation separately and then integrate the 2D location of the detected object with the depth information to achieve real-time detection and depth estimation.

Further, Atapour-Abarghouei et al. [56] developed a monocular depth estimation approach that uses pixel-level semantic information to estimate the depth of certain segmented objects within a scene and integrates the generated depth outputs to have the overall scene's depth. The proposed method decomposes the input scene into four main object groups using four segmentation networks. Each segmented object group is passed as input to a depth generator, which outputs a depth map corresponding to the object group, and the four generated depth maps are fused to produce the final depth. The entire model was trained end-to-end using a synthetic dataset [57] containing both ground truth pixel-wise segmentation labels and depth.

Finally, the authors in [15] use an ensemble network architecture for combining the learned features from N-many U-Nets into a single pixel-by-pixel output. The network's primary task, depth estimation, is enhanced by integrating component U-Nets pre-trained on supplementary tasks, i.e., semantic segmentation. Two variants of NU-Nets were compared: simple and connected. Simple W-Net consists of two U-Nets in series, where the first U-Net is a pre-trained network for segmentation. The second U-Net is an untrained network for depth estimation, which takes the output from the first U-Net. During the training of the depth estimation U-Net, the learned features for image segmentation are preserved by freezing the weights of the first U-Net. The second variation of the Serial U-Nets, W-Net connected, connects the output of the first U-Net concatenated with the original RGB image as input to the second U-Net for depth estimation. W-Net and W-Net Connected both trained and validated on the NYUv2 dataset. The two Serial U-Nets were benchmarked against the existing U-Net architecture using MSE (Mean Square Error) and RMSE (Root Mean Square Error) metrics. The results show that W-Net Connected presents the best performance in terms of MSE, while U-Net performs 75% better than W-Net and W-Net Connected on average runtime.

Although these works mentioned above succeeded in combining depth with other computer vision tasks, they still lack a demonstration in a real application. Our work demonstrates the integration of monocular depth with object detection and semantic segmentation to identify a target object and its estimated distance from a ground mobile robot for navigation. Since there are very few works on what depth networks have learned and what depth cues they exploit [23, 58], we investigate the mechanisms of MDE by choosing different scenarios, such as occlusion, to evaluate their impact on the estimated distance. For example, the authors in [23] investigate what visual cues the network exploits for depth estimation and expects that both the vertical image position and apparent size of objects are the candidate cues by modifying or disturbing the input images (i.e., adding conflicting visual cues) and then looking for a correlation in the generated depth maps. The experiments show that the MonoDepth network [55] is mainly based on the vertical image position cue, with the assumption that objects rest on the ground. Moreover, changes in camera position (pitch and roll) disturb the estimated distance towards obstacles even if MonoDepth can detect these changes. Regarding obstacle recognition, the experiment results suggest that MonoDepth relies on a set of features applicable to cars, not to other objects inserted into the test images. Changes in color and texture of the obstacles where the value information in the image remains unchanged have marked only a slight degradation in performance, which indicates that the exact color of obstacles does not strongly impact the depth estimate. Lastly, the experiments show that the MonoDepth network can estimate the depth based on the object's bottom and side edges. The presence of shadows underneath objects can provide a high contrast bottom edge, which improves the detection of arbitrary objects MonoDepth could not detect before.

Unlike [23], Hu et al. [58] proposed a visualization approach that investigates which parts of the input image have a more significant impact on the estimated depth. The authors assume that CNNs can accurately infer a depth map from a small number of image pixels and formulate this as a sparse optimization problem. Their approach uses two separated CNNs to estimate an image mask that determines the smallest number of pixels. The target CNN can generate the maximally similar depth map to the estimated depth from the original input. The second CNN is the target CNN for the visualization. The results suggest that CNNs can accurately estimate a depth map from only a sparse subset of the input image specified by the predicted saliency map. The experimentation of the predicted masks has been done for different input images for various depth estimation networks, ResNet-50-based model of [59] and three models of [60] whose backbones are ResNet-50, DenseNet-161, and SENet-154, respectively.

2.4 Mapless Robotic Navigation

An example of a map-less navigation method was proposed in [61] that navigated to a target location by using landmark images. The system used the name and image of landmarks as input to a CNN which outputs the bounding box. The difference between the centers of the bounding box and the input image is used in a Visual Servoing (VS) system to determine the direction towards the target location. A second DNN takes the target location's initial and desired images and outputs the moving amount to the target position. However, the proposed VS method presents several issues:

1. The navigation system does not consider the dynamic environment (moving and static obstacles), whether cluttered or sparse.
2. The system learns the relationship between the initial and reference image, which means that a new network has to be trained for each reference pose, making it impractical for new environments.
3. The system is limited to the presence of landmarks.

In another recent example [62], the authors use a mapless LiDAR navigation control method for a wheeled mobile robot. The proposed method employs a deep CNN network that learns the mapping between the input LIDAR data and the desired motion behavior through end-to-end imitation learning.

Similarly, the authors in [7] employ LiDAR data as visual features in a VS system for positioning tasks. The proposed VS system uses the pose relationship between the desired and the current point cloud to move the robot from an initial to the desired position.

Further, Nguyen et al. [63] propose a deep learning solution NMFNet for the autonomous navigation of a mobile wheeled robot. The authors use an RGB-D camera to capture the RGB images and points cloud, while the distance map is built from a Lidar device. NMFNet consists of three modules for learning the depth features from the distance map, extracting the features from RGB images, and handling 3D point cloud data. The output of these three modules is fused to predict the steering angle.

Another VS scheme that uses RGB-D data to navigate towards the target location is presented [64]. The proposed method drives the robot by minimizing the depth map error between the initial and target position.

Moreover, other several recent research papers survey the current works in vision-based robot navigation, such as [65–68].

Besides the issues and shortcomings that we mentioned above, none of these surveys and the other previous examples discuss or use a DNN-based depth map solution in their VS navigation system, specifically monocular depth estimation MDE.

2.5 Contributions

The main contribution of this work is two-fold. Firstly, we estimate distance to target objects by combining depth estimation, object detection, and image segmentation generated by deep neural networks (DNNs), allowing us to provide more precise distance estimates, particularly for partially occluded objects, in comparison to using object detection alone. Secondly, we investigate the impact of different monocular cues on monocular depth estimation using the BTS model. This analysis is conducted through a series of sensitivity experiments that explore the effects of various factors on the estimated distance, and highlight the significant influence of camera height on monocular depth estimation (MDE) accuracy.

Furthermore, we present a map-less obstacle avoidance algorithm that leverages the monocular depth map obtained from the DNN. We integrate it with an object detection network to achieve object tracking and obstacle avoidance, where the depth map provides crucial information for obstacle detection. Our approach is distinct from previous works because it solely relies on a pre-trained MDE model for obstacle avoidance, and segments the depth map into regions to enhance its usefulness for obstacle detection. We also utilize a PID controller for robot navigation, which takes the estimated signal as input from the MDE network acting as a sensor.

3 Distance Estimation and Sensitivity Analysis

In this section, we demonstrate the integration of monocular depth with object detection and image segmentation to identify a target object and estimate its distance from a ground mobile robot. We also perform a sensitivity analysis using a statistical and graphical method based on the variance of the estimated distance error.

