Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
March-April 2022: 1–11

DOI: 10.1177/17298806221089198
journals.sagepub.com/home/arx
SAGE

Mobile robot 3D trajectory estimation on a multilevel surface with multimodal fusion of 2D camera features and a 3D light detection and ranging point cloud

Vinicio Rosas-Cervantes[1,2], Quoc-Dong Hoang[1,2], Sooho Woo[1,2] and Soon-Geul Lee[1,2]

Abstract

Nowadays, multi-sensor fusion is a popular tool for feature recognition and object detection. Integrating various sensors allows us to obtain reliable information about the environment. This article proposes a 3D robot trajectory estimation based on a multimodal fusion of 2D features extracted from color images and 3D features from 3D point clouds. First, a set of images was collected using a monocular camera, and we trained a Faster Region Convolutional Neural Network. Using the Faster Region Convolutional Neural Network, the robot detects 2D features from camera input and 3D features using the point's normal distribution on the 3D point cloud. Then, by matching 2D image features to a 3D point cloud, the robot estimates its position. To validate our results, we compared the trained neural network with similar convolutional neural networks. Then, we evaluated their response for the mobile robot trajectory estimation.

Keywords

Mobile robot, feature recognition, odometry and mapping, 3D localization

Date received: 24 August 2021; accepted: 7 March 2022

Topic: Mobile Robots and Multi-Robot Systems
Topic Editor: Nak-Young Chong
Associate Editor: Qinghao Meng

Introduction

Indoor environments may include slopes to transit from different multilevel areas. Most structured environments provide an even surface useful for robot mapping and exploration, where feature or image extraction is easier than in unstructured environments. Since modern indoor infrastructure includes slopes, mobile robots can navigate multilevel areas.

RGB cameras and light detection and ranging (LIDAR) sensors allow robots to explore structured even-surface scenarios with a robust response.[1–3]

RGB cameras capture scenes in 2D images, and we can classify one image into pixels and superpixels. LIDAR is an active sensor that does not depend on the lighting conditions and provides an accurate distance measurement. However, RGB camera performance depends on illumination, and LIDAR point clouds do not have texture or color information. To overcome those limitations, we can use multimodal sensor fusion.[4–7]

[1] Integrated Education Institute for Frontier Science and Technology (BK21Four), Kyung Hee University, Yongin, South Korea
[2] Department of Mechanical Engineering, Kyung Hee University, Yongin, South Korea

Corresponding author:
Soon-Geul Lee, Integrated Education Institute for Frontier Science and Technology (BK21 Four) and Department of Mechanical Engineering, Kyung Hee University, Yongin 17104, South Korea.
Email: sglee@khu.ac.kr

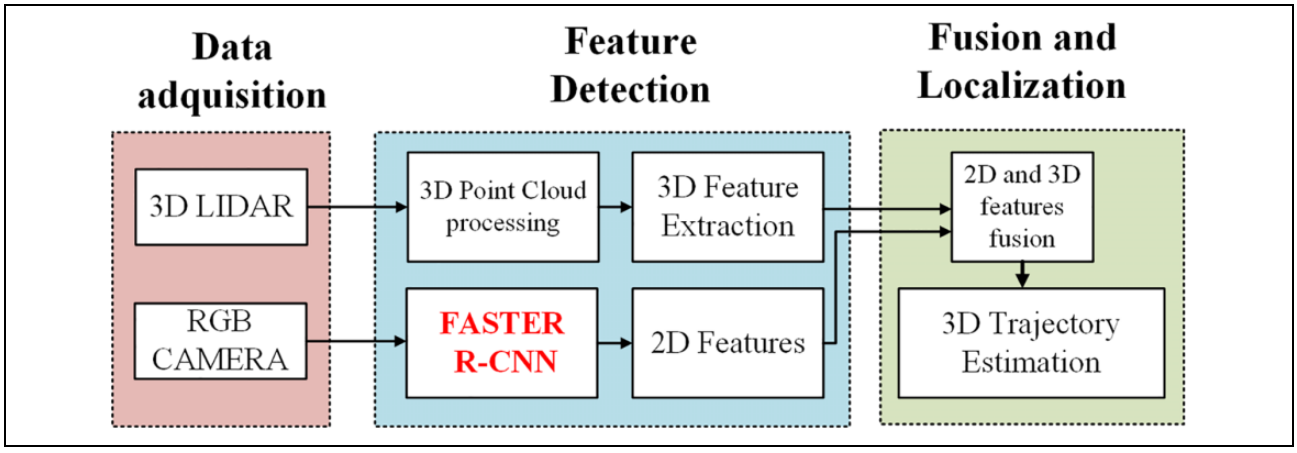

Figure 1. Proposed features-fusion model pipeline.

Multimodal fusion uses region levels,[8] and conditional random fields (CRFs)[9] help to model contextual information, but some LIDAR information is lost, resulting in labeling problems.[6,10] Since 3D LIDAR point clouds have plenty of noise, a solution is treating the point cloud as a mesh.[11,12] Point cloud labeling or mesh treatment is viable for large outdoor scenarios. However, for indoor scenarios, the feature extraction has limitations. Mobile robot localization on unstructured scenarios with uneven or multilevel surfaces is still a challenge.

The proposed method aims to provide an efficient solution for image feature detection and mobile robot localization in indoor environments. Nowadays indoor environments are provided with ramps to allow the connection of different levels in indoor environments, such as ramps or access points for wheelchairs. Similar feature extraction methods certainly identify and extract features with a good response. However, our method allows a robust feature extraction and robot localization in indoor environments including multilevel surfaces.

Since 3D point cloud treatment is critical for mobile robot exploration, we propose a multimodal sensor fusion for robot localization on multilevel surfaces employing an RGB camera and a 3D-LIDAR. Using a convolutional neural network (CNN), we extracted 2D features from RGB images and matched them into a 3D point cloud. To perform the 2D feature detection, the input from the RGB camera trains a Faster Region Convolutional Neural Network (Faster R-CNN).

In parallel, the robot extracts features from the 3D point cloud generated from the 3D LIDAR. Finally, we match the features from the 3D point cloud and the camera. Figure 1 shows the pipeline of the proposed concept. The contributions of this document are as follows: a trained neural network for 2D feature detection on multilevel scenarios and a 3D LIDAR–2D camera fusion that enables mobile robot trajectory estimation based on rapid feature detection. Figure 2 shows the proposed strategy to do the neural network training.

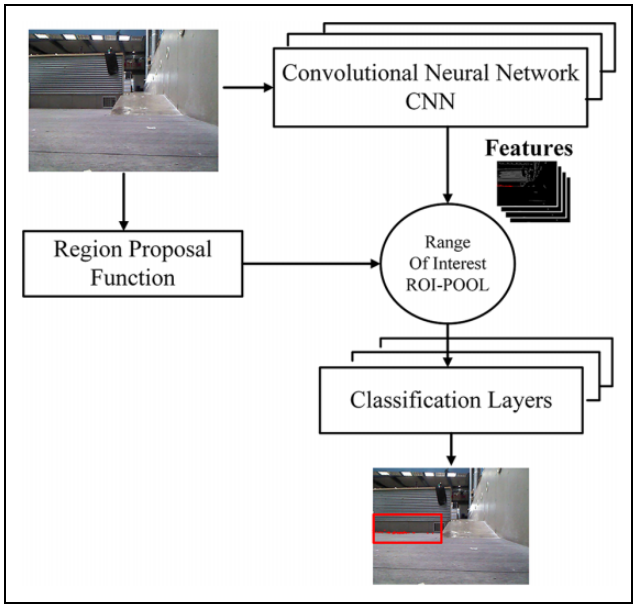

Figure 2. Feature extraction and training strategy using Faster R-CNN. Faster R-CNN: Faster Region Convolutional Neural Network.

