Received: 10 March 2023 | Revised: 5 February 2024 | Accepted: 7 March 2024

DOI: 10.1002/rob.22319

RESEARCH ARTICLE

WILEY

Developing an expansion-based obstacle detection using panoptic segmentation

Saied Pirasteh[1,2] | Masood Varshosaz[1,3] | Samira Badrloo[1,3] | Jonathan Li[4]

[1]Shaoxing Institute of Artificial Intelligence, Shaoxing University, Shaoxing, Yuecheng District, Zhejiang Province, China

[2]Department of Geotechnics and Geomatics, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, India

[3]Department of Photogrammetry and Remote Sensing, K.N. Toosi University of Technology, Tehran, Iran

[4]Department of Geography and Environmental Management, Geospatial Sensing and Data Intelligence Lab, University of Waterloo, Waterloo, Ontario, Canada

Correspondence
Masood Varshosaz, Shaoxing Institute of Artificial Intelligence, Shaoxing University, 508 West Huancheng Rd, Postal Code 312000, Shaoxing, Yuecheng District, Zhejiang Province, China.
Email: varshosazm@usx.edu.cn and varshosazm@kntu.ac.ir

Abstract

Safe Micro Aerial Vehicle (MAV) navigation requires detecting and avoiding obstacles. For safe MAV navigation, expansion-based algorithms are effective for detecting obstacles. However, accurate and real-time obstacle detection is a fundamental challenge. Some traditional methods focus on extracting geometric features from images and applying geometric constraints to identify potential obstacles. Others may leverage machine learning algorithms for object detection and classification, using features such as texture, shape, and context to distinguish obstacles from background clutter. The choice of approach depends on factors such as the specific requirements of the application, the complexity of the scene, and the available computational resources. Since obstacles, in reality, take the form of objects (e.g., persons, walls, pillars, trees, automobiles, and other structures), it is preferable to represent them according to human comprehension and as objects. Therefore, the objective of this study is to reflect on the previous research and address the issues mentioned above by extracting objects from the fisheye image using a panoptic deep-learning network. The extracted object regions are, then, used to identify obstacles with a novel area-based expansion rate we developed in a previous study. We compared the accuracy of obstacle detection in our proposed method to the existing method when moving forward and to the right; thus, we improved it between 10% and 18%, respectively. In addition, compared with the existing method, and due to replacing a single object with multiple regions, obstacle-detection runtime for forward and right direction is 15.71 and 25.5 times faster, respectively, and the required match points have decreased by 49% and 55%.

KEYWORDS
deep learning, expansion-based method, fisheye image, MAVs, objects, obstacle detection, panoptic segmentation

1 | INTRODUCTION

Obstacle detection is essential for self-driving Micro Aerial Vehicles (MAVs). Detection of obstacles can be categorized into image-based (Aharchi & Kbir, 2022; Shi et al., 2023), sensor-based (Goodin et al., 2021; Wilshin et al., 2023), and hybrid (Beul et al., 2017; Hu et al., 2020). Image-based approaches use image information, such as gray levels (Mashaly et al., 2016), points (Aguilar et al., 2017; Al-Kaff et al., 2017), edges (Mashaly et al., 2016), and regions (Badrloo et al., 2022b). Initially, image-based approaches employed conventional image processing algorithms for obstacle detection (Al-Kaff et al., 2017; Ball et al., 2016, 2017; Chen, 2019; Gharani & Karimi, 2017; Häne et al., 2017; Jiang et al., 2023; Tsai et al., 2018; Zhang et al., 2020). Although, real-time and precise detection of obstacles was challenging using conventional image processing techniques (Badrloo et al., 2022b; Bharati et al., 2018; Liu, Li, Liu, et al., 2021; Tijmons et al., 2017). Recent studies have demonstrated significant developments in deep-learning artificial neural networks, which have improved image processing quality (LeCun et al., 2015). Therefore, several obstacle-detection methods have recently employed deep-learning artificial neural networks for obstacle detection (Gayathri et al., 2023; He et al., 2022; Opromolla & Fasano, 2021; Pehlivan et al., 2019; Rane et al., 2020). Following this, image-based approaches to obstacle detection are studied so that deep-learning neural networks can be employed in accurate and real-time obstacle detection for MAVs. There are two categories of image-based methods: (a) stereo methods and (b) monocular methods (Badrloo et al., 2022a).

Two cameras are employed in stereo techniques to detect obstacles (Barry et al., 2018; Grinberg & Ruf, 2021; Sun et al., 2022). In addition, deep-learning neural networks have recently been used for stereo techniques (Dairi et al., 2018; Yu et al., 2023). Evidently, stereo approaches are computationally inefficient for microprocessors in MAVs (Vargas et al., 2021; Zhang et al., 2023). Therefore, a strong Graphics Processing Unit (GPU) is necessary (Mendes et al., 2017). Moreover, as these technologies rely on precise system calibration, any errors can deteriorate the method's reliability with time (Lin et al., 2020).

Contrary to that, monocular obstacle-detection algorithms employ one camera. These techniques detect obstacles by installing cameras on the sides or front of the autonomous vehicle (Badrloo et al., 2022b; Lee et al., 2021; Shi et al., 2023). Monocular methods are classified as follows: appearance-based (Rane et al., 2020; Talele et al., 2019), motion-based (Tsai et al., 2018), depth-based (Hatch et al., 2021; Ho et al., 2018), and expansion-based (Escobar-Alvarez et al., 2018; Lee et al., 2021). Recently, these methods have employed deep-learning neural networks for real-time applications (see Section 2). Appearance-based approaches are based on a simple assumption and usually encounter difficulties in the outdoor environment due to the presence of numerous obstacles and objects (Mashaly et al., 2016). In addition, motion-based approaches typically fail to identify frontal obstacles. Depth-based approaches generate a complete depth map of the surrounding environment and use it for obstacle detection (Häne et al., 2017; Ho et al., 2018). However, when detecting obstacles, we only want close ones. Hence, a depth map with numerous calculations is unnecessary. Expansion-based approaches are one of the easiest and quickest monocular algorithms that use the expansion of an object (in sequential frames) to detect obstacles (Badrloo et al., 2022b; Lee et al., 2021). In fact, they work on a much simpler principle than methods based on appearance, motion, and depth. For this, we have chosen this technique to detect obstacles.

Currently, expansion-based methods are limited because most rely on conventional image processing techniques, such as point extraction and matching algorithms (Al-Kaff et al., 2017; Padhy et al., 2019; Zeng et al., 2016). As a result of using points and detecting points of the obstacle, they lose a complete and correct detection of the obstacle.

To overcome this challenge, Badrloo et al. (2022b) presented new research that employs image regions for obstacle detection. So that it captures two sequential images from the front-mounted camera of the drone. It segments the regions of the second image and extracts the points in the first and second images that are matched (Badrloo et al., 2022b). Then, the area of regions in two sequential frames is calculated using the matched points. If the area expansion rate exceeds a specified threshold, the area is considered to be an obstacle (Badrloo et al., 2022b). Due to the use of conventional image processing techniques for segmentation, Seeded Region Growing (SRG) (Asmussen et al., 2015), their research requires a sufficient threshold value for each image. Additionally, at least three noncollinear match points are required in each region to calculate the area. Also, extracting and analyzing each image region in images with various objects is time-consuming. Although it divides the image into three equal parts to reduce the obstacle-detection time and selects the desired part of the image based on the direction of the MAV's movement, the usage of a part of the image restricts the observation space, and the obstacle-detection time is still long. Nevertheless, it is capable of detecting an obstacle completely and accurately to a high percentage. However, due to the high processing time and the need for a large number of matched points, their method is unsuitable for use in real-time applications. While complete and real-time obstacle detection is crucial for the safe navigation of MAVs, especially, in situations where MAVs are traveling through a complex environment with various objects.