3.1 Distance Estimation

Our proposed vision-based system (described in Fig. 2) takes a single image as input. It estimates the distance to a target object (a teddy bear) from the robot's camera by combining the outputs from the pretrained MDE network (BTS) and the image segmentation network (Mask-RCNN). The obtained 2D mask from Mask-RCNN is overlayed onto the depth map to select only the region corresponding to

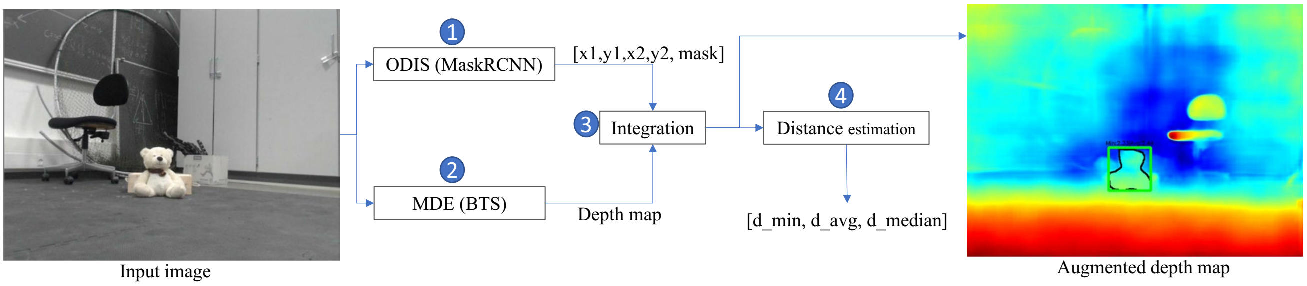


Fig. 2 System overview. (1) Object Detection and Image Segmentation (ODIS) Mask-RCNN module generates bounding box and 2D mask. (2) BTS network outputs depth map. (3) Integration of outpus from (1) and (2) and the generation of Augmented depth map. (4) Distance estimation module provides the 3 distance measurements

the target object and calculate the distance described in the following pipeline.

The pipeline of the distance estimation system can be summarized in the following steps:

1. Read the input image.
2. Run Mask CNN to detect object and return bounding box [x1, x2, y1, y2] and 2D mask.
3. Generate the depth map using BTS network.
4. Overlay the obtained 2D mask onto the generated depth map to select only the region corresponding to the target object, resulting into an augmented depth map.
5. Calculate the minimum, the median, and the average distances from the selected filtered region.

The pipeline results into 3 measurements of the distance:

- **d_min**: The minimum distance inside the filtered region of the depth map.
- **d_avg**: The average distance inside the filtered region of the depth map.
- **d_median**: The median distance inside the filtered region of the depth map.

The pipeline above is repeated for every image taken by the robot's camera while navigating towards the target object. We compare these three distances **d_min, d_avg,** and **d_median** with the true distance **d_true**. **d_true** is calculated based on the Optitrack system, which can measure the position and orientation of the objects in a physical space [69]. We use the 2D coordinates of the robot and the target object from these measurements to calculate the Euclidean distance as **d_true**.

3.2 Sensitivity Analysis of the Estimated Distance

The captured images with monocular cameras can be subject to different variations, such as an improper lens alignment resulting in blurred vision or external light conditions increasing the brightness of the image pixels. In this work, we imitate these variations in the image to study the behavior of the MDE network BTS under these conditions. We conduct a sensitivity analysis in the form of a statistical and graphical method based on the variance of the estimated distance error.

We present 5 different scenarios (Figs. 3 and 4):

1. **Scenario1 (Normal)**: The robot navigates towards the target object without any obstacles. In this case, we use the RGB image taken by the robot's camera without any image modification to estimate the distances.
2. **Scenario2 (Grayscale)**: Similar to Scenario1, but instead of using the RGB image, we convert it to a grayscale image and use the grayscale image to estimate the distances.
3. **Scenario3 (Blur)**: Similar to Scenario1 but we apply a Gaussian blur function on the RGB image and use the blurred image to estimate the distances.
4. **Scenario4 (Brightness)**: Similar to Scenario1, we increase the RGB image's brightness and use the illuminated image to estimate the distances.
5. **Scenario5 (Occlusion)**: The target object is partially occluded while the robot navigates towards it. In this case, we use the RGB image taken by the robot's camera without any image modification to estimate the distances.

3.3 Depth Evaluation

Based on the graphical results presented in Fig. 4, it is evident that the BTS model's accuracy is consistently poor across all scenarios. We suspect that the height of the Turtlebot3 with the camera mounted at around 20 cm may significantly affect the accuracy of the depth model, which was initially trained using a specific camera height, as indicated in Table 5. In order to investigate this further, we conducted MDE accuracy measurements using different camera height configurations and encoders. Additionally, to evaluate the performance of the BTS model on a dataset with a low camera height, we utilized the TUM dataset [70] recorded by the Pioneer robot. The results of our experiments are presented in Tables 1, 2, 3, 4, and 5. The results of our experiments will be discussed in the results section.

3.4 Implementation

The algorithms developed in this work were tested in an indoor laboratory environment, with an area of 4.54 m^2. The overall system described in Fig. 2 runs on a remote workstation, and it was implemented using PyTorch library and Matlab.

For navigation, we use a Turtlebot3 robot equipped with a monocular camera Logitech 1080p. A Matlab program reads and saves images from the Turtlebot3 via ROS network. A python program executes the modules in Fig. 2 and generates the augmented depth map and the estimated distances.

In addition, for the classical depth evaluation, we used an Intel Realsense l515 camera to collect the ground truth depth.

In the navigation system, the code is executed remotely on a Linux workstation. The Turtlebot3 sends images to the workstation via ROS network. The python code runs on the remote workstation which is a Lenovo P53 equipped with a Quadro RTX 4000 GPU.

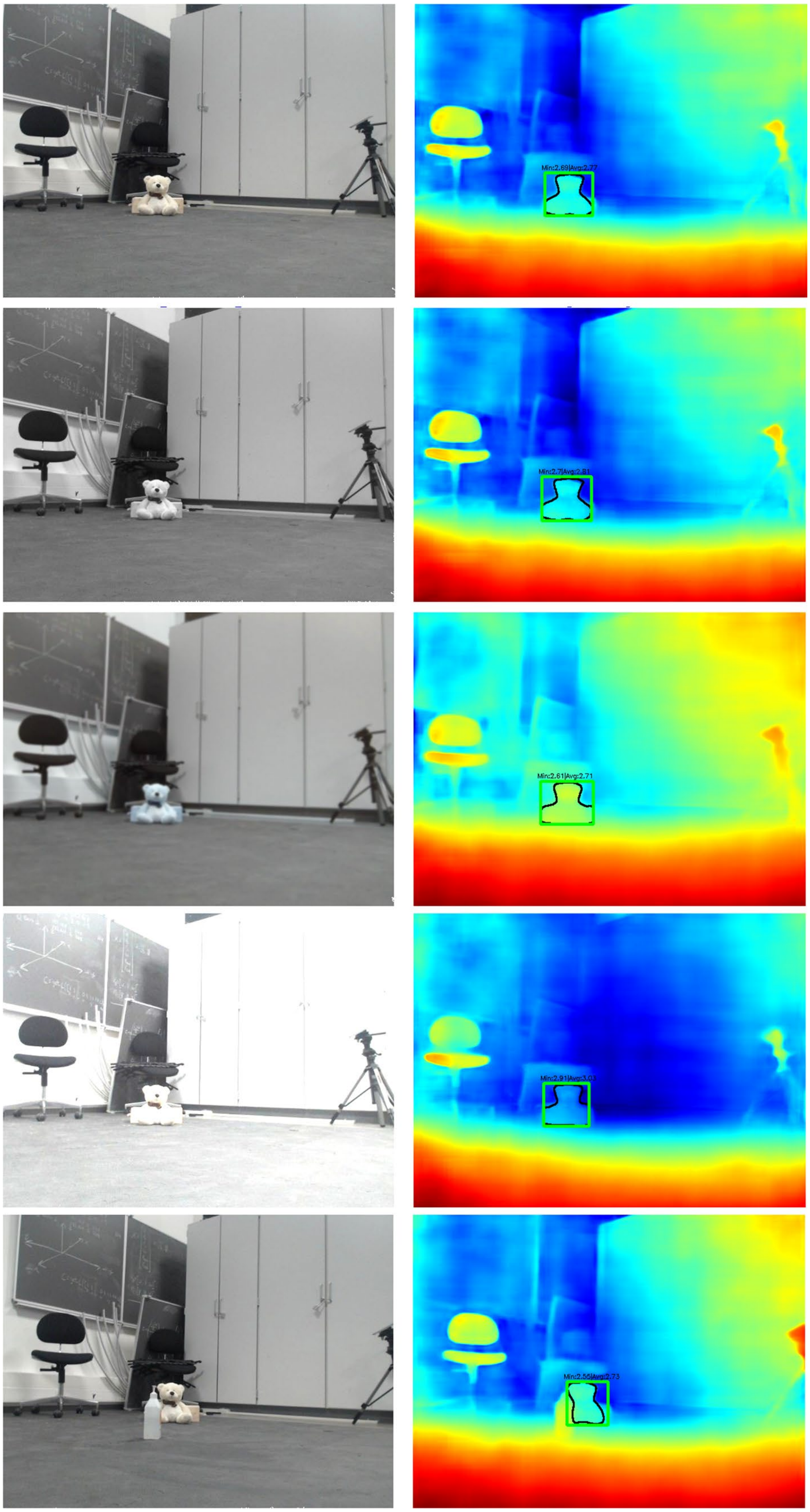


Fig. 3 Input images and corresponding depth images with detected target object, respectively, in different scenarios (Best viewed in color). From top to down: Normal, Grayscale, Blur, Brightness, Occlusion. From left to right: Input image, augmented depth map