Related work

To review the background studies, we divided them into two major areas. The first is CNNs for object and feature detection techniques. The second is the 3D LIDAR point cloud and RGB camera fusion.

ConvNet-based approaches for object detection

ConvNet is an image feature extractor.[13,14] The most popular object detectors are sliding window and region based. Sliding window ConvNet: This is a classic method for object detection. It employs a sliding window mechanism suggested by Sermanet et al.[15] Region-based ConvNets: R-CNN[16] and selective search[10] are methods for object proposal generation. The Faster R-CNN[17] uses spatial

pyramid pooling networks.[18] In Faster R-CNN, the image passes through a convolutional layer and finishes with end-to-end training. The fully convolutional network[19] improves object detection and time efficiency. Xiang et al. modify the Faster R-CNN using 3D voxel patterns.[20] Single-shot object detectors: You Only Look Once (YOLO)[21,22] and single-shot detector[23] use a single ConvNet. YOLO divides every input image into a grid, and each grid detects an object within a bounding box. One disadvantage is that the detection accuracy increases during training when YOLO tries to use the entire image, and the detection of small objects could be challenging. Cai et al.[24] implement detection at multiple intermediate networks, dealing with objects of different sizes. Oliveira et al.[25] proposed outdoor localization based on speed invariant inertial transformation and deep learning for the sensor. This localization has applications on terrain classification. With 3D object detection, Yang et al.[26] use convolutional features, and cascade classifiers to reject negative object proposals. Li et al.[27] use deep learning for fusion 2D LIDAR and inertia measurement unit (IMU). They used a recurrent neural network.

3D LIDAR and RGB camera fusion

3D LIDAR is critical for 3D scene perception, as it can capture data both at day and at night. Combining 3D LIDAR with 2D and 2.5D images allows for better 3D scene perception. Shinzato et al.[28] used a graphical method to recognize obstacles. Approaches such as Xiao et al.[6,29] use CRFs. For CNNs, Eitel et al.[30] propose object detection combining color images and depth maps. Schlosser et al.[31] transformed LIDAR point clouds into Horizontal disparity Height above the ground and Angle (HHA) fused with RGB. Asvadi et al.[32] integrated LIDAR and a color camera using deep learning for object detection. Bellone et al.[33] employ a support vector machine (SVM) which identifies roads using 3D LIDAR data. Zhou et al.[34] built an online learning road detector. Quan et al.[35] use the projection of 2D lines into 3D lines. However, the approach depends on geometric computation as initialization to bundle adjustments. Ouyang et al.[36] fuse odometry and wheel encoder to provide localization. Still, the approach is highly dependent on the gyroscope for positioning.

Mobile robots have sensors such as mono and stereo cameras, sonar, 2D, and 3D LIDAR.[37,38] 3D LIDAR is an important solution for high-level safety and environment recognition. The wide field of view, distance measurement, and night-vision capability are among the advantages of the 3D LIDAR. The cost of the integrated mechanical parts and the high-power requirements are major limitations for 3D LIDAR. Wisth et al.[39] use multi-sensory odometry for mobile robot localization. The information from visual reference is combined with IMU. He et al.[40] integrate global navigation satellite based on simultaneous localization and mapping pose estimation performing large-scale 3D map building. The authors used global positioning for pose estimation.

Feature detection uses monocular cameras fused with various sensors. For example, to perform road-background detection and classification, the multi-sensor divides an image into pixels and superpixels. Machine learning has important applications such as a mixture of Gaussian,[41] SVM,[4,42,43] and boosting[44] structure random forest.[6] These methods classify each unit independently, but the prediction could be noisy. Xu et al.[45] proposed a multi-sensory fusion using factor graph topology for optimal navigation. 3D LIDAR is crucial for autonomous vehicles too. Markov random fields can model LIDAR information generating a grid map.[46] Yuan et al.[47] proposed a location-based landmark recognition and used a novel quadrupole potential field for obstacle avoidance. Shinzato et al.[5] propose a simple camera–LIDAR fusion for road detection, but LIDAR and cameras have some drawbacks. Sensor fusion is an alternative for a single sensing modality. The extent of work on data fusion[26] techniques in multimodal object detection is classified into three categories: low level, combines the multiple sensor data; middle level, integrated the detected features; high Level, combines classified outputs.[48]

For pedestrian detection, Premebida et al.[49] use Velodyne LIDAR with color data. Combining color images and depth maps improves the performance of object detection. González et al.[50] use depth maps and color images as inputs. Schlosser et al.[31] use ConvNet-based fusion for pedestrian detection. Deep learning enhances the HHA data channel from 3D-LIDAR.[51,52] The authors employ color images and 3D-LIDAR point clouds as inputs. These inputs get region-wise features.

Multimodal feature detection

We propose a multimodal feature detection method based on 3D LIDAR and an RGB camera. The RGB images are collected from a Kinect camera and the 3D point clouds from a Quanergy 3D LIDAR. Since the projection of the 3D LIDAR points into the image is sparse, only the 3D point cloud main corners are extracted. The corner extraction process is described in sections "Spatial planar coordinates transformation" and "Division of voting space." Then, the extracted main corners are fused with the 2D features from the RGB images.

3D point cloud plane extraction

An efficient way to represent a 3D LIDAR point cloud is to segment it into small-scale 3D scenes. Kd-tree accelerates the point cloud segmentation using the normal vector for each point in the 3D point cloud. To proceed with the normal estimation, we use the K-nearest neighbor process to search around the pending points. Then, using the pending and neighbor points, the normal vectors are estimated using principal component analysis. Here, $\vec{\mathbf{n}} = (n_x, n_y, n_z)$

is normal for each vector and $\vec{r} = (x, y, z)$ is the incident direction of the 3D LIDAR. The constraint $(\vec{\mathbf{n}} \cdot \vec{r} < 0)$ adjusts the normal orientation. After the normal values are obtained, fussy clustering (FC) combines the normal angles with the Euclidean distance, segmenting the point cloud into physical planes. FC divides the points, separating them into two different facades. According to the number of points, FC allows us to check the point cloud space. Additionally, to reduce the computational time we employed two steps: spatial planar coordinates transformation and division of voting space.