As it can be seen, some traditional methods focus on extracting geometric features from images and applying geometric constraints to identify potential obstacles. Others may leverage machine learning algorithms for object detection and classification, using features such as texture, shape, and context to distinguish obstacles from background clutter. The choice of approach depends on factors, such as the specific requirements of the application, the complexity of the scene, and the available computational resources. Thus, in continuation of our previous work (i.e., Badrloo et al., 2022b), in this paper, we have employed a panoptic segmentation with deep learning (Kirillov et al., 2019) to extract image objects. The algorithm calculates the expansion rate of the object across sequential frames and detects the obstacles. Additionally, we used a fisheye camera that can scan wide parts of the surrounding area. This camera is positioned in front of MAV. Consequently, the suggested method improves existing methods and provides more accurate outcomes. Overall, the main contributions of this study are as follows:

- Several regions are replaced with each extracted object.
- In contrast to conventional segmentation techniques, in panoptic segmentation (Kirillov et al., 2019), obstacles are extracted accurately and without merging.
- The need for match points and time for obstacle detection is reduced significantly.
- The detected obstacle also has a label.
- It uses the entire fisheye image to detect the obstacle.

This article contains five sections. Section 2 explains the related works for monocular techniques with deep-learning neural networks.

The proposed algorithm steps for obstacle detection are described in Section 3. Experiments evaluating the proposed method and comparing its findings to those of one of the most effective region-based techniques by Badrloo et al. (2022b) are analyzed in Section 4 and followed by limitations in Section 5. Additionally, Section 6 offers conclusions, future research, and suggestions.

2 | RELATED WORK

In monocular methods, the camera is located around or in front of the MAV for obstacle detection (Liu, Li, Luo, et al., 2021; Rane et al., 2020; Shikishima et al., 2022; Sleaman et al., 2023). Then, the camera captures an image at each instant, and the captured image or sequential images are processed for obstacle detection. Monocular methods can be divided into four groups (Badrloo et al., 2022a): (a) appearance-based (Rane et al., 2020), (b) motion-based (Tsai et al., 2018), (c) depth-based (Hatch et al., 2021), and (d) expansion-based (Lee et al., 2021). Various studies have employed monocular methods for obstacle detection; initially, conventional image processing techniques were used for obstacle detection in these studies (Badrloo et al., 2022b; Liu, Li, Liu, et al., 2021; Padhy et al., 2019). Recently, studies applied artificial neural networks to real-time obstacle identification due to insufficient conventional image processing techniques for real-time applications (Hatch et al., 2021; Lambert et al., 2022; Shi et al., 2023). Among these four monocular techniques, appearance-based, depth-based, and expansion-based methods have been the focus of most of the research that has utilized artificial neural networks for obstacle detection. Since the goal of our proposed method is complete and real-time obstacle detection using monocular methods and artificial neural networks, we will now discuss the works that studied monocular methods using artificial neural networks. Appearance-based techniques regard the obstacle as a foreground object on a unified background (e.g., sky or road). These methods require prior background information, such as edge features (Liu, Li, Liu, et al., 2021), color (Shih An et al., 2019), texture (Mashaly et al., 2016), or shape (Liu, Li, Luo, et al., 2021). Detection of obstacles is performed on a single image. It is decided whether the image pixel matches the sky or the ground. If not, it is classified as an obstacle pixel. Also, this procedure is executed on each image pixel. The outcome is a binary image with white pixels representing obstacles and black ones representing others (Badrloo et al., 2022a). For example, using TensorFlow (Abadi et al., 2016) and OpenCV (Bradski, 2000), Talele et al. (2019) detected pixels distinct from the ground as obstacles. Similarly, Rane et al. (2020) uncovered obstacles using TensorFlow. Qiu et al. (2020) have employed You Only Look Once v3 (YOLOv3) (Redmon & Farhadi, 2018) and deep Simple Online and Real-time Tracking to identify and monitor dynamic obstacles. Moreover, Liu, Li, Luo, et al. (2021) created a fast feature integration strategy to enhance obstacle detection in hazy conditions. Liu, Li, Liu, et al. (2021) have introduced a new technique for semantic segmentation (Kirillov et al., 2019) that finds obstacles in the water in real-time. In another study, Du et al. (2022) introduced the YOLO model for locating and classifying traffic obstacles. On the basis of the improved YOLOv3 (YOLOv-4L) algorithm, Wang et al. (2022) also proposed intelligent obstacle detection for unmanned electric locomotives. He et al. (2022) suggested ME Mask R-CNN to enhance the precision of rail transit obstacle detection. Kumar et al. (2022) created an integrated intelligent central processing of the YOLOv3 neural network to detect obstacles and visualize traffic signs from the input image of a moving vehicle camera. In general, appearance-based approaches are applied to cases where the obstacle is easily distinguishable from the background. However, this supposition is invalidated in complicated environments with diverse objects, like, buildings, trees, and people (Zeng et al., 2016). Although deep-learning neural networks resolved this issue by identifying the object and differentiating them with semantic labels, accurate obstacle detection is currently dependent on the variety and quantity of training data required by these networks (Lee et al., 2021). In addition, these methods do not estimate the distance to the obstacle.

Also, in recent years, several researchers have used artificial neural networks and deep learning to detect obstacles with better accuracy (Haseeb et al., 2018; Hatch et al., 2021; Urban & Caplier, 2021). Initially, once trained, the network can build a depth map from one image (Lee et al., 2020). Then, similar to a conventional method (Häne et al., 2017), pixels with a depth less than a certain threshold are detected as obstacles. Kumar et al. (2018) used a Convolutional Neural Network (CNN) and four fisheye cameras to estimate the depth around the automobile. Mancini et al. (2018) have introduced a new CNN structure called Joint Monocular Obstacle Detection (J-MOD^2). Then, they used it to estimate the depth and detect the obstacle. In addition, Haseeb et al. (2018) introduced DisNet, a multi-hidden-layer neural network distance estimator. Hatch et al. (2021) demonstrated a system for collision avoidance with obstacles in another study. This system is effective due to the combination of an emergency policy, a network for collision prediction, and a high-level control network. Moreover, Urban and Caplier (2021) built a subsystem for visually impaired person's navigation that uses CNN to estimate the location and distance of the obstacle from the pedestrian. Even though depth estimation based on a single image and deep learning has been widely investigated and improved over recent years, it still has serious limitations that need to be overcome. One of these limitations is improving accuracy; researchers deepen the neural network for this goal, increasing memory usage and computational complexity (Ming et al., 2021). Also, depth estimation based on deep-learning techniques always uses several networks, which increases the processing and memory requirements. The second limitation is the availability of numerous, high-quality, and various educational data (Ming et al., 2021). More training data improve the results of these approaches. Therefore, the algorithm learns the objects' properties during training and can reveal them as a result.