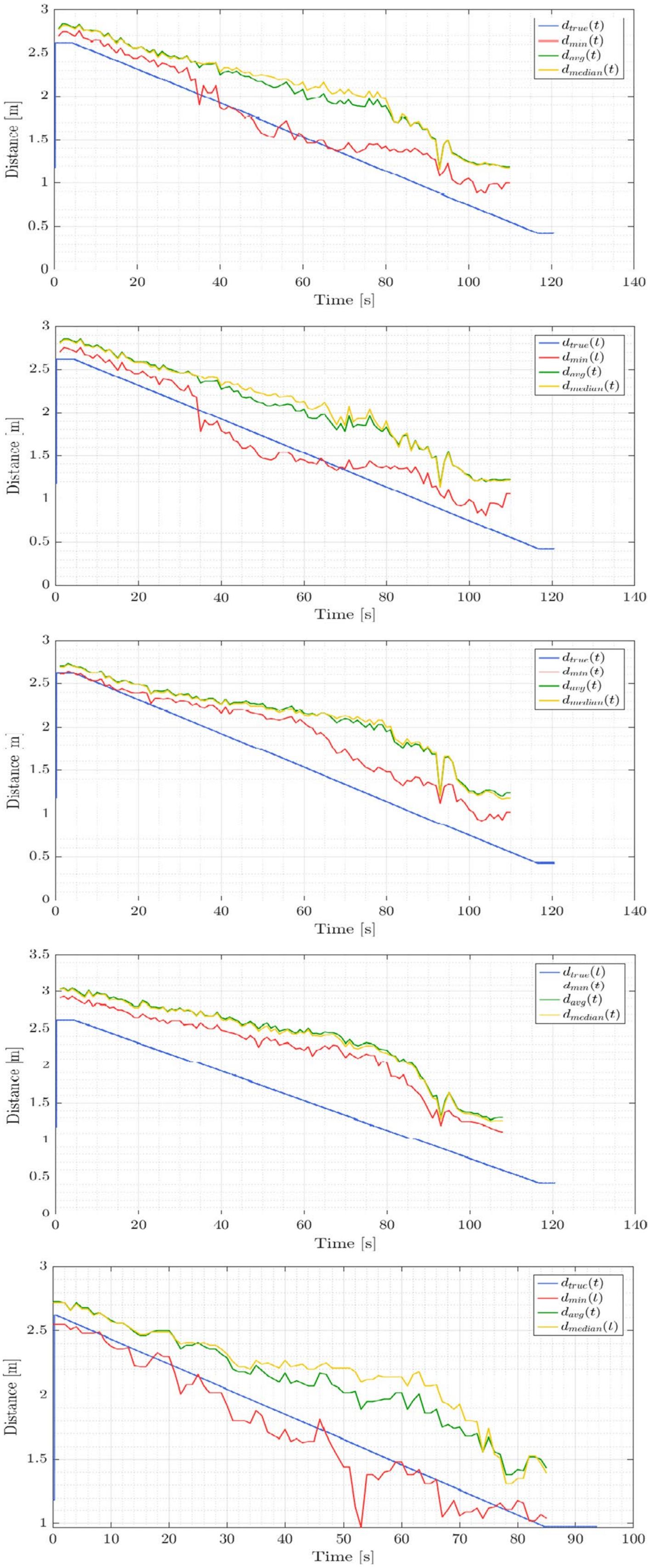


Fig. 4 Distance estimation in different scenarios. From top to down: Normal, Grayscale, Blur, Brightness, Occlusion. With **d_min** (red), **d_avg** (green), **d_median** (yellow), and with the true distance **d_true** (blue)

Table 1 The evaluation of BTS on the custom indoor dataset (Camera height ≈ 120 cm) using different encoders

Encoder	d1	d2	d3	AbsRel	SqRel	RMSE	RMSElog	SILog	log10
DenseNet161	0.728	0.943	0.991	0.166	0.086	0.412	0.213	16.347	0.076
DenseNet121	0.811	0.967	0.992	0.148	0.070	0.349	0.180	14.812	0.064
ResNet101	0.747	0.977	0.998	0.153	0.072	0.380	0.188	14.404	0.069
ResNet50	0.771	0.977	0.994	0.152	0.070	0.366	0.185	14.965	0.067
ResNeXt101	0.764	0.968	0.995	0.155	0.073	0.375	0.192	13.953	0.070
ResNeXt50	0.745	0.967	0.992	0.162	0.083	0.397	0.196	15.352	0.072
MobileNetV2	0.742	0.947	0.984	0.175	0.102	0.428	0.212	16.959	0.075

Table 2 The evaluation of BTS on the custom indoor dataset (Camera height ≈ 60 cm) using different encoders

Encoder	d1	d2	d3	AbsRel	SqRel	RMSE	RMSElog	SILog	log10
Densenet161	0.414	0.757	0.911	0.281	0.246	0.681	0.365	19.929	0.137
Densenet121	0.491	0.897	0.981	0.228	0.162	0.567	0.276	16.334	0.107
Resnet101	0.450	0.830	0.976	0.244	0.178	0.606	0.306	18.854	0.118
Resnet50	0.556	0.908	0.982	0.218	0.149	0.528	0.262	16.341	0.100
Resnext101	0.487	0.785	0.910	0.264	0.240	0.663	0.344	21.319	0.129
Resnext50	0.464	0.792	0.941	0.264	0.230	0.660	0.333	20.260	0.126
Mobilenetv2	0.507	0.857	0.965	0.248	0.199	0.600	0.298	19.640	0.111

Table 3 The evaluation of BTS on the custom indoor dataset (Camera height ≈ 20 cm) using different encoders

Encoder	d1	d2	d3	AbsRel	SqRel	RMSE	RMSElog	SILog	log10
Densenet161	0.337	0.483	0.594	0.850	1.120	1.043	0.671	42.783	0.231
Densenet121	0.322	0.491	0.615	0.788	0.958	0.974	0.637	39.185	0.220
Resnet101	0.288	0.434	0.598	0.869	1.132	1.104	0.669	37.629	0.241
Resnet50	0.308	0.436	0.595	0.854	1.098	1.069	0.666	38.482	0.236
Resnext101	0.319	0.489	0.649	0.790	0.954	0.982	0.633	38.687	0.221
Resnext50	0.324	0.494	0.626	0.809	1.015	0.995	0.649	41.563	0.223
Mobilenetv2	0.312	0.467	0.599	0.832	1.059	1.038	0.657	39.424	0.230

Table 4 The evaluation of BTS on the TUM dataset (Camera height ≈ 45 cm) using different encoders

Encoder	d1	d2	d3	AbsRel	SqRel	RMSE	RMSElog	SILog	log10
Densenet161	0.433	0.810	0.963	0.326	0.383	1.066	0.343	31.740	0.128
Densenet121	0.406	0.789	0.979	0.320	0.365	1.083	0.337	30.588	0.126
Resnet101	0.344	0.759	0.961	0.337	0.418	1.221	0.362	33.059	0.138
Resnet50	0.379	0.825	0.974	0.319	0.350	1.102	0.334	31.504	0.127
Resnext101	0.474	0.860	0.972	0.289	0.317	1.027	0.310	28.202	0.115
Resnext50	0.428	0.825	0.962	0.306	0.367	1.153	0.336	30.997	0.126
Mobilenetv2	0.404	0.807	0.966	0.320	0.384	1.144	0.341	31.363	0.129