Spatial planar coordinates transformation. We follow some procedures from Zhang et al. (1) for planar extraction. First, we transformed the spatial planar coordinates to polar coordinates using equation (1)

$$\rho = x \cos\theta \cos\phi + y \sin\theta \cos\phi + z \sin\phi \tag{1}$$

where θ is the angle between the normal $\vec{\mathbf{n}}$ and the x-axis, ϕ is the angle between the normal $\vec{\mathbf{n}}$ on the plane xoy, and ρ is the distance from origin O to the plane xoy. Figure 3(b) shows the normal $\vec{\mathbf{n}}$ with the corresponding angles θ, ϕ and the distance ρ. We use equations (2)–(4) for the vector conversion $\vec{\mathbf{n}} = (n_x, n_y, n_z)$ to the angles θ, ϕ and the distance ρ.

$$\theta = \arctan(n_y/n_x), \quad \theta \in [0,\ 2\pi] \tag{2}$$

$$\phi = \arcsin(n_z), \quad \phi \in [-\pi/2,\ \pi/2] \tag{3}$$

$$p = d \tag{4}$$

Division of voting space. For plane fitting with the width (ρ), theta (θ), and phi (ϕ) obtained from equations (2)–(4) to decide the ranges of the values in the space as $D = (d_{\max} - d_{\min})/\rho$, $T = 360/\theta$, and $P = 180/\phi$, where $d_{\max}$ and $d_{\min}$ represent the maximum and minimum distance values from the point to the plane. Therefore, we made a 3D array Vote(ρ, θ, ϕ). The array size is $D \times T \times P$, and all the elements started from 0. The values $plane_1$ and $plane_2$ were extracted using equations (5) and (6). Figure 3(a) shows the scenario scanned, (b) shows the original 3D point cloud, (c) and (d) shows the planes extracted from the 3D point cloud.

$$\overrightarrow{plane_1} \times \overrightarrow{plane_2} > AngleTheta \quad |Dist_{plane1} \tag{5}$$

$$-Dist_{plane2}| < Width \tag{6}$$

Feature extraction

Once we completed the 3D planes segmentation, the section "3D Point cloud feature extraction." describes the procedures for 3D feature extraction from the 3D segmented planes, and the section "2D RGB images feature extraction" describes the 2D feature extraction from RGB images.

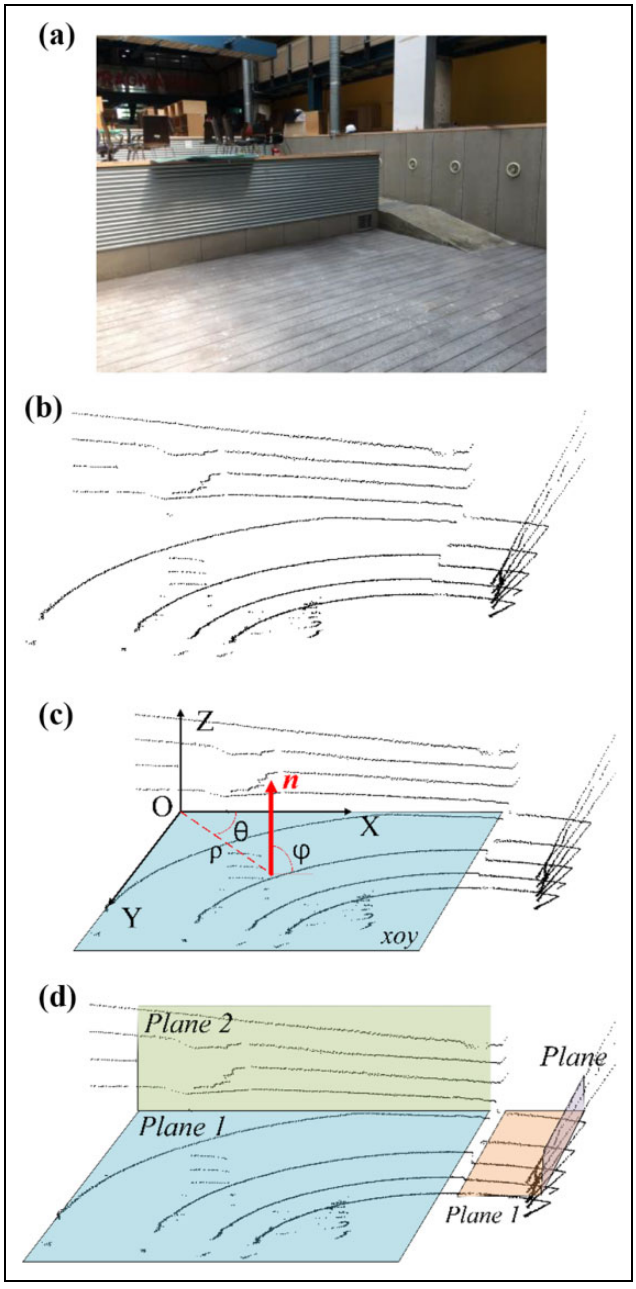

Figure 3. Extracted planes from the 3D point cloud: (a) scanned scenario image, (b) original 3D point cloud, (c) normal vector representation with corresponding angles, and (d) extracted 3D planes.

3D point cloud feature extraction. Once the 3D point clouds were segmented into planes, we found the intersecting points. The values plane_1 and plane_2 are perpendicular to each other, and their normal vectors are perpendicular to each plane. The value normal_1 is assigned to plane_1 and normal_2 to plane_2. The intersection vector noted as $\overrightarrow{\mathbf{P}_0}$, and the direction is given by $\vec{\mathbf{u}} = \overrightarrow{\mathbf{n}_1} \times \overrightarrow{\mathbf{n}_2}$ and

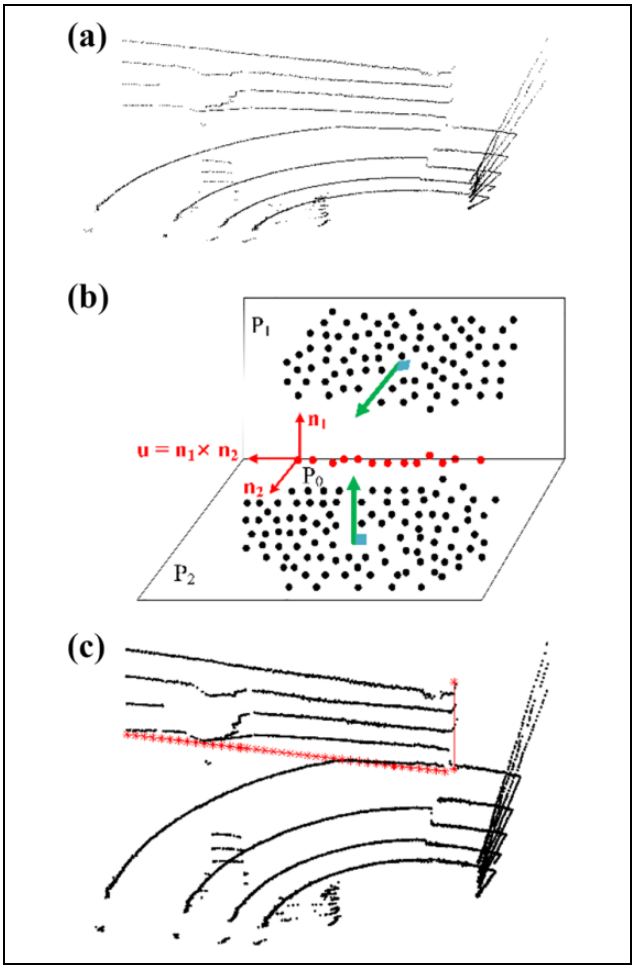

Figure 4. 3D point cloud plane intersection: (a) original point cloud, (b) intersected points located, and (c) intersected point located in the point cloud.