Consequently, the need for radiometrically high-quality training data is the main issue of these approaches (Lee et al., 2021).

Also, since we are faced with various and unknown locations in obstacle detection, providing suitable training data in these situations would be time-consuming and costly. In addition, improving depth calculation in complex environments to meet the requirements of actual applications is still a hard task (Ming et al., 2021). In general, depth-based approaches have a significant volume of calculations due to the preparation of three-dimensional (3D) information from the surrounding environment (Silva et al., 2020). Moreover, due to the limitation on heavy processing and the requirement for a powerful GPU, this problem is more severe in MAVs (Pestana et al., 2019). In obstacle detection, however, there is no need to generate a depth map from the surrounding places; it is sufficient to identify the location of obstacles without creating a depth map.

In addition to the above research, expansion-based methods use the expansion rate of the object between sequential photos, which is the same principle that is used to detect obstacles. We know that as an object approach grows in size; consequently, various expansion criteria such as points (Aguilar et al., 2017), distances (Padhy et al., 2019), or regions (Badrloo et al., 2022b) can be used to estimate the expansion rate of an object in sequential images (Badrloo et al., 2022a). In expansion-based techniques, an object is regarded as an obstacle if its expansion rate exceeds a certain threshold. For example, Lee et al. (2021) trained the Faster Region-based CNN to recognize tree trunks for Unmanned Aerial Vehicle (UAV) navigation. Then, the obstacle trees are detected by calculating the tree image height to image height ratio. In addition, the image width of the trees was employed to locate paths free of obstacles. However, their research is limited to detecting tree trunks and does not provide a complete or accurate representation of the obstacle (Lee et al., 2021; Zeng et al., 2016).

Expansion-based approaches outperform the two other techniques by recognizing obstacles without the requirement to generate a depth map of the surrounding areas. In addition, they employ Speeded Up Robust Features (Mori & Scherer, 2013), Scale Invariant Feature Transform point scale (Aguilar et al., 2017; Al-Kaff et al., 2017), and distance ratio (Padhy et al., 2019), therefore, they are used in scenarios with a variety of objects. However, if the texture is smooth and featureless, it is challenging to extract the necessary features; hence, the identified obstacle points may be insufficient or include gaps that result in obstacle-detection failure.

Badrloo et al. (2022b) have used the extraction of image regions and their rate of area change to detect obstacles. Due to the use of traditional image processing algorithms for segmentation, the research mentioned above requires an appropriate threshold for each image. Additionally, at least three noncollinear match points are required in each region to calculate the area. Moreover, extracting and analyzing each image region in images with various objects is time-consuming. Furthermore, due to the high processing time and the necessity for a large number of match points, the method mentioned above is impractical for real-time applications. Yet, it has a high degree of completeness and accuracy in detecting obstacles.

To improve the speed and accuracy of our previous work (i.e., Badrloo et al., 2022b), we present a new method in which a panoptic deep-learning segmentation is employed to define object regions. Indeed, we use deep learning for object identification and, like, our previous paper, the object area expansion rate for accurate obstacle detection. This approach potentially streamlines the obstacle-detection process by reducing the need for manual threshold adjustments and enhancing the interpretation of obstacles as distinct entities. In addition to increasing the speed, this could improve handling complex scenes with respect to other expansion-based approaches that may employ feature extraction, image segmentation, and depth estimation for obstacle detection. The technique could be versatile and less sensitive to noise and variations in image quality. It also has more potential for automatic learning and adaptation to different environmental conditions. Section 3 describes our proposed obstacle-detecting mechanism.

3 | METHODOLOGY FRAMEWORK

This section explains the general framework of the proposed technique (Figure 1). This structure includes three parts: (1) data acquisition and preparation, (2) object extraction and point matching, and (3) obstacle detection. The outcomes of applying various steps of the suggested method to one of the data are displayed in Figure 2.

3.1 | Data acquisition and preparation

This step includes calibrating the fisheye camera in the lab and taking images. Images captured by a fisheye camera are highly distorted, which distortion increases with distance from the image's center, particularly at the image's edges; thus, this distortion decreases the accuracy of the measurements. Therefore, before performing computations, determining and applying the calibration parameters to the image coordinates is essential (Zhou et al., 2022). In this study, calibration is accomplished once in the lab using a set of chessboard images. These images were captured from various positions, angles, and distances from the chessboard. Figure 3 displays an example of images.

Several models and techniques for calibrating fisheye images have been presented (Huang et al., 2022; Ji et al., 2020; Scaramuzza, 2014; Urban et al., 2015; Xue et al., 2019). Urban et al. (2015) improved the method of Scaramuzza (2014) and presented a novel method for fisheye image calibration with an accuracy of less than one pixel that is able to model a fisheye camera with a 195° viewing angle (Urban et al., 2015). As a result, this model has been incorporated into the proposed methodology. This equation computes camera parameters:

$$\lambda \cdot g(m) = \lambda \cdot (u, v, f(u, v))^T = \lambda \cdot (u, v, f(\rho))^T = X_c, \quad \lambda > 0, \tag{1}$$

where u and v are the point's image coordinates, X_c are the point's ground coordinates, and λ is the scale parameter. $\rho = \sqrt{u^2 + v^2}$ represents the radial Euclidean distance between the point and the center of the image. The projection is typically modeled by the function f, which depends on the system lens. Urban et al. (2015)