Table 5 The evaluation of BTS on the NYU dataset (Camera height ≈ 130 cm) using different encoders

Encoder	d1	d2	d3	AbsRel	SqRel	RMSE	RMSElog	SILog	log10	Params
DenseNet161	0.885	0.978	0.994	0.110	0.066	0.392	0.142	11.533	0.047	47.0 M
DenseNet121	0.871	0.977	0.993	0.118	0.072	0.410	0.149	12.028	0.050	21.2 M
ResNet101	0.871	0.977	0.995	0.113	0.068	0.407	0.148	11.886	0.049	68.5 M
ResNet50	0.865	0.975	0.993	0.119	0.075	0.419	0.152	12.368	0.051	49.5 M
ResNext101	0.880	0.977	0.994	0.111	0.069	0.399	0.145	11.680	0.048	112.8 M
ResNext50	0.867	0.977	0.995	0.116	0.070	0.414	0.150	12.186	0.050	49.0 M
MobileNetV2	0.843	0.968	0.990	0.133	0.088	0.485	0.170	13.778	0.057	16.3 M

3.5 Results

Each of the 5 scenarios (Figs. 3 and 4) is repeated 10 times, with more than 100 images each. Figure 3 shows an example of each scenario where **d_min** (red), **d_avg** (green), **d_median** (yellow) are estimated and ploted with the true distance **d_true** (blue).

From Fig. 3, the **d_median** (yellow) distance is much less accurate compared to **d_min** (red) and **d_avg** (green) in all scenarios, so we considered discarding it from the subsequent analysis.

For each scenario, we calculate the root mean square error (RMSE) of each of **d_min** and **d_avg**, and we plot the box-plot to catch the variation of these measurements (Fig. 5). In all scenarios, the minimum distance **d_min** is more accurate than the average distance **d_avg**, therefore we will use **d_min**. Among all 5 scenarios, **d_min** has registered the highest accuracy in the case of Scenario5 (Occlusion), and the lowest accuracy in Scenario4 (Brightness).

From Fig. 3, we noticed that **d_min** becomes less accurate when the robot gets closer to the target object, around 1.5 ms or less. To visualize clearly this behavior, we calculate the RMSE of **d_min** when the distance of the robot from the target object is less than 1.5 m and when it is between 1.5 m and 2.5 m. The results are shown in Fig. 6.

Regardless the scenario and distance estimation metric employed (such as **d_min**), the accuracy of depth estimation is significantly low. The choice of camera height has a significant impact on depth estimation, as demonstrated by the classical depth evaluation metrics of the BTS model in different camera height configurations. For instance, Table 5 shows the performance of the BTS model on the NYU dataset, where the images and corresponding ground truth depth were captured at a standing hand height. Conversely,

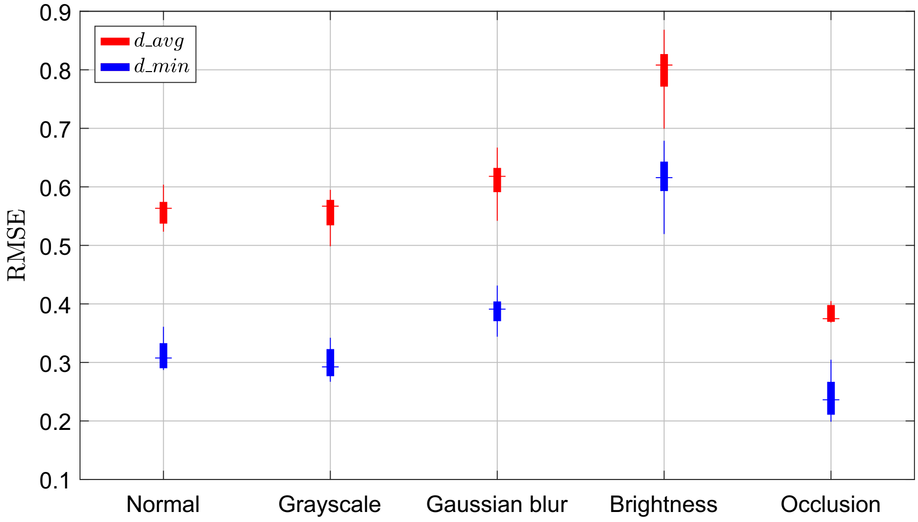


Fig. 5 RMSE of the estimated distances **d_min** and **d_avg** for all scenarios

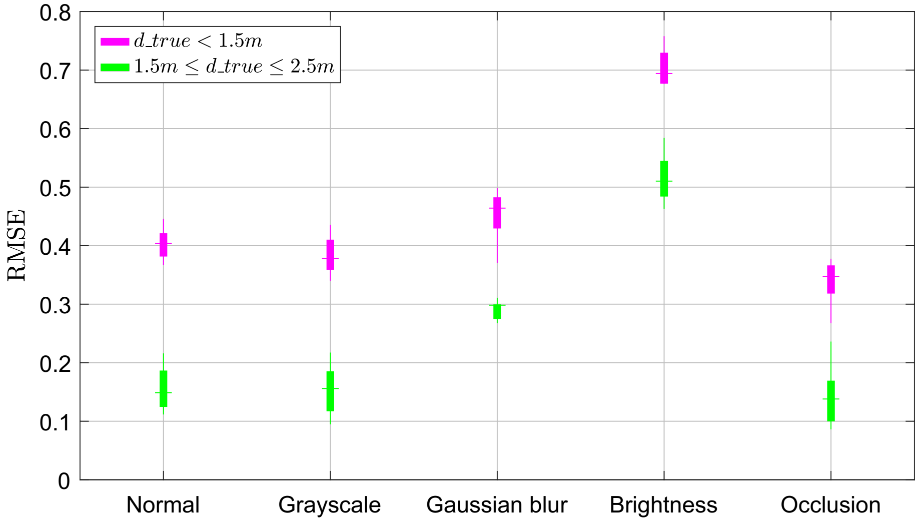


Fig. 6 RMSE of the estimated distance **d_min** for all scenarios where the distance *d_true* < 1.5 m and 1.5 m ≤ *d_true* ≤ 2.5 m

Tables 1, 2, and 3 present the performance of BTS on our indoor dataset with camera heights set to 120 cm, 60 cm, and 20 cm, respectively, with the latter being the height of the robot. The results show that the MDE accuracy is highest when the camera height is similar or closer to that of the NYU dataset, and the opposite holds for the lowest height. Additionally, Table 4 reports the performance of BTS on the TUM dataset captured by a Pioneer robot with a camera height of approximately 45 cm. The results indicate that the BTS accuracy is between the values obtained for camera heights of approximately 60 cm and 20 cm.

4 Mapless Monocular Depth Navigation

This section demonstrates a new algorithm that leverages the relative positions of objects in the scene to facilitate obstacle avoidance and navigation, rather than depending entirely on precise depth measurements. The algorithm's main objective is to identify gaps or free spaces in the environment that a ground robot can traverse safely.

4.1 Problem Formulation

We consider a point-mass robot in a closed bounded workspace $\mathscr{W} \subset \mathbb{R}^3$ which consists of $n \in \mathbb{N}$ obstacles $\mathscr{O}_i \subset \mathscr{W}$, $i \in 1, ..., n$. We denote $\mathscr{F}$ as the free space in $\mathscr{W}$ as $\mathscr{F} = \mathscr{W} \setminus \bigcup_{i=1}^{n} \mathscr{O}_i$.

We are using a two-wheel nonholonomic differential drive robot equipped with a monocular camera, to reach a target object in the environment without access to (or maintenance of) a map.

The robot must detect the free spaces or gaps within its camera's Field Of View (FOV), and it has to choose the appropriate gap through which can navigate without collision. We define L_g as the gap length between two obstacles, and L_r as the robot length (see Fig. 7). The navigation in the workspace $\mathscr{W}$ is plausible if:

$$\begin{cases} \exists \varphi \in [0, 2\pi], \exists \mathscr{O}_i, \mathscr{O}_j \subset \mathscr{W}, i, j \in 1, ..., n \mid n \in \mathbb{N}^*, i \neq j \\ \forall p, q \in \mathscr{O}_i \times \mathscr{O}_j, L_g > L_r, L_g = min\|p - q\| \end{cases} \tag{1}$$

with φ is the rotation angle of the robot around the z-axis.