$\vec{u} = (u_x, u_y, u_z)$. The direction of the intersection vector is perpendicular to the normal $\overrightarrow{\mathbf{n}_1}$ and $\overrightarrow{\mathbf{n}_2}$. To determine the coordinates of the intersection points, we selected a nonzero coordinate $\vec{\mathbf{u}}$ ($u_z \neq 0$) and set the corresponding coordinate of $\overrightarrow{\mathbf{P}_0}$ to 0. $\overrightarrow{\mathbf{P}_0} = (x_0, y_0, 0)$ lies on the intersection line L. The plane_1 equation is given by $a_1x_0 + b_1y_0 + d_1 = 0$, and the plane_2 equation is $a_2x_0 + b_2y_0 + d_2 = 0$. Equations (7) and (8) are derived from the plane_1 and plane_2 equations, respectively. Equation (9) represents the obtained line feature. Figure 4(a) shows the original 3D point cloud, (b) shows the normal points in the intersection of the plane, and (c) shows the extracted features from the 3D point cloud.

$$x_0 = \frac{b_1d_2 - b_2d_1}{a_1b_2 - a_2b_1} \tag{7}$$

$$y_0 = \frac{a_2d_1 - a_1d_2}{a_1b_2 - a_2b_1} \tag{8}$$

$$L(s) = \frac{\left(\begin{vmatrix} b_1 & b_2 \\ d_1 & d_2 \end{vmatrix}, \begin{vmatrix} d_1 & d_2 \\ a_1 & a_2 \end{vmatrix}, 0\right)}{\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix}} + s(n_1 \times n_2) \tag{9}$$

2D RGB images feature extraction. A trained Faster R-CNN (2) detects the 2D features of RGB images using a region proposal network. Due to our robot employed a Kinect camera for collecting RGB images, the experiment environment has standard lighting conditions of 200–300 Lux. Since the experiment is oriented to indoor environments, we considered a location provided with average indoor lighting conditions. For the particular case of this experiment, the considered lighting conditions (minimum of 200–300 lux) allow the mobile robot to extract features.

To proceed with the neural network training and testing, we started collecting a set of 300 images of the experiment scenario. We used transfer learning for training our neural network. All the images were resized to a 224 × 224 resolution, transformed into gray scale and the Canny edge detector ran on every image. As a result, we obtained a CNN composed of 15 convolutional layers and two fully connected layers. We label 200 images identifying the main corners. The Faster R-CNN ran on the entire image during the training time and testing time. For testing, we used a set of 100 images. Figure 5(a) shows the 2D image capture by the Kinect camera, (b) shows the 2D gray scale of the captured image, and (c) shows the extracted edge using the Faster R-CNN.

The 2D features were extracted using equations (10) and (11). Here, i is the sequence number of the prior images, j is the index of features in each image, I^i_{prior} is the previous image, and I_{crt} is the current image, both of which were obtained from the Kinect camera. The 3D point cloud noted as S and the corresponding 2D image I^i_{prior} have six Degrees of Freedom (DoF)

$$f^{j,i}_{\text{prior}} = \text{Faster\_RCNN}(I^i_{\text{prior}}) \in \mathbb{R}^2 \tag{10}$$

$$f^{j}_{\text{crt}} = \text{Fast\_RCNN}(I_{\text{crt}}) \in \mathbb{R}^2 \tag{11}$$

2D and 3D feature fusion for robot localization

Adapting the procedure in Rublee et al.,[3] we projected the 2D RGB features f^i_{prior} and f^i_{crt} using ray tracing. Before proceeding with the 2D feature extraction, the Kinect camera was calibrated using the procedure described in Raposo et al.,[53] and the mobile robot odometry was calibrated using the procedure described in Borenstein and Feng.[54]

Then, a clustering algorithm run on the 2D feature's coordinates. Equation (12) calculates the candidate's images $I^k_{\text{candidate}}$ among I^i_{prior}, and $X_{\text{approximate}}$ is the robot position given by the IMU.

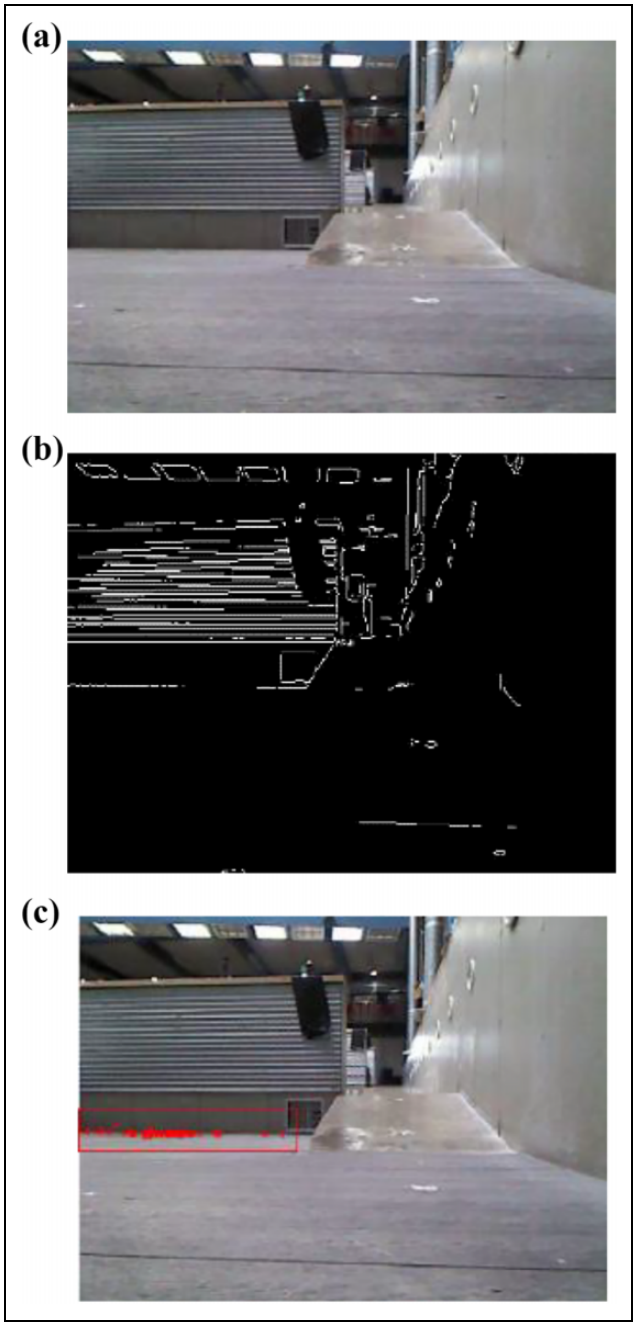

Figure 5. Feature extraction from RGB camera images using a Faster R-CNN: (a) original 2D image obtain by the Kinect camera, (b) main edges in the 2D image, and (c) edge extracted using the Faster R-CNN. Faster R-CNN: Faster Region Convolutional Neural Network.

$$(I^k_{\text{candidate}}, T^k_{\text{candidate}}) = \text{neighboring\_pos}(I^i_{\text{prior}}, T^i_{\text{prior}}, X_{\text{approximate}}) \tag{12}$$