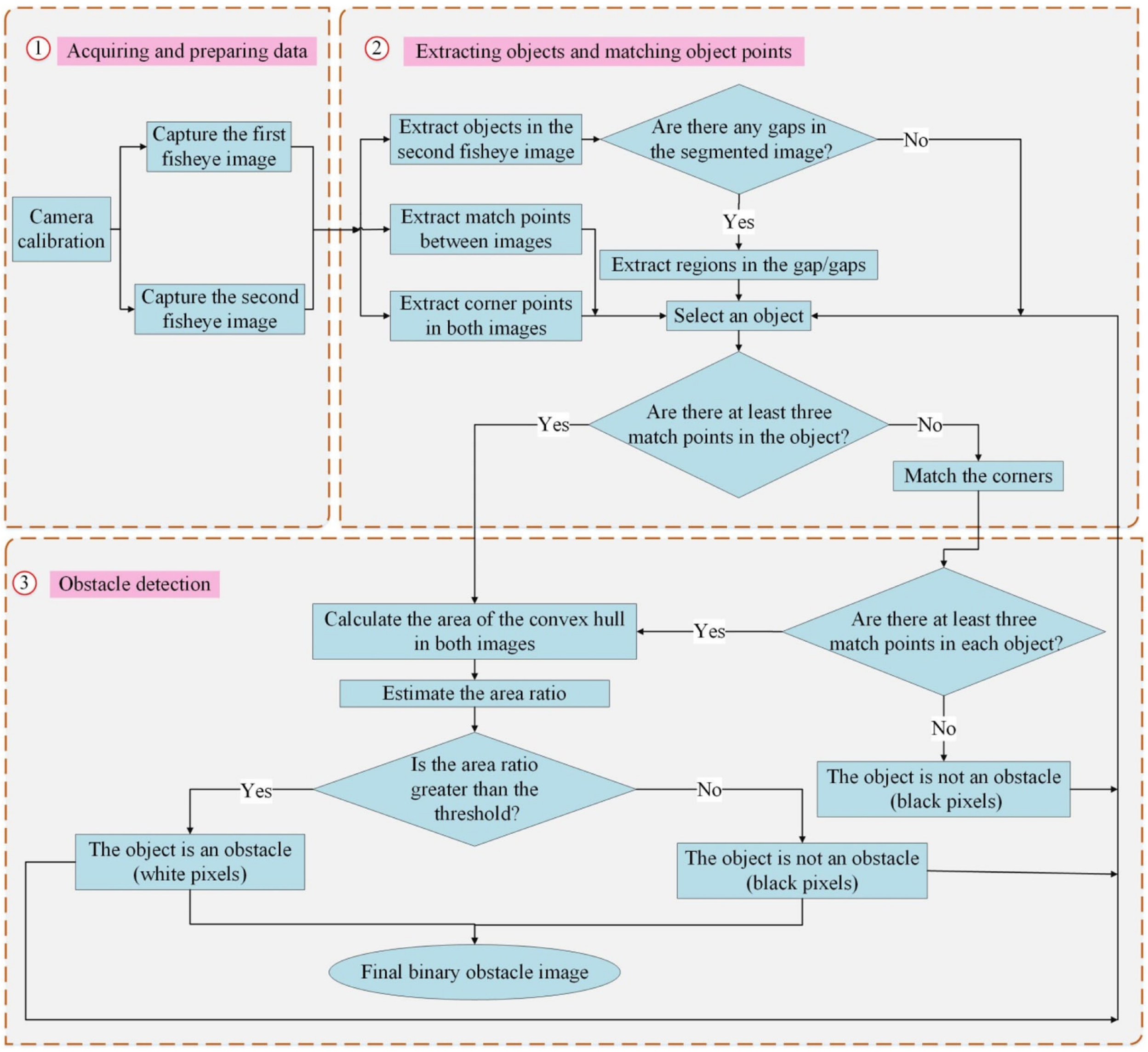
① Acquiring and preparing data
Camera calibration
Capture the first fisheye image
Capture the second fisheye image
② Extracting objects and matching object points
Extract objects in the second fisheye image
Are there any gaps in the segmented image?
No
Yes
Extract regions in the gap/gaps
Extract match points between images
Extract corner points in both images
Select an object
Are there at least three match points in the object?
Yes
No
Match the corners
③ Obstacle detection
Calculate the area of the convex hull in both images
Yes
Are there at least three match points in each object?
No
Estimate the area ratio
Is the area ratio greater than the threshold?
Yes
No
The object is not an obstacle (black pixels)
The object is an obstacle (white pixels)
The object is not an obstacle (black pixels)
Final binary obstacle image

FIGURE 1 The overall structure of the suggested technique. [Color figure can be viewed at wileyonlinelibrary.com]

approximate it with the Taylor expansion (Equation 2) and make it applicable to other lenses without requiring prior knowledge. The goal of calibration is to obtain coefficients $a_0, a_2, \ldots, a_n$.

$$f(\rho) = a_0 + a_2\rho^2 + \cdots + a_n\rho^n. \tag{2}$$

After calibration, sequential images are captured while MAV moves towards the object (Figure 2a,b). The spatial distance between the images is determined according to the drone's desired speed and the camera's frame rate.

3.2 | Object extraction and matching object points

In this phase of the proposed method, objects are extracted from the recent image (i.e., the second image). Also, the matched points in the first and second images are obtained. We implemented panoptic segmentation to extract objects (Kirillov et al., 2019). Because the proposed method aimed to complete obstacle detection, this algorithm extracts all countable and uncountable image objects (Li et al., 2020). Thus, it is now the most comprehensive segmentation approach (Kirillov et al., 2019; Li et al., 2020; Petrovai & Nedevschi, 2022). The panoptic segmentation is explained below.

Two deep-learning neural networks, instance segmentation and semantic segmentation, are merged in panoptic segmentation. Semantic segmentation allocates a label corresponding to a semantic class to each pixel, such as a car, house, tree, or similar object. There are two types of semantic class tags: (a) stuff (including uncountable nouns, such as road, sky, and sea) and (b) thing. Moreover, instance segmentation segments pixels with common labels, such as cars, individually and uniquely and assigns each one a unique identifier (Kirillov et al., 2019). Consequently, panoptic segmentation allocates