4.2 Proposed Method

To navigate successfully towards a target object using our proposed method, the robot must be able to:

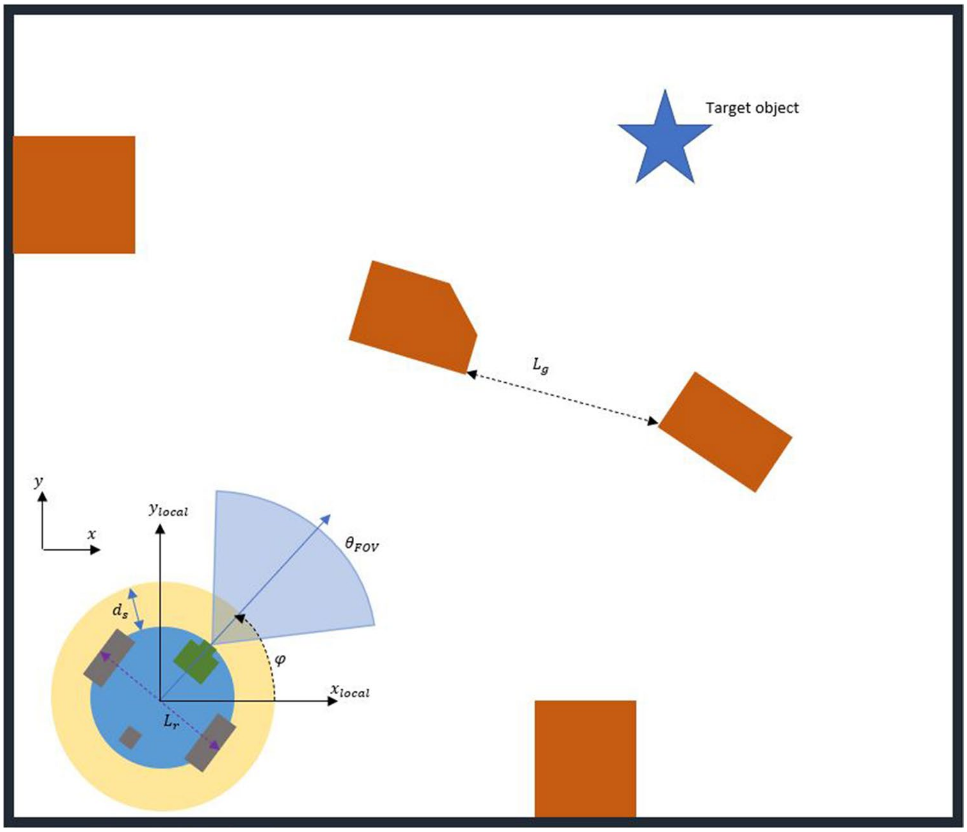


Fig. 7 Obstacle avoidance problem definition

1. Detect the obstacles inside a given range.
2. Detect the gaps between these obstacles.
3. Choose the appropriate gap and navigate through it.
4. Detect and navigate toward the target object.

We address the aforementioned problems using mainly a monocular camera as the robot sensor.

4.2.1 Obstacle Detection

Obstacle detection can not be fully done using only object detection models since this requires learning about all possible objects that could exist in an unknown scene. Instead, we are using a Monocular Depth Estimation (MDE) model [71], which helps in detecting obstacles even without using an object detection model.

However, as the depth map image used is estimated by a DNN rather than obtained from an active sensor, e.g., Lidar or Kinect, the resulting distance is inaccurate in practice, as seen previously in Table 3 and Fig. 4.

We propose to use the relative positions of objects in the scene to overcome the requirement of having a precise depth map.

First, we define a safety distance d_s (see Fig. 8) within which we need to keep out all obstacles while navigating. This safe space can also be considered as the ground space, which can be tagged in the depth map as a close space to the robot. The idea is to ignore everything in this space and consider only the obstacles outside it.

To detect the obstacles within the robot camera's FOV, we generate the monocular depth map and apply a threshold on it based on the distance color related to the accepted range. This will keep only the objects that are within a certain range and eliminate all other objects from the depth map image.

4.2.2 Free Space Detection

In this stage, we generate a binary map from the thresholded depth map (see Fig. 9) to detect the obstacles and the possible gaps between them. We assign 0 to free spaces and 1 to obstacles. In order to localize the gaps between the obstacles, we take the horizontal line defining the top edge of the safe space and we extract the corresponding array of pixel values (see Fig. 10).

Each group of 1 s denotes an obstacle $\mathcal{O}_i$, and each group of 0 s is a gap G_j with a length L_{G_j}.

4.2.3 Best Gap Selection

For plausible navigation, we need to find a free space G_j with $L_{G_j} > L_r$ (see Eqn. (1)). However, L_{G_j} is expressed in pixels and should be converted to real-world coordinates. The two extremities of the gap G_j are defined as (x_1, y_1) and (x_2, y_2), with $y_1 = y_2 = d_s$ (see Fig. 11).

By knowing the camera's intrinsic parameters and the real depth value, we can have this conversion as follows:

$$\begin{cases} X_1 = Z_1 \frac{x_1}{f} \\ Y_1 = Z_1 \frac{y_1}{f} \\ X_2 = Z_2 \frac{x_2}{f} \\ Y_2 = Z_2 \frac{y_2}{f} \end{cases} \tag{2}$$

with Z_1 and Z_2 are the real distances towards the point (x_1, y_1) and the point (x_2, y_2), respectively, and f is the camera's focal length.

The converted value of L_{Gj} becomes:

$$L_{Gj} = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \tag{3}$$

We define the set of candidate gaps as:

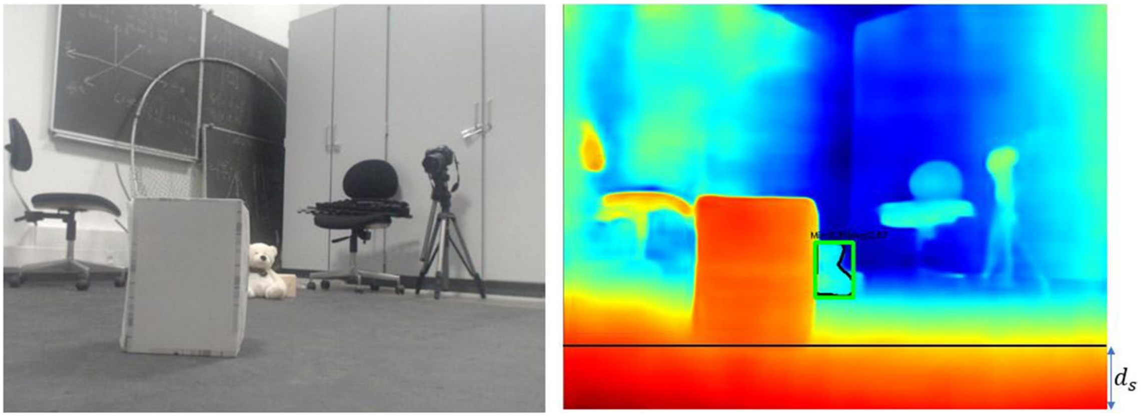


Fig. 8 Safety distance d_s for robotic navigation

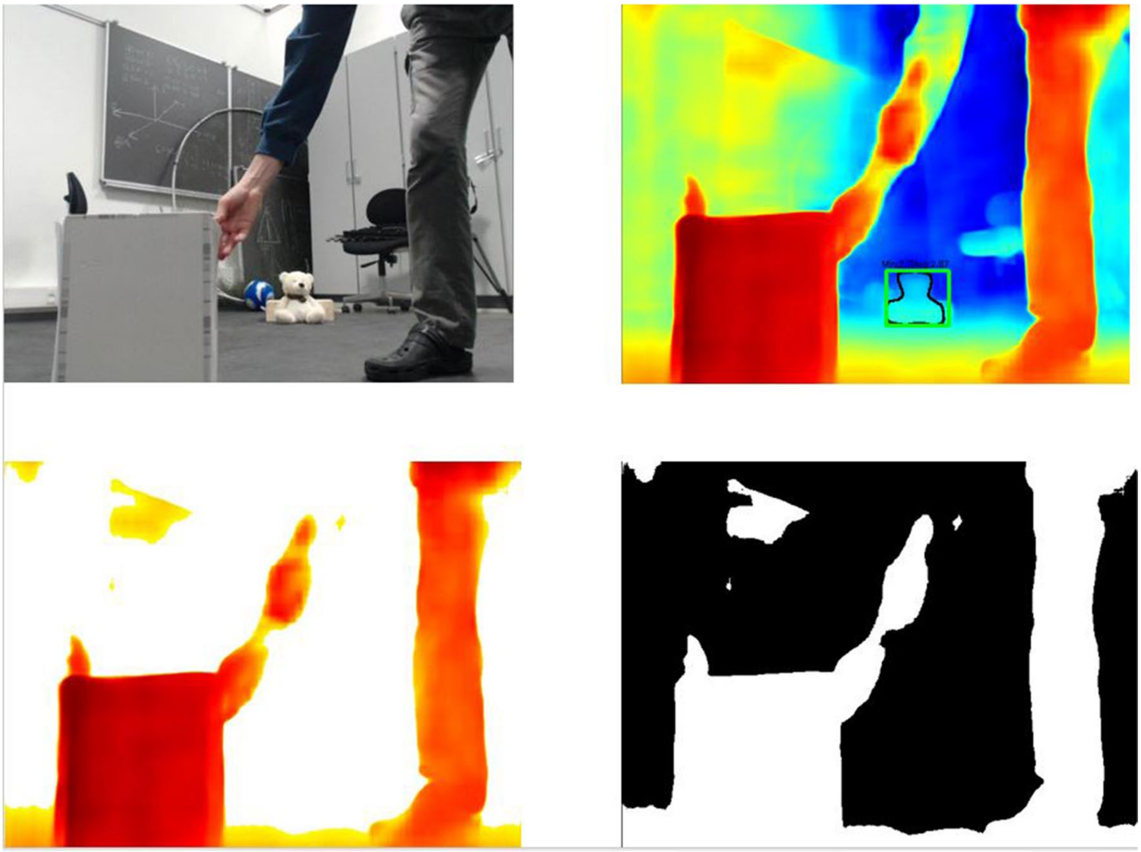


Fig. 9 The algorithm's main steps. (From top to bottom, from left to right): Original image, estimated depth map, thresholded depth map, binary thresholded depth map

$$S_G = \{G_j, j \in 1, \ldots, n \in \mathbb{N}^* \mid L_{G_j} > L_r\} \tag{4}$$

and among this set S_G, we choose the most appropriate gap $G_a \in S_G$, with C_a as the center of G_a and $\phi_a = \angle(\mathbf{OC_a}, \mathbf{OT})$ the bearing angle between OC_a and OT the center of the target object, such as:

$$\begin{cases} \forall \phi_i = \angle(\mathbf{OC_i}, \mathbf{OT}), \|\phi_a\| = \min_{G_i \in S_G} \|\phi_i\|, \\ \text{if target object within FOV.} \\ \\ L_{G_a} = \max_{G_i \in S_G} L_{G_i}, \text{otherwise.} \end{cases} \tag{5}$$

4.2.4 Navigate Towards the Target Object

After selecting the best free space to navigate through, we calculate the bearing angle between the current direction and the direction of the center of the gap, as (Fig. 12):

$$\phi = \arccos \frac{OC \times OP}{\|OC\| \times \|OP\|} \tag{6}$$

If the target object is within the FOV, we estimate the distance toward it using the integrated DNN models for MDE, object detection.

To control the translation and rotation velocities of the robot and its direction (front, right, or left), we use a PID controller that takes the bearing angle ϕ, and the difference between the target object center T and the image center P along the x axis (Fig. 12).

5 Discussion

We present the integration of MDE with object detection and image segmentation to identify a target object and its estimated distance from a ground mobile robot. Among different measurements of the distance, only the minimum distance **d_min** had the closest value to the true distance **d_true**. The accuracy of the estimated distance is proportional to the presence of monocular cues in the image. In MDE, the presence of occlusion in the scene improves the accuracy of the estimated depth [72], therefore the estimated object distance. However, occlusion can be considered an issue in some applications, such as estimating