Here, k is the index of the candidates, and the transformation T^i_{prior} represents the 3D position and orientation of the robot for I^i_{prior}. The function neighboring_pos finds the candidates by comparing the distance between T^i_{prior} and $X_{\text{approximate}}$ to the 3D coordinates. $I^k_{\text{candidate}}$ with low correlation concerning I_{crt} are removed using the random sample consensus. Equations (13) and (14) generate the 3D coordinate features $f^{j,k}_S$ of $f^{j,k}_{\text{candidate}}$ in the 3D point cloud S. The features are obtained from the ray-tracing algorithm for $T^i_{\text{candidate}}$ and are expressed using the pinhole camera model

$$Q^{j,k}_S(\lambda) = P^{-1}(f^{j,k}_{\text{candidate}}, T^k_{\text{candidate}}, K) \tag{13}$$

$$f^{j,k}_S = \text{RayTracing}(T^k_{\text{candidate}}, Q^{j,k}_S(\lambda)) \in \mathbb{R}^3 \tag{14}$$

Here, k is an index of the candidate and P^{-1} is the projection operator. The notation $Q_s(\lambda)$ represents the 3D coordinates in S. The RayTracing function finds the point that lies on the line between $T^k_{\text{candidate}}$ and $Q^{j,k}_S(\lambda)$. In equations (13) and (14), we can replace $f^{j,k}_{\text{candidate}}$ and $T^i_{\text{candidate}}$ with $f^{j,k}_{\text{prior}}$ and T^i_{prior}. Finally, the 3D to 2D projection ($f^{j,k}_S \in \mathbb{R}^3$) calculates the robot position. The largest cluster selected gives the multimodal fusion. Figure 6 shows the matching process between the 2D and 3D features.

Minimization of the robot 3D localization. Once we obtained the 2D and 3D feature fusion, the robot localization is minimized for every position and time $t + \Delta t$. We consider each pose at i and the coordinates of the matched features as $f^j_{\text{crt}} + f^j_S = (x_f, y_f, z_f)$. Each point is matched at each position $i + 1$, and then each feature coordinate is matched within the rotation matrices R_x and R_z. Using the values of R^f and t^f, we transformed to the robot's current position (x_r, y_r, z_r) using equations (15)–(17)

$$R_{x(\alpha_{\text{IMU}})} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_{\text{IMU}}) & \sin(\alpha_{\text{IMU}}) \\ 0 & -\sin(\alpha_{\text{IMU}}) & \cos(\alpha_{\text{IMU}}) \end{bmatrix} \tag{15}$$

$$R_{z(\beta_{\text{IMU}})} = \begin{bmatrix} \cos(\beta_{\text{IMU}}) & \sin(\beta_{\text{IMU}}) & 0 \\ -\sin(\beta_{\text{IMU}}) & \cos(\beta_{\text{IMU}}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

$$\begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = R_{x(\alpha_{\text{IMU}})} R_{z(\beta_{\text{IMU}})} \begin{bmatrix} x_f \\ y_f \\ z_f \end{bmatrix} \tag{17}$$

Using the values of (x_r, y_r, z_r) we minimized the error in the coordinate values obtained from odometry (x_o, y_o, z_o), as shown in equation (18)

$$E(R, t) = \min \sum_{i=1}^{N} ||f_i - R^f_i \cdot P_i - t^f_i|| \tag{18}$$

where P_i is the pose obtained from the previous feature alignment and f_i is the reference robot pose obtained from the IMU. Both f_i and P_i are 6×1 vectors $(x, y, z, \alpha, \gamma, \beta)$, where x, y, z are the 3D coordinates and α, γ, β are the

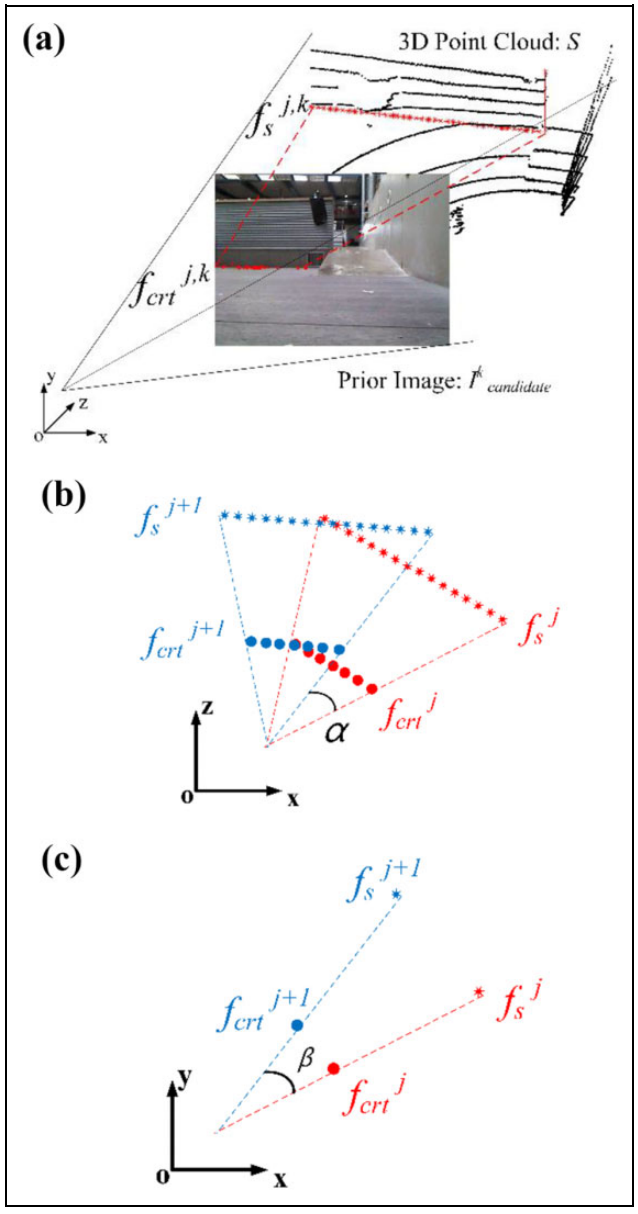

Figure 6. 2D and 3D feature projection and rotation: (a) ray tracing projection on the 3D point cloud, (b) RGB camera and 3D point cloud features on the *X*–*Z* plane, and (c) RGB camera and 3D point cloud features on the *X*–*Y* plane.

roll, pitch, and yaw angles, respectively. Algorithm 1 shows how multi-sensor fusion works to obtain the robot localization. We use the input from the collected set of local point clouds S_l and the set of images I_l to proceed with the multi-sensor localization. The number of iterations is given by the size of S_l. Using the procedure described in sections "Spatial planar coordinates transformation" and "Division of voting space," we generated a function "3D_point_cloud_feature." Then we obtained the features f_S. From the section "2D RGB images feature extraction," we obtained the image features f_{crt}. Using the fusion described in the section "2D and 3D feature fusion for robot localization," we combined the features f_S and f_{crt}, obtaining the local coordinates x_l, y_l, z_l. Then using the process described in the section "Minimization of the robot 3D localization," the local coordinates were transformed into global coordinates x_G, y_G, z_G. Finally, the robot coordinates are minimized using equation (18).

Algorithm 1. Multi-sensor localization.