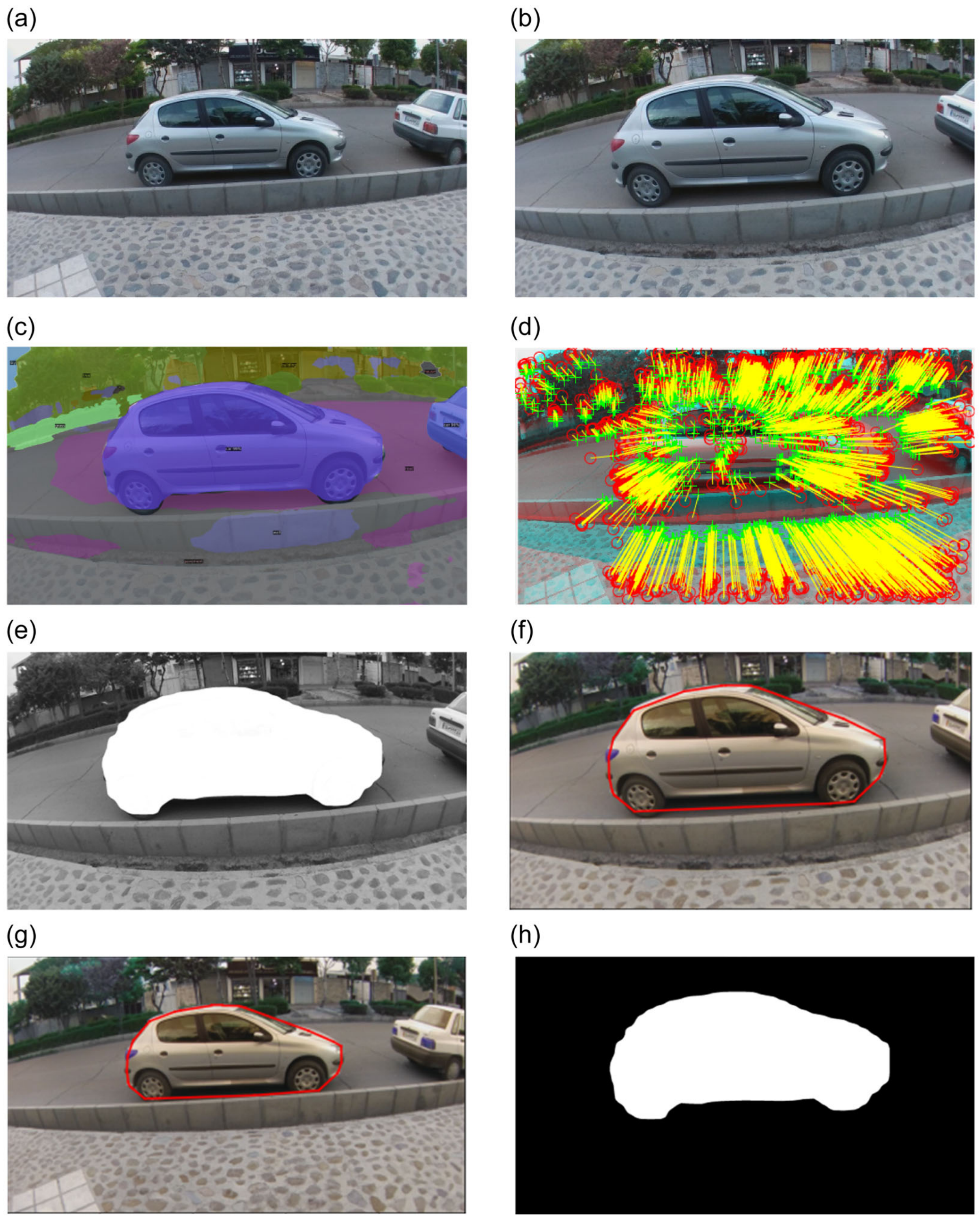

FIGURE 2 Process of obstacle detection. (a) The first image, (b) the second image, (c) objects that have been extracted from the second image, (d) matched points, (e) an object selected, (f) the object's convex hull in the second image, (g) the object's convex hull in the first image, and (h) the obstacle's binary image. [Color figure can be viewed at wileyonlinelibrary.com]

a semantic class label (l_i) and an instance identifier (z_i) to each pixel i of the image by combining both techniques. Therefore, objects are extracted entirely separately and without overlap in the panoptic segmentation method. l_i is a member of the set $l := \{0, \ldots, L-1\}$ with the number L of the semantic class. Also, the z_i identifier divides pixels belonging to a class into separate features. If the pixel class label is of type stuff, then the class will be uncountable, and the entire class will be assigned a z_i ID. In contrast, distinct pixels of a class-labeled thing will have different z_i identifiers due to countability (Kirillov et al., 2019).

For panoptic segmentation, several deep-learning networks have been introduced. These networks include UPSNet (Xiong et al., 2019), FPSNet (de Geus et al., 2020), EPSNet (Chang et al., 2020), VPSNet (Kim et al., 2020), DenseBox (Hou et al., 2020), Panoptic-deepLab (Cheng et al., 2020), and LPSNet (Hong et al., 2021). Also, various data sets are used to train panoptic segmentation, which includes: Cityscapes (Cordts et al., 2016) with traffic-related images containing 8 thing-type data and 11 stuff-type data, COCO (Lin et al., 2014) with 80 thing-type data and 91 stuff-type data, and Mapillary Vistas (Neuhold et al., 2017) with traffic-related images containing 37 thing

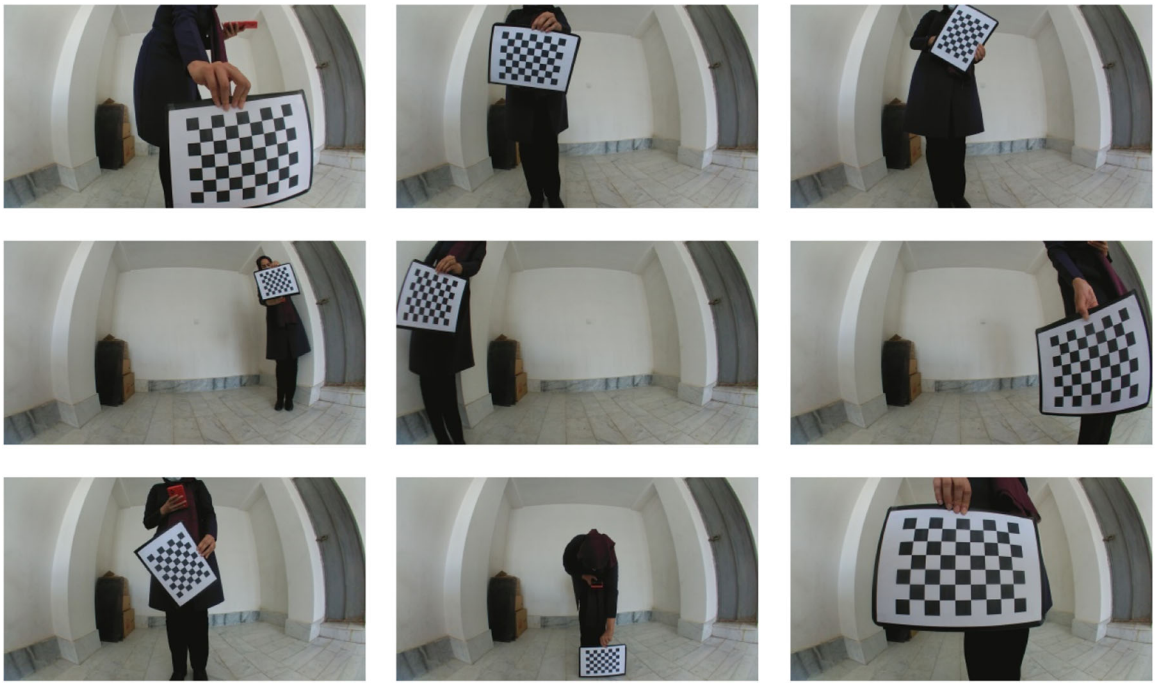

FIGURE 3 Example of images used to calibrate the fisheye image. [Color figure can be viewed at wileyonlinelibrary.com]

type data and 28 stuff type data. The proposed approach should have used one of these networks and data sets. The proposed obstacle-detection approach should be as precise and real-time as possible and able to detect numerous environmental objects. This approach requires using a precise and quick network that has been as fully trained as possible to detect various objects using the appropriate data set. In addition, COCO is one of the most commonly used and difficult data sets for implementing and evaluating panoptic segmentation outcomes in images with a variety of objects (Hong et al., 2021; Kirillov et al., 2019; Li et al., 2020; Petrovai & Nedevschi, 2022; Xiong et al., 2019). Therefore, we select COCO as our data set. Furthermore, Panoptic-deepLab networks (Cheng et al., 2020), FPSNet (de Geus et al., 2020), LPSNet (Hong et al., 2021), and PanoNet (Chen et al., 2020) are presented in recent research for fast panoptic segmentation. In addition to the speed parameter, from existing networks, Panoptic-deepLab networks (Cheng et al., 2020) employing COCO data have a better degree of precision (Elharrouss et al., 2021; Petrovai & Nedevschi, 2022). Therefore, the proposed method uses a Panoptic-deepLab network trained with COCO data to extract objects. The Panoptic-deepLab design is conceptually straightforward. So that it is composed of four parts (Cheng et al., 2020): (1) encoder backbone for use in semantic and instance segmentation, (2) decoupled ASPP components for semantic segmentation, (3) decoupled decoder components for instance segmentation, and (4) task-specific forecasting units. In addition, as shown in Figure 4, the Panoptic-deepLab network has a single-shot, bottom-up architecture.

After object extraction with the Panoptic-deepLab network trained on COCO data (Figure 2c), there will be pixels in the image that are not segmented and do not belong to any object, resulting in image gaps. Besides, the lack of variety in the training data for the network causes this issue. Therefore, similar to Badrloo et al. (2022b), we applied the SRG segmentation (Asmussen et al., 2015) method to segment the remaining pixels and fill the gaps.