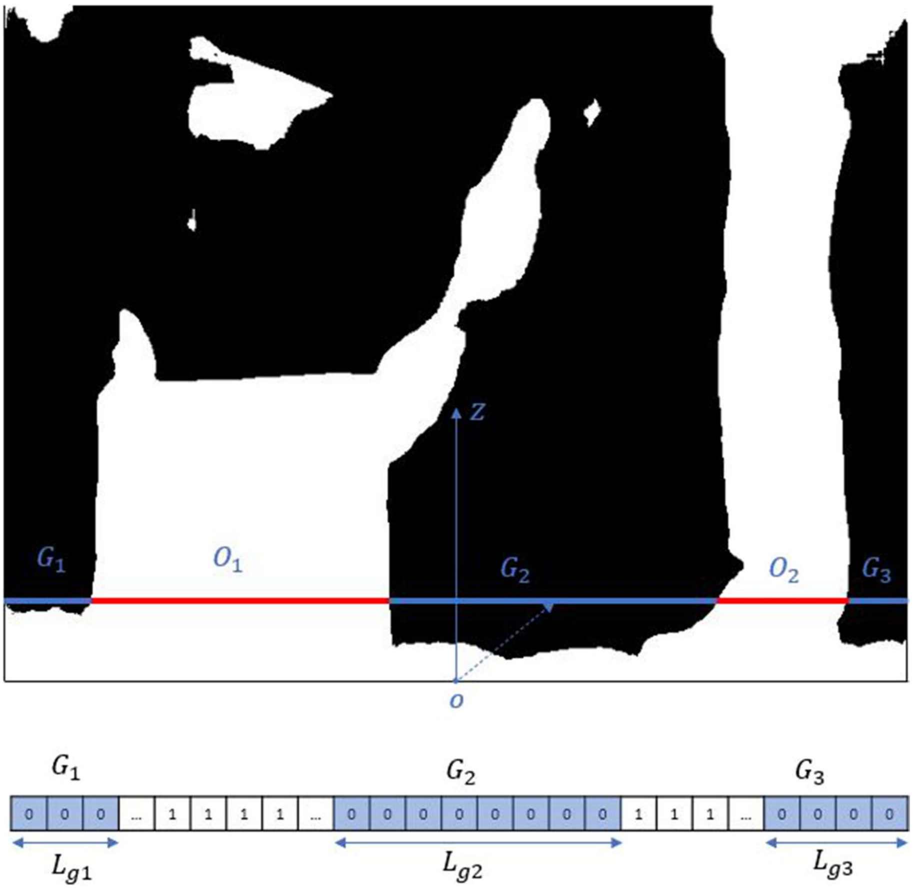


Fig. 10 Gaps and obstacles detection

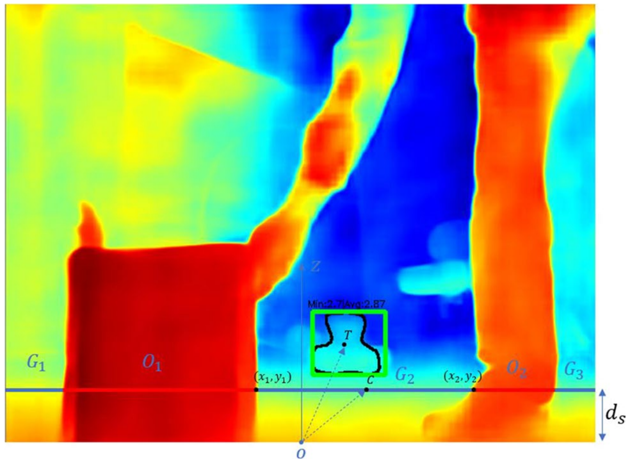


Fig. 11 Image coordinates of a gap

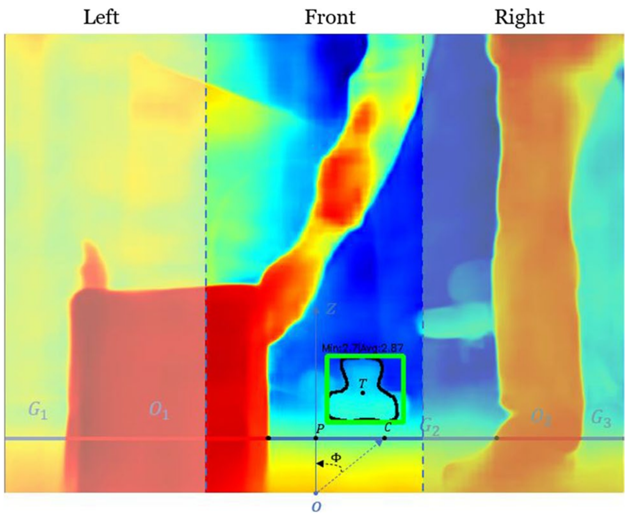


Fig. 12 Robot navigation towards target object

distance from a partially occluded person. The authors in [73] proposed a solution based on the pose estimation technique OpenPose [74], which can recognize the coordinates of an almost fully occluded person where only the head of the person is visible and use these coordinates in the distance estimation.

The estimated distance measured from grayscale images is close to the one calculated from RGB images, indicating that the color cue does not affect the MDE too much.

The results (Fig. 5) show that blurred and high brightness images give the lowest accuracy in distance estimation, wherein in these two scenarios, the edges and corners are partially deleted and distorted. This may indicate that edges and corners are also of great importance for MDE.

Combining RGB and grayscale images can also improve the performance of MDE in the case of poor light conditions, as reported by Jang et al. [75].

It is anticipated that the accuracy of MDE is directly influenced by the selection of camera height, irrespective of the scenario or encoder employed. For better accuracy, it is recommended to use a camera height during testing that is comparable to the height used during training. Alternatively, one could train the network on the desired camera height for the specific application, such as the height of the Turtlebot3 in our case. This could potentially result in improved accuracy during testing with the same camera height.

The results in Fig. 6 show that the estimated distance gets less accurate when the distance towards the target object is less than 1.5 m. One of the possible explanations for this could be the lack of linear perspective cue from the background scene due to the small and narrowed space in which the experiments took place. The linear perspective is a monocular cue that causes parallel lines to appear to intersect at a vanishing point in the distance, where closer objects to the vanishing point on the image plane are farther from the camera [76]. The training of the network for small ranges could solve this problem.

The proposed approach to monocular mapless navigation using deep learning techniques has shown promising results for unknown indoor environments. By utilizing the monocular depth estimation network and object detection network, a new algorithm was developed for obstacle avoidance and object tracking without relying on a pre-built map or simultaneous localization and mapping (SLAM) technologies.

One of the main advantages of this approach is its ability to work in real-time with a low-cost monocular camera, which is easily accessible and widely used in robotics. Additionally, the region-based segmentation of the depth map allows for a more precise and robust representation of the environment, reducing the problem's complexity by focusing on specific regions of interest.

In summary, this proposed approach demonstrates the potential of using deep learning techniques for monocular mapless navigation in unknown indoor environments.

6 Conclusions

We presented a monocular-based navigation system for an autonomous ground robot that can identify a target object and its distance from the camera in an unknown environment, using two pre-trained networks for "object detection - image segmentation" and monocular depth estimation. Our approach addresses the challenge of having partially occluded objects during distance estimation. On one hand, occlusion helps the depth estimator detect objects in front and behind. On the other hand, the area within the bounding box is reduced. We solved this problem by detecting and segmenting the target object and using only the segmented area instead of the entire bounding box to estimate the distance.

Our experiments showed that the estimated distance in the presence of occlusion cues has the highest accuracy among the other scenarios.

The experiments also showed, that color information does not strongly affect the accuracy of the estimated distance. Therefore, we suggest using gray scale images during training of the MDE network in our future work.

Importantly, the obstacle avoidance and object tracking capabilities of our navigation algorithm, which relies solely on a low-cost monocular camera, does not require a pre-built map or SLAM technologies, or a very precise depth map.

Acknowledgements Not applicable.

Author Contributions Conceptualization, ZM and DO-A and PD; methodology, ZM and DO-A and PD; software, ZM and DO-A and PD; validation, ZM and DO-A and PD; formal analysis, ZM and DO-A and PD; investigation, ZM and DO-A and PD; resources, ZM and DO-A and PD; data curation, ZM and DO-A and PD; writing-original draft preparation, ZM and DO-A and PD; writing-review and editing, ZM and DO-A and PD; visualization, ZM and DO-A and PD; project administration, ZM and DO-A and PD.

Funding This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Availability of data and materials The finding results are reported in the article. The code is available locally and it will be uploaded to a public repository once the paper has been conditionally accepted.

Declarations

Conflict of interest The author(s) declare(s) that there is no conflict of interest.

References

1. DeSouza, G.N., Kak, A.C.: Vision for mobile robot navigation: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **24**(2), 237–267 (2002)

2. Rubio, F., Valero, F., Llopis-Albert, C.: A review of mobile robots: concepts, methods, theoretical framework, and applications. Int. J. Adv. Robot. Syst. **16**(2), 1729881419839596 (2019)
3. Madhevan, B., Sreekumar, M.: Identification of probabilistic approaches and map-based navigation in motion planning for mobile robots. Sādhanā **43**(1), 1–18 (2018)
4. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: large-scale direct monocular slam. In: European Conference on Computer Vision, pp. 834–849. Springer (2014)
5. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE Trans. Robot. **31**(5), 1147–1163 (2015)
6. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al.: On evaluation of embodied navigation agents. Preprint at arXiv:1807.06757 (2018)
7. Zhang, S., Gong, Z., Tao, B., Ding, H.: A visual servoing method based on point cloud. In: 2020 IEEE International Conference on Real-time Computing and Robotics (RCAR), pp. 369–374. IEEE (2020)
8. de Villiers, F., Brink, W.: Learning fine-grained control for mapless navigation. In: 2020 International SAUPEC/RobMech/PRASA Conference, pp. 1–6. IEEE (2020)
9. Zhang, J., Sun, J., Wang, J., Li, Z., Chen, X.: An object tracking framework with recapture based on correlation filters and siamese networks. Comput. Electr. Eng. **98**, 107730 (2022)
10. Debeunne, C., Vivet, D.: A review of visual-lidar fusion based simultaneous localization and mapping. Sensors **20**(7), 2068 (2020)
11. Cen, S.H., Newman, P.: Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 6045–6052. IEEE (2018)
12. Rhee, J.H., Seo, J.: Low-cost curb detection and localization system using multiple ultrasonic sensors. Sensors **19**(6), 1389 (2019)
13. Haseeb, M.A., Guan, J., Ristic-Durrant, D., Gräser, A.: Disnet: a novel method for distance estimation from monocular camera. In: 10th Planning, Perception and Navigation for Intelligent Vehicles (PPNIV18), IROS (2018)
14. Durdevic, P., Ortiz-Arroyo, D.: A deep neural network sensor for visual servoing in 3d spaces. Sensors **20**(5), 1437 (2020)
15. Cantrell, K.J., Miller, C.D., Morato, C.: Practical depth estimation with image segmentation and serial u-nets. In: VEHITS, pp. 406–414 (2020)
16. Bhoi, A.: Monocular depth estimation: a survey. Preprint at arXiv:1901.09402 (2019)
17. Zhao, C., Sun, Q., Zhang, C., Tang, Y., Qian, F.: Monocular depth estimation based on deep learning: an overview. Sci. China Technol. Sci. **63**, 1612–1627 (2020)
18. Ming, Y., Meng, X., Fan, C., Yu, H.: Deep learning for monocular depth estimation: a review. Neurocomputing **438**, 14–33 (2021)
19. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. Adv. Neural Inf. Process. Syst. **27**, 2366–2374 (2014)
20. Godard, C., Mac Aodha, O., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3828–3838 (2019)
21. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: European Conference on Computer Vision, pp. 746–760. Springer (2012)
22. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the kitti dataset. Int. J. Robot. Res. **32**(11), 1231–1237 (2013)
23. Dijk, T.V., Croon, G.D.: How do neural networks see depth in single images? In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2183–2191 (2019)
24. Xu, D., Wang, W., Tang, H., Liu, H., Sebe, N., Ricci, E.: Structured attention guided convolutional neural fields for monocular depth estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3917–3925 (2018)
25. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2002–2011 (2018)
26. Alhashim, I., Wonka, P.: High quality monocular depth estimation via transfer learning. Preprint at arXiv:1812.11941 (2018)
27. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
28. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6546–6555 (2018)
29. Lee, J.H., Han, M.-K., Ko, D.W., Suh, I.H.: From big to small: multi-scale local planar guidance for monocular depth estimation. Preprint at arXiv:1907.10326 (2019)
30. Wofk, D., Ma, F., Yang, T.-J., Karaman, S., Sze, V.: Fastdepth: fast monocular depth estimation on embedded systems. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 6101–6108. IEEE (2019)
31. TeslaAI: Tesla AI,". https://www.tesla.com/AI (2022). Accessed 6 Dec 2021
32. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
33. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37. Springer (2016)
34. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
35. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. **28**, 91–99 (2015)
36. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2961–2969 (2017)
37. Lu, X., Li, Q., Li, B., Yan, J.: Mimicdet: Bridging the gap between one-stage and two-stage object detection. In: European Conference on Computer Vision, pp. 541–557. Springer (2020)
38. Kumar, P., Sharma, A., Kota, S.R.: Automatic multiclass instance segmentation of concrete damage using deep learning model. IEEE Access **9**, 90330–90345 (2021)
39. G Braga, J.R., Peripato, V., Dalagnol, R., P Ferreira, M., Tarabalka, Y., OC Aragão, L.E., de F Campos, Velho H., Shiguemori, E.H., Wagner, F.H.: Tree crown delineation algorithm based on a convolutional neural network. Remote Sens. **12**(8), 1288 (2020)
40. Chadwick, A.J., Goodbody, T.R., Coops, N.C., Hervieux, A., Bater, C.W., Martens, L.A., White, B., Röeser, D.: Automatic delineation and height measurement of regenerating conifer crowns under leaf-off conditions using uav imagery. Remote Sens. **12**(24), 4104 (2020)
41. Shen, C., Zhao, X., Liu, Z., Gao, T., Xu, J.: Joint vehicle detection and distance prediction via monocular depth estimation. IET Intell. Transp. Syst. **14**(7), 753–763 (2020)
42. Zhou, C., Liu, Y., Sun, Q., Lasang, P.: Joint object detection and depth estimation in multiplexed image. IEEE Access **7**, 123107–123115 (2019)
43. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. Preprint at arXiv:1804.02767 (2018)