1: **Input:** Set Local Clouds (S_l), Set images (I_l)
2: **Output:** Global Robot Localization: $P_{G:m}$
3: **for** $i = 1$ to $n = \text{Size}(S_l)$ **do**
4: $f_S = $ 3D_point_cloud_feature(S_l) (S_l)
5: $f_{\text{crt}} = \text{Faster\_RCNN}(I_l)$
6: $[x_l, y_l, z_l]_{\text{local}} = \text{fusion}(f_S, \quad f_{\text{crt}})$
7: $[x_G, y_G, z_G]_{\text{global}} = \text{rotate}(x_l, y_l, z_l)$
8: $\min[x_G, y_G, z_G] \quad \overrightarrow{P_{G:M}} = [x_{G:m}, y_{G:m}, z_{G:m}]$
9: **end for**

Results

To test our experiment, we chose a university location provided with a multilevel surface. The location is divided into three sections, and each section is connected with a 10° ramp, as shown in Figure 7(c). We used a Kobuki robot,[55] a Kinect camera,[56] a Quanergy M8 3D LIDAR manufactured by Quanergy Systems (Sunnyvale, California, USA),[57] and a laptop computer (8 GB RAM and processor intel i7) running MATLAB. Figure 7(a) shows the employed mobile robot and (b) shows the experiment scenario. For the experiment, the linear velocity of the robot was considered without any slip. The robot moved with a linear speed of 0.05 m/s, and the sampling period was 1s. We used a scanning frequency of 0.5 m for LIDAR point cloud registration. Using the mentioned sampling parameters, the robot had enough time for the multi-sensor acquisition.

For the neural network training, we started collecting all the RGB camera images and trained a Faster R-CNN. Our method employs grayscale segmentation before proceeding with the 2D image feature extraction. The proposed segmentation allows a faster extraction of the major obstacles in front of the mobile robot. Then the robot localizes itself using the features from the trained neural network and the 3D point cloud.

The proposed method was compared with similar transfer learning CNNs as Vgg16 and AlexNet. The response of our method using the Faster R-CNN has a lower training loss compared to the mentioned techniques. The three methods have a high training accuracy and a robust response for object recognition. AlexNet and Vgg16 have trajectory errors of 0.28 m and 0.24 m, respectively. However, our method with the Faster R-CNN reduces the trajectory error to 0.16 m. Table 1 shows a quantitative comparison of the proposed method compared with AlexNet and Vgg16. To optimize the neural network weight

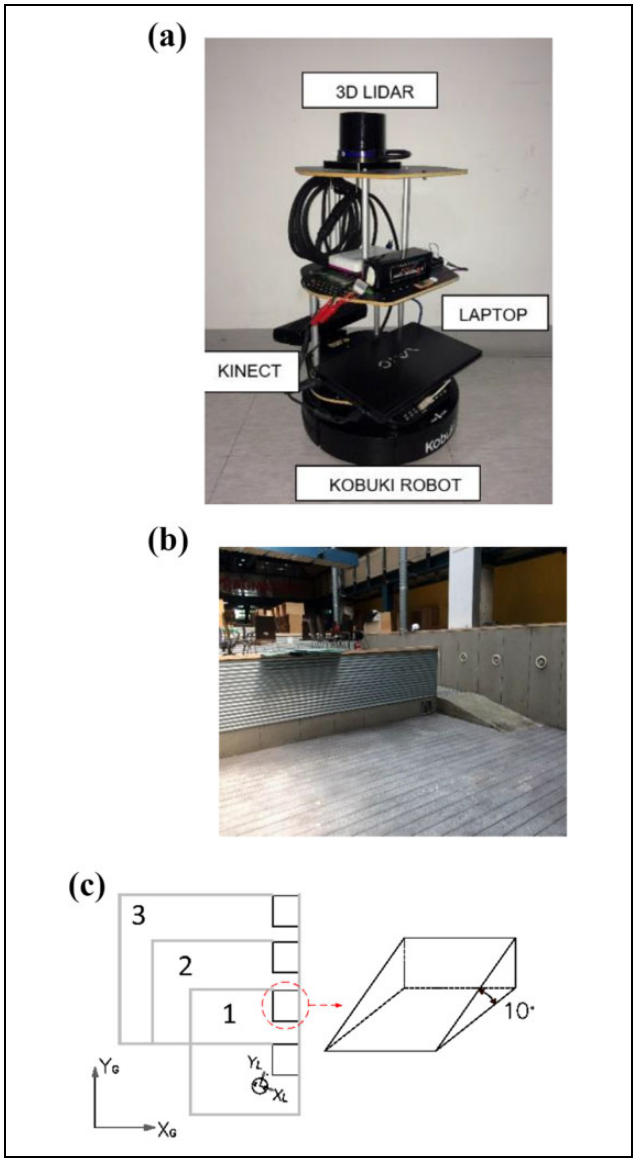

Figure 7. Scenario and equipment used for the experiment: (a) mobile robot, (b) experiment scenario, and (c) scenario diagram showing three different levels.

Table 1. Proposed faster neural network training comparison.

	Trajectory error (m)	Training accuracy (%)	Training loss
Faster R-CNN	0.16	100	0.0013
Vgg16	0.24	99.99	0.0029
AlexNet	0.28	99.99	0.0042

Faster R-CNN: Faster Region Convolutional Neural Network.

updating, we evaluated the training loss. The training loss allows us the interpretation of how well the model is doing for every set. The lower the training loss, the better is the model. Unlike accuracy, training loss is not a percentage, and it is a summation of errors made for each example in training or validation sets. Loss values imply how well the model behaves after each iteration of optimization. Ideally, we expect the reduction of loss after each or several iterations. To obtain the best possible accuracy, we use a Mini Bach size of 128 and a max epoch of 100 for optimal training response. Figure 8(a) shows a quantitate comparison in the training process of the proposed neural network versus similar networks such as VGG16 and AlexNet.

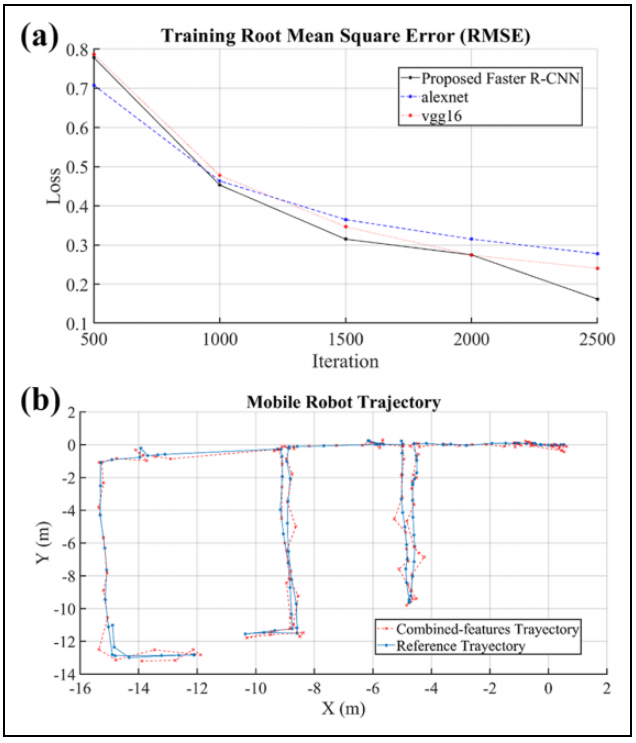

Figure 8. Mobile robot training performance and trajectory: (a) training root mean square error comparison and (b) mobile robot trajectory compared with the reference values.