In addition to extracting objects from the second image, we obtained the match points from the two images (Figure 2d). The matching algorithm must be able to extract sufficient matching points from highly distorted fisheye images. It should also be quick to accelerate the obstacle-detection process. For matching, we employed the Fast Affine Invariant (FAI) image-matching algorithm (Rodríguez et al., 2018). This algorithm is independent of the four affine transformation parameters; therefore, it can extract sufficient points from fisheye images despite severe distortion. Moreover, extracting and matching interest points in images is very fast. We first modified their coordinates to eliminate false match points using the calibration parameters determined in Section 3.1. Then, because it is insensitive to false match points, the fundamental matrix is estimated using the LMedS algorithm (Rusiecki, 2012). After that, we used the fundamental matrix to remove the false match points. When there are insufficient match points in an object (i.e., there are fewer than three noncollinear points), similar to Badrloo et al. (2022b) by applying the Shi-Tomasi algorithm (Mu & Li, 2018), the object is searched for more corner points in both images. Then, after the Least Square Matching algorithm (LSM) (Wang et al., 2017), matching match corners are added to the objects' match points. We hope that, as a result of this procedure, the object now has sufficient match points. Otherwise, the object is disregarded as nonobstacles.

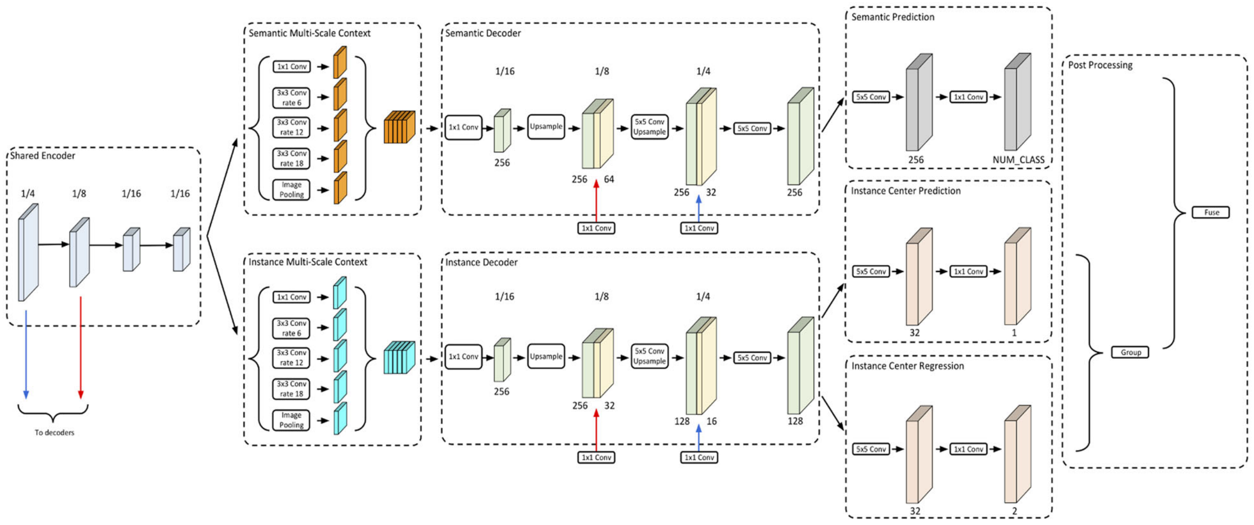

FIGURE 4 The framework of the Panoptic-deepLab network for performing panoptic segmentation (Cheng et al., 2020). [Color figure can be viewed at wileyonlinelibrary.com]

After extracting objects and their match points, we determined whether or not they represent obstacles in Section 3.3.

3.3 | Obstacle detection

Testing for obstacle detection is performed on any object containing at least three noncollinear match points (Figure 2e). To do it, in both images, the convex hull of the match points is created (Figure 2g,f), and then the convex hull area is calculated. According to Badrloo et al. (2022b), we used the convex hull area ratio to determine whether the object should be considered an obstacle (Equation 3). In Equation (3), $C_{2\text{area}}$ area and $C_{1\text{area}}$ area represent, respectively, the convex hull area of the object in the second and first images. If $\text{Ratio}(C_{\text{area}})$ exceeds a predefined threshold, the object is regarded as an obstacle (Figure 2h). Equation (3) is used to compute the threshold value (Badrloo et al., 2022b).

$$\text{Ratio}(C_{\text{area}}) = C_{2\text{area}}/C_{1\text{area}}, \tag{3}$$

$$\frac{C_{2\text{area}}}{C_{1\text{area}}} = \left(\frac{H+h}{H}\right)^2. \tag{4}$$

In Equation (4), h represents the distance between the two images, whereas H represents the distance of the second image from the object. Therefore, the minimum distance a UAV must maintain for obstacle collision avoidance is regarded as a reaction distance H_m, which is the smallest value of H. This indicates that the ratio of areas must be greater than or equal to the amount acquired by inserting $H = H_m$ into Equation (4).

The above procedure is repeated until every pixel in the second image has been identified as a nonobstacle or obstacle. The ultimate result is a binary image in which nonobstacle pixels appear black and obstacle pixels appear white.

4 | EXPERIMENTAL RESULTS AND DISCUSSION

We analyzed the performance of the presented algorithm in this section. In the beginning, sequential images were captured with the fisheye camera. These images have been taken in the frontal and lateral directions in an environment containing multiple experimental objects. In each test, a binary image was generated with obstacles represented by white pixels and nonobstacles by black pixels, and then the results were evaluated. In addition, the amount of time spent on each test was investigated.

Moreover, the outcomes are compared with those of Badrloo et al. (2022b). The assessment was carried out by using the following equations (Hong & Oh, 2021):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\%, \tag{5}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\%, \tag{6}$$

$$\text{Overall accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \times 100\%. \tag{7}$$

The variables in the above equations are defined as follows:

TP: The number of pixels that are obstacles and are detected as obstacles.

FP: The number of pixels that are not obstacles but are detected as obstacles.

FN: The number of pixels that are obstacles but are detected as nonobstacles.

TN: The number of pixels that are not obstacles and are detected as nonobstacles.

A low recall number indicates that obstacles have not been fully detected and prevent the UAV's safe navigation. The precision calculates the accuracy of the approach in nonobstacles detection. A low value for the precision indicates that the nonobstacles are mistakenly detected as obstacles (Badrloo et al., 2022b).

4.1 | Data

We took 60 pairs of static images with the LG 360 CAM fisheye camera. The image dimensions and camera field of view were, respectively, 1260 × 2560 pixels with 206° field of view. Of these, 30 pairs were taken by moving the camera forward (Figure 5), and another 30 pairs were taken by moving the camera to the sides (Figure 6). To calculate the expansion of the objects between sequential images, the images were processed in pairs. To determine the distance between each pair of images, we assumed a UAV speed of 10 m/s and a capture rate of 30 frames/second. Consequently, the computed value for h is 33 cm. In addition, we estimated that the UAV's minimum response time is equal to 250 ms. Therefore, H_m is equal to 2.5 m. Therefore, objects should be regarded as obstacles if the convex hull area ratio is greater than 1.28. For greater safety, we used 1.20 as the threshold that defines an object as an obstacle. Figures 5 and 6 provide examples of images used in conjunction with their associated outputs for each step, as described in Section 3.