44. Urban, D., Caplier, A.: Time-and resource-efficient time-to-collision forecasting for indoor pedestrian obstacles avoidance. J. Imaging **7**(4), 61 (2021)
45. Cao, Y., Shen, C., Shen, H.T.: Exploiting depth from single monocular images for object detection and semantic segmentation. IEEE Trans. Image Process. **26**(2), 836–846 (2016)
46. Li, S., Xu, C., Xing, J., Ning, Y., Chen, Y.: Sdod: real-time segmenting and detecting 3d object by depth. Preprint at arXiv:2001.09425 (2020)
47. Kang, B., Lee, Y., Nguyen, T.Q.: Depth-adaptive deep neural network for semantic segmentation. IEEE Trans. Multimed. **20**(9), 2478–2490 (2018)
48. Liu, F., Shen, C., Lin, G., Reid, I.: Learning depth from single monocular images using deep convolutional neural fields. IEEE Trans. Pattern Anal. Mach. Intell. **38**(10), 2024–2039 (2015)
49. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning. ICML '01, pp. 282–289. Morgan Kaufmann Publishers Inc., San Francisco (2001)
50. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
51. Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., He, K.: Detectron. https://github.com/facebookresearch/detectron (2018)
52. Lin, X., Sánchez-Escobedo, D., Casas, J.R., Pardàs, M.: Depth estimation and semantic segmentation from a single rgb image using a hybrid convolutional neural network. Sensors **19**(8), 1795 (2019)
53. Wang, Y., Chao, W.-L., Garg, D., Hariharan, B., Campbell, M., Weinberger, K.Q.: Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8445–8453 (2019)
54. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4203–4212 (2018)
55. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 270–279 (2017)
56. Atapour-Abarghouei, A., Breckon, T.P.: Monocular segment-wise depth: monocular depth estimation based on a semantic segmentation prior. In: 2019 IEEE International Conference on Image Processing (ICIP), pp. 4295–4299. IEEE (2019)
57. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3234–3243 (2016)
58. Hu, J., Zhang, Y., Okatani, T.: Visualization of convolutional neural networks for monocular depth estimation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3869–3878 (2019)
59. Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: 2016 Fourth International Conference on 3D Vision (3DV), pp. 239–248 IEEE (2016)
60. Hu, J., Ozay, M., Zhang, Y., Okatani, T.: Revisiting single image depth estimation: toward higher resolution maps with accurate object boundaries. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1043–1051. IEEE (2019)
61. Kanayama, H., Ueda, T., Ito, H., Yamamoto, K.: Two-mode mapless visual navigation of indoor autonomous mobile robot using deep convolutional neural network. In: 2020 IEEE/SICE International Symposium on System Integration (SII), pp. 536–541. IEEE (2020)
62. Tsai, C.-Y., Nisar, H., Hu, Y.-C.: Mapless lidar navigation control of wheeled mobile robots based on deep imitation learning. IEEE Access **9**, 117527–117541 (2021)
63. Nguyen, A., Tran, Q.D.: Autonomous navigation with mobile robots using deep learning and the robot operating system. In: Robot Operating System (ROS), pp. 177–195. Springer (2021)
64. Xiong, Y., Zhang, X., Peng, J., Yu, W.: 3d depth map based optimal motion control for wheeled mobile robot. In: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2045–2050. IEEE (2017)
65. Li, C., Li, B., Wang, R., Zhang, X.: A survey on visual servoing for wheeled mobile robots. Int. J. Intell. Robot. Appl. **5**(2), 203–218 (2021)
66. Islam, S., Dias, J., Sunda-Meya, A.: On the design and development of vision-based autonomous mobile manipulation. In: IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society, pp. 1–6. IEEE (2021)
67. Xiao, X., Liu, B., Warnell, G., Stone, P.: Motion planning and control for mobile robot navigation using machine learning: a survey
68. Machkour, Z., Ortiz-Arroyo, D., Durdevic, P.: Classical and deep learning based visual servoing systems: a survey on state of the art. J. Intell. Robot. Syst. **104**(1), 1–27 (2022)
69. Furtado, J.S., Liu, H.H., Lai, G., Lacheray, H., Desouza-Coelho, J.: Comparative analysis of optitrack motion capture systems. In: Advances in Motion Sensing and Control for Robotic Applications, pp. 15–31. Springer (2019)
70. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 573–580. IEEE (2012)
71. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: mixing datasets for zero-shot cross-dataset transfer. IEEE Trans. Pattern Anal. Mach. Intell. **44**(3), 1623–1637 (2022)
72. Palou Visa, G.: Monocular depth estimation in images and sequences using occlusion cues (2014)
73. Tsai, Y.-S., Hsu, L.-H., Hsieh, Y.-Z., Lin, S.-S.: The real-time depth estimation for an occluded person based on a single image and openpose method. Mathematics **8**(8), 1333 (2020)
74. Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)
75. Jang, H., Ko, Y., Lee, Y., Jeon, M.: Light robust monocular depth estimation for outdoor environment via monochrome and color camera fusion. Preprint at arXiv:2202.12108 (2022)
76. Tian, Y., Hu, X.: Monocular depth estimation based on a single image: a literature review. In: Twelfth International Conference on Graphics and Image Processing (ICGIP 2020), vol. 11720, p. 117201. International Society for Optics and Photonics (2021)