The ground truth for the robot localization was obtained using the odometer and the IMU integrated into the robot. The information collected from these sensors was fused using the extended Kalman filter within a ROS node. Figure 8(b) shows a comparison of the obtained trajectory versus the ground truth trajectory. The axes X and Y represent the coordinates of each robots' pose measured in meters. As additional validation, we calculated the root mean square error (RMSE) comparing our trajectory versus the ground truth in each level. Table 2 shows the quantitative registration results of the obtained trajectory.

Discussion

3D mapping and registration face different challenges, such as overlapping areas or sparse features. Our robotic registration framework detects and computes 2D and 3D features using as input two sensors for multimodal localization.

Table 2. RSME for multi-sensor registration.

	Multi-sensor input ID		
RMSE (m)	#1–#40 Level 1	#40–#90 Level 2	#90–#130 Level 3
X-axis	0.10	0.04	0.02
Y-axis	0.01	0.04	0.01
Z-axis	0.01	0.01	0.01

RSME: root mean square error.

The proposed plane extraction is based on the geometric properties of the 3D point cloud. For the image feature extraction, we trained an artificial neural network using transfer learning. Feature matching into the 3D point cloud is based on the reference described in the section "2D and 3D feature fusion for robot localization." Lastly, we proposed Algorithm 1 for multi-sensory fusion and mobile robot localization. In the proposed localization, a mobile robot only uses visual reference (color images) and the surrounding environment (3D point clouds). Although we obtained a robust 3D localization during a dynamic scan, we identified two weaknesses. First, if the robot's velocity is higher than the established linear velocity of 0.05 m/s, the robot may not have enough time to process the data from all sensor inputs. Second, the robot cannot crossover a ramp bigger than 10° due to the wheel diameter. The proposed localization approach can impact on the service industry, improving the monitoring and control of mobile robots in multilevel areas. As future work, the experiment will be performed in outdoor scenarios, extending the neural network training for detecting and extracting more scenario features.

Conclusions

We presented a 2D and 3D feature fusion for mobile robot localization in a multilevel area. The 3D point cloud feature extraction based on plane segmentation reduces the point cloud processing time. The Faster R-CNN identifies the main corners in 2D images. The mobile robot extracts 3D features using 3D point cloud processing. As presented in the Introduction, plenty of methods still rely on positioning sensors as IMU or global positioning system (GPS), which offer a good solution in outdoor scenarios. On the other hand for indoor scenarios, the response of these sensors is limited. The proposed method presents an RMSE around the axis X in 0.053 m and around the axis Y in 0.02 m. Those values are acceptable for mobile robot indoor exploration, meaning that our method has a reliable response in localization, providing an alternative to sensors as IMU or GPS. In terms of neural network efficiency, the proposed method reduced the training loss significantly compared to Vgg16 in 55.18%, and Alexnet in 69.05% keeping the lowest robot trajectory error.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was supported by Ministry of Trade, Industry and Energy under Robot Industrial Core Technology Development Project program (20015052 and K_G012000921401) supervised by the KEIT.

ORCID iD

Vinicio Rosas-Cervantes https://orcid.org/0000-0002-0913-9071

References

1. Lowe DG. Object recognition from local scale-invariant features. In *Proceedings of the IEEE international conference on computer vision*, Cambridge, USA, 6 August 1999, pp. 1150–1157. New York, USA: IEEE.
2. Bay H, Ess A, Tuytelaars T, et al. Speeded-up robust features (SURF). *Comput Vis Image Underst* 2008; 110: 346–359.
3. Rublee E, Rabaud V, Konolige K, et al. ORB: an efficient alternative to SIFT or SURF. In: *2011 International conference on computer vision*, Barcelona, 2–6 November 2011, pp. 2564–2571. New York, USA: IEEE.
4. Liu H, Yu Y, Sun F, et al. Visual–Tactile fusion for object recognition. *IEEE Trans Autom Sci Eng* 2017; 14: 996–1008.
5. Shinzato PY, Wolf DF, and Stiller C. Road terrain detection: avoiding common obstacle detection assumptions using sensor fusion. In: *2014 IEEE intelligent vehicles symposium proceedings*, Dearborn, MI, USA, 8–11 June 2014, pp. 687–692. New York, USA: IEEE.
6. Xiao L, Dai B, Liu D, et al. CRF based road detection with multi-sensor fusion. In: *2015 IEEE intelligent vehicles symposium (IV)*, 2015, pp. 192–198.
7. Xiao W, Liu H, Sun F, et al. Likelihood confidence rating based multi-modal information fusion for robot fine operation. In: *2014 13th international conference on control automation robotics & vision (ICARCV)*, Singapore, 10–12 December 2014, pp. 259–264.
8. Zhang R, Candra SA, Vetter K, et al. Sensor fusion for semantic segmentation of urban scenes. In: *2015 IEEE international conference on robotics and automation (ICRA)*, Washington, 25–30 May 2015, pp. 1850–1857. New York, USA: IEEE.
9. Shotton J, Winn J, Rother C, et al. TextonBoost: joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Computer vision—ECCV 2006*, Berlin, Heidelberg, 2006, pp. 1–15.
10. Kolmogorov V and Zabin R. What energy functions can be minimized via graph cuts? *IEEE Trans Pattern Anal Mach Intell* 2004; 26: 147–159.