4.2 | Experimental findings

As described in Section 3.1, we calibrated the camera using the Urban et al. (2015) method, which has an accuracy of less than one pixel and can model a fisheye camera with a 195° viewing angle (Urban et al., 2015). Thus, the image range up to a 195° viewing angle was utilized. The method of Urban et al. (2015) performed calibration with

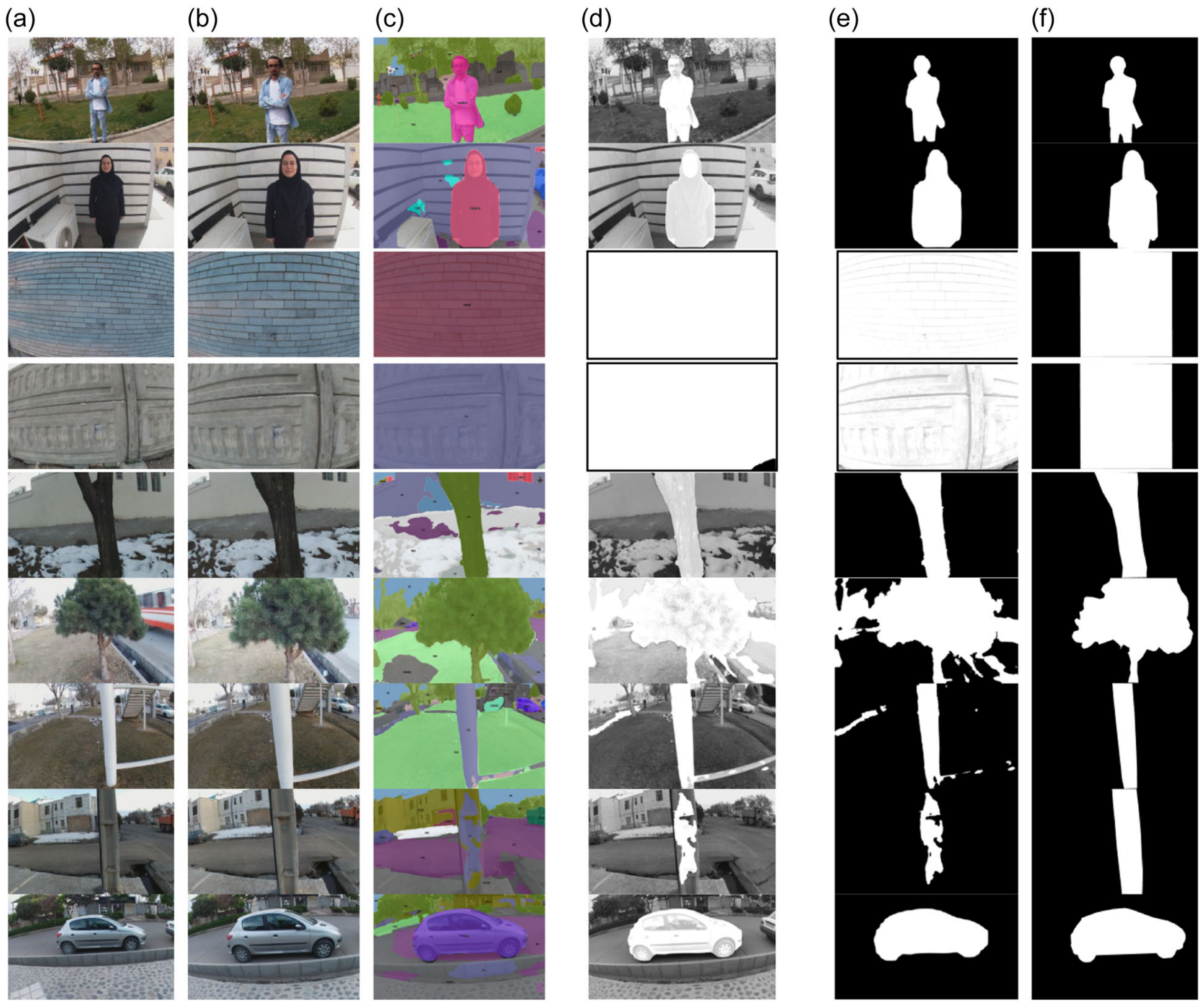

FIGURE 5 Obstacle detection in the forward movement. (a) The first image, (b) the second image, (c) objects extracted from the second image, (d) obstacles detected by our approach, (e) the final obstacle binary image, and (f) the actual obstacles binary image. [Color figure can be viewed at wileyonlinelibrary.com]