11. Wu H, Zha H, Luo T, et al. Global and local isometry-invariant descriptor for 3D shape comparison and partial matching. In: *2010 IEEE computer society conference on computer vision and pattern recognition*, San Francisco, 13–18 June 2010, pp. 438–445. New York, USA: IEEE.
12. Guo Y, Sohel F, Bennamoun M, et al. Rotational projection statistics for 3D local surface description and object recognition. *Int J Comput Vis* 2013; 105: 63–86..
13. LeCun Y, Boser B, Denker JS, et al. Backpropagation applied to handwritten zip code recognition. *Neural Comput* 1989; 1: 541–551..
14. Krizhevsky A, Sutskever I, and Hinton GE. ImageNet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*, 2012, pp. 1097–1105.
15. Sermanet P, Eigen D, Zhang X, et al. Overfeat: integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229, 2013.
16. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *2014 IEEE conference on computer vision and pattern recognition*, Columbus, 23–24 June 2014, pp. 580–587. New York, USA: IEEE.
17. Girshick R. Fast R-CNN. In: *2015 IEEE international conference on computer vision (ICCV)*, Santiago de Chile, 7–13 December 2015, pp. 1440–1448. New York, USA: IEEE.
18. He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans Pattern Anal Mach Intell* 2015; 37: 1904–1916.
19. Shelhamer E, Long J, and Darrell T. Fully convolutional networks for semantic segmentation. arXiv preprint arXiv:1605.06211, 2016.
20. Xiang Y, Choi W, Lin Y, et al. Subcategory-Aware convolutional neural networks for object proposals and detection. In: *2017 IEEE winter conference on applications of computer vision (WACV)*, Santa Rosa, 24–31 March 2017, pp. 924–933. New York, USA: IEEE.
21. Redmon J and Farhadi A. YOLO9000: better, faster, stronger. https://arxiv.org/abs/1612.08242 (2016 accessed 18 March 2022).
22. Redmon J, Divvala S, Girshick R, et al. You Only Look Once: unified, real-time object detection. In: *2016 IEEE conference on computer vision and pattern recognition (CVPR)*, Las Vegas, 27–30 June 2016, pp. 779–788. New York, USA: IEEE.
23. Liu W, Anguelov D, Erhan D, et al. SSD: single shot MultiBox detector. In: *Computer Vision—ECCV 2016*, Cham, 2016, pp. 21–37.
24. Cai Z, Fan Q, Feris RS, et al. A unified multi-scale deep convolutional neural network for fast object detection. In: *Computer Vision—ECCV 2016*, Cham, 2016, pp. 354–370.
25. Oliveira FG, Neto AA, Howard D, et al. Three-dimensional mapping with augmented navigation cost through deep learning. *J Intell Robot Syst* 2021; 101: 50..
26. Yang F, Choi W, and Lin Y. Exploit all the layers: fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers. In: *2016 IEEE conference on computer vision and pattern recognition (CVPR)*, Las Vegas, 27–30 June 2016, pp. 2129–2137. New York, USA: IEEE.
27. Li C, Wang S, Zhuang Y, et al. Deep sensor fusion between 2D laser scanner and IMU for mobile robot localization. *IEEE Sens J* 2021; 21: 8501–8509.
28. Shinzato PY, Wolf DF, and Stiller C.Road terrain detection: avoiding common obstacle detection assumptions using sensor fusion. In: *IEEE intelligent vehicles symposium, proceedings*, 2014, pp. 687–692.
29. Xiao L, Wang R, Dai B, et al. Hybrid conditional random field based camera-LIDAR fusion for road detection. *Inf Sci* 2018; 432: 543–558.
30. Eitel A, Springenberg JT, Spinello L, et al. Multimodal deep learning for robust RGB-D object recognition. In: *IEEE international conference on intelligent robots and systems*, Hamburg, 28 September–2 October 2015, pp. 681–687. New York, USA: IEEE.
31. Schlosser J, Chow CK, and Kira Z. Fusing LIDAR and images for pedestrian detection using convolutional neural networks. In: *2016 IEEE international conference on robotics and automation (ICRA)*, Stockholm, 16–20 May 2016, pp. 2198–2205. New York, USA: IEEE.
32. Asvadi A, Garrote L, Premebida C, et al. Multimodal vehicle detection: fusing 3D-LIDAR and color camera data. *Pattern Recogn Lett* 2018; 115: 20–29.
33. Bellone M, Reina G, Caltagirone L, et al. Learning traversability from point clouds in challenging scenarios. *IEEE Trans Intell Transp Syst* 2018; 19: 296–305.
34. Zhou S, Gong J, Xiong G, et al. Road detection using support vector machine based on online learning and evaluation. In: *2010 IEEE intelligent vehicles symposium*, La Jolla, California, USA, 2010, pp. 256–261.
35. Quan M, Piao S, He Y, et al. Monocular visual SLAM with points and lines for ground robots in particular scenes: parameterization for lines on ground. *J Intell Robot Syst* 2021; 101: 72.
36. Ouyang M, Cao Z, Guan P, et al. Visual-gyroscope-wheel odometry with ground plane constraint for indoor robots in dynamic environment. *IEEE Sens Lett* 2021; 5: 1–4.
37. Urmson C, Anhalt J, Bagnell D, et al. Autonomous driving in urban environments: Boss and the Urban Challenge. *J Field Robot* 2008; 25: 425–466.
38. Ziegler J, Bender P, Schreiber M, et al. Making bertha drive—an autonomous journey on a historic route. *IEEE Intell Transp Syst Mag* 2014; 6: 8–20.
39. Wisth D, Camurri M, Das S, et al. Unified multi-modal landmark tracking for tightly coupled LIDAR-visual-inertial odometry. *IEEE Robot Autom Lett* 2021; 6: 1004–1011.
40. He G, Yuan X, Zhuang Y, et al. An integrated GNSS/LiDAR-SLAM pose estimation framework for large-scale map building in partially GNSS-denied environments. *IEEE Trans Instrum Meas* 2021; 70: 1–9.
41. Dahlkamp H, Kaehler A, Stavens D, et al. Self-supervised monocular road detection in desert terrain. In: *Science and*

systems II, 16–19 August 2006. University of Pennsylvania: Philadelphia, Pennsylvania, USA.
42. Alon Y, Ferencz A, and Shashua A. Off-road path following using region classification and geometric projection constraints. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, New York, NY, USA 17–22 June 2006, pp. 689–696. New York, USA: IEEE.
43. Zhu W, Miao J, Hu J, et al. Vehicle detection in driving simulation using extreme learning machine. *Neurocomputing* 2014; 128: 160–165.
44. Fritsch J, Kühnl T, and Kummert F. Monocular road terrain detection by combining visual and spatial information. *IEEE Trans Intell Trans Syst* 2014; 15: 1586–1596.
45. Xu J, Yang G, Sun Y, et al. A multi-sensor information fusion method based on factor graph for integrated navigation system. *IEEE Access* 2021; 9: 12044–12054.
46. Liu H, Qin J, Sun F, et al. Extreme kernel sparse learning for tactile object recognition. *IEEE Trans Cybern* 2017; 47: 4509–4520.
47. Yuan W, Li Z, and Su CY. Multisensor-based navigation and control of a mobile service robot. *IEEE Trans Syst Man Cybern: Syst* 2021; 51: 2624–2634.
48. Garcia F, Martin D, Escalera ADL, et al. Sensor fusion methodology for vehicle detection. *IEEE Intell Trans Syst Mag* 2017; 9: 123–133.
49. Premebida C, Carreira J, Batista J, et al. Pedestrian detection combining RGB and dense LIDAR data. In: *2014 IEEE/RSJ international conference on intelligent robots and systems*, Chicago, 14–18 September 2014, pp. 4112–4117. New York, USA: IEEE.
50. González A, Vázquez D, López AM, et al. On-board object detection: multicue, multimodal, and multiview random forest of local experts. *IEEE Trans Cybern* 2016; 47: 3980–3990.
51. Gupta S, Girshick R, Arbeláez P, et al. Learning rich features from RGB-D images for object detection and segmentation. In: *Computer vision—ECCV 2014*, Cham, 2014, pp. 345–360.
52. Chen X, Ma H, Wan J, et al. Multi-view 3d object detection network for autonomous driving, https://arxiv.org/abs/1611.07759 (2016 accessed 18 March 2022).
53. Raposo C, Barreto JP, and Nunes U. Fast and accurate calibration of a Kinect sensor. In: *2013 international conference on 3D vision—3DV*, Seattle, WA, USA, 29 June–1 July 2013, pp. 110–111. New York, USA: IEEE.
54. Borenstein J and Feng L. Measurement and correction of systematic odometry errors in mobile robots. *IEEE Trans Robot*. 1996; 12(6): 869–880.
55. Kobuki mobile robot, Yujin Robot, Seoul, South Korea. http://kobuki.yujinrobot.com/about2/ (2015, accessed 18 March 2022).
56. Blair TB and Davis CE. Innovate engineering outreach: a special application of the Xbox 360 Kinect sensor. In: *2013 IEEE frontiers in education conference (FIE)*, Oklahoma City, OK, 23–26 October 2013, pp. 1279–1283. New York: IEEE.
57. Quanergy M8 3D LIDAR sensor, Sunnyvale, California, USA. https://quanergy.com/wp-content/uploads/2019/12/M8-Datasheet_QPN-98-00037-Rev-M.pdf (2020, accessed 18 March 2022).