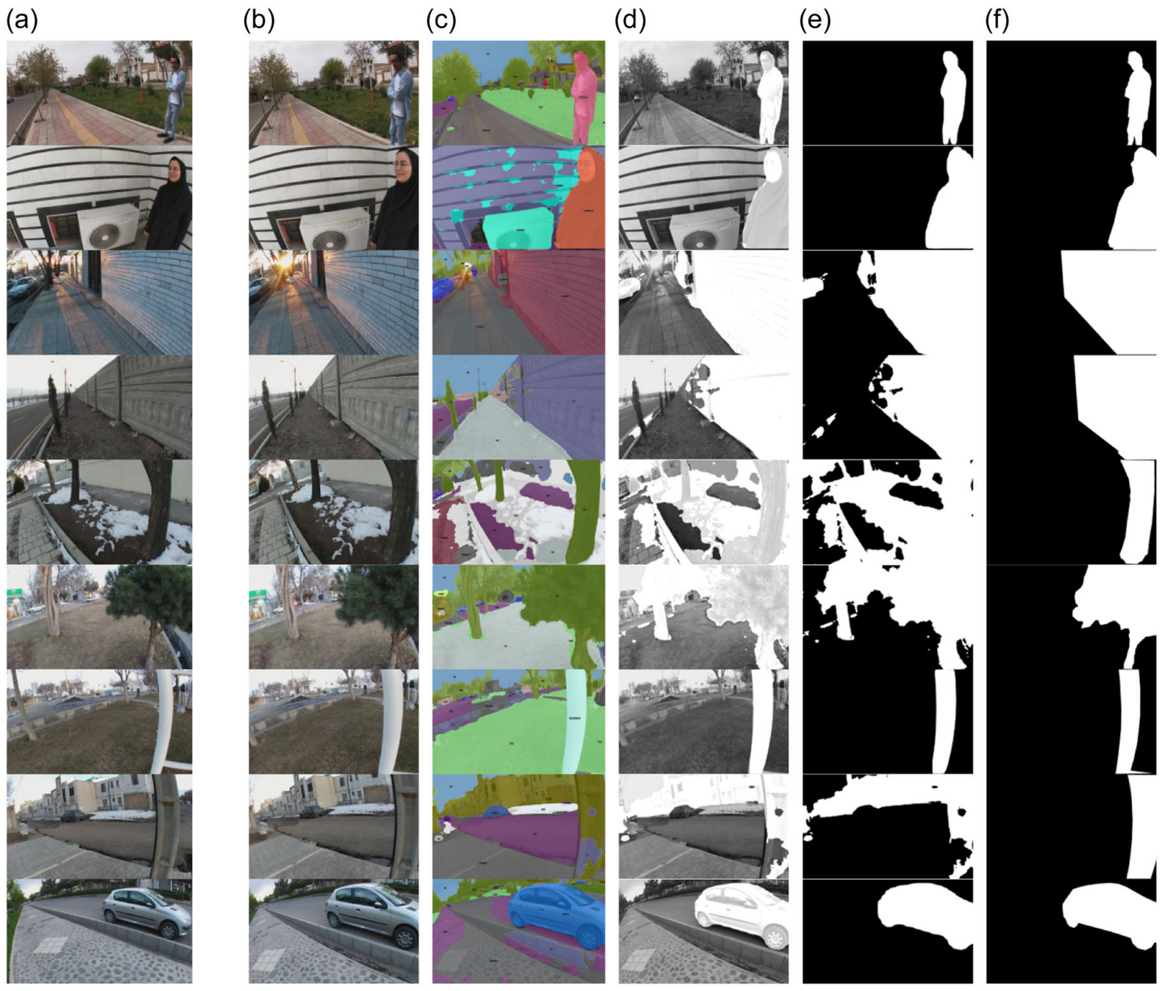

FIGURE 6 Obstacle detection in right movement. (a) The first image, (b) the second image, (c) objects extracted from the second image, (d) obstacles detected by our approach, (e) the final obstacle binary image, and (f) the actual binary obstacle image. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 1 Calculated calibration parameter values.

Calibration parameters	a_0	a_1	a_2	a_3	a_4
Values	−941.45672	100	0.0004287517777	−0.0000002060059	0.0000000001971

five optimal parameters of a_0, a_1, a_2, a_3, and a_4. We calibrated the fisheye lens with an average reprojection error of 0.53 pixels. Table 1 displays the calibration parameter values.

Then, using a Panoptic-deepLab network trained on COCO data, the objects were extracted. In addition, the SRG algorithm with a threshold value of 10 was used to segment the remaining pixels. To eliminate false match points, we selected 2000 points at random to compute the fundamental matrix parameters. In addition, the Shi-Tomasi algorithm was used to extract extra corner points. The match corner points were extracted utilizing the LSM algorithm with a threshold value of 0.5 and a 15×15 pixel window size. The final findings are displayed in Figures 5e and 6e, with nonobstacles indicated in black and obstacles marked in white.

Table 2 displays the findings of each step for forward and right movement. As shown in the first row of Table 3, the average number of objects searched for moving forward and to the right are 8 and 12, respectively. A large percentage of objects (between 84% and 88%) have at least three interest match points. Corner matching is used to determine the corresponding points for the remaining objects (16% and 9%). As seen in the fourth line of Table 2, there is no object without at least three corresponding points during forward movement. However, when moving to the right, due to the fisheye image's poor quality at the edges, sufficient match points could not be extracted, and 3% of the objects were regarded as nonobstacle.

At this step, it is necessary to evaluate the results and compare the detected obstacles to the actual obstacles. For this purpose, we

TABLE 2 The findings of this study.

Parameters	Forward motion data						Right motion data					
	People	Building	Tree	Pillar	Other obstacles	Total	People	Building	Tree	Pillar	Other obstacles	Total
The average number of extracted objects	9	2	9	10	9	**8**	12	9	12	13	12	**12**
Objects with at least three interest points (%)	93	50	85	100	93	**84**	87	83	93	95	80	**88**
Objects with at least three corner match points (%)	7	50	15	0	7	**16**	8	9	7	5	16	**9**
Objects that do not have three match points (%)	0	0	0	0	0	**0**	5	8	0	0	4	**3**

Note: Bold values indicate average results.

manually generated a binary image of obstacles (Figures 5f and 6f). Thus, pixels belonging to obstacles with a distance of less than 2.5 m were manually detected and highlighted with white in the binary image. The overall accuracy, precision, and recall percentages were then computed through a comparison of the overlap between the binary image generated by the proposed technique and the actual binary images of the obstacle. The average results evaluation and runtime are displayed in Table 3.

Recall accuracy achieved in the first row of Table 3 clearly shows that the presented technique successfully detects obstacles such as buildings and humans and other obstacles such as cars in two forward and rightward motion modes. Due to the training of the Panoptic-deepLab network to detect items, such as humans, buildings, and cars, these objects can be identified and used for obstacle detection with simplicity. However, compared with other data, the proposed algorithm performs poorly in detecting trees, pillars, and other obstacles, particularly when moving to the right. This problem is due to the poor training of the Panoptic-deepLab network to recognize trees and pillars, which results in their nonidentification. Therefore, when moving to the right, these objects are positioned at the image's edge, and due to the severe distortion of fisheye images at the image's edges, it becomes impossible to recognize these objects. Future research will be necessary to train the Panoptic-deepLab network further to recognize obstacles, such as trees and structures.

In addition, the average runtime of each algorithm step is provided in Table 4 to study algorithm runtime further. Note that a computer determined the algorithm's runtime in the Python environment with these details:

- Processor: Intel Core i7-8550U CPU @ 1.80 GHz.
- Memory: 12 GB.
- Graphics card: NVIDIA GeForce MX130.

In the first step, the camera is calibrated before flight. In addition, at the moment of flight, the latest captured image is the second image, and the image before it is the first; therefore, no time is spent selecting the sequential images. Therefore, the time required for the first stage, data acquisition and preparation, is deemed to be zero. The time presented in Table 4 corresponds to the second (object extraction and matching object points) and third (obstacle-detection) phases.

The fourth, fifth, sixth, and eighth rows in Table 4 demonstrate that the most time is spent on the second stage and significantly for interest and corner points matching. If faster matching algorithms can be used to match fisheye images, the time required by the algorithm for detecting obstacles will be drastically reduced. In addition, the runtime reduction of the parts of “object extraction” and “false match points elimination” can be considered in future research.

Additionally, to evaluate the method in different weather and lighting conditions, we grouped the images into windy, cloudy, rainy, snowy, and sunny subgroups. Some sample images are shown in Figure 7. while the results of this experiment are shown in Table 5. As can be seen, the best results are obtained in sunny conditions and the

TABLE 3 The suggested algorithm's results.

Motion direction	Parameters	Data					
		People	Building	Tree	Pillar	Other obstacles	Total
Forward	Recall (%)	98	99	92	69	95	**91**
	Precision (%)	97	49	87	93	98	**85**
	Overall accuracy (%)	99	49	93	96	98	**87**
	Time (s)	25	29	22	20	20	**7.71**
Rightward	Recall (%)	98	99	67	68	50	**76**
	Precision (%)	97	88	28	48	49	**62**
	Overall accuracy (%)	99	93	66	82	85	**85**
	Time (s)	21	17	16	19	17	**6.64**

Note: Bold values indicate average results.

TABLE 4 The average runtime of each proposed algorithm step.

Steps	Runtime in the forward direction for an average of eight objects (seconds)			Runtime in the right direction for an average of 12 objects (seconds)		
	Parts	Steps	Total	Parts	Steps	Total
First						
Data acquisition and preparation	0	0	**7.71**	0	0	**6.64**
Second						
Object extraction	1.20	7.458		1.17	6.259	
Region extraction in the gaps	0.60			0.48		
Image matching	2.00			1.946		
False match points elimination	1.10			0.65		
Corner points extraction	0.56			0.68		
Considering the presence of three match points in objects	0.488			0.733		
Corner points matching	1.50			0.60		
Third						
Considering the presence of three match points in objects	0.244	0.253		0.367	0.381	
Creating a convex hull in both images and calculating its area	0.008			0.012		
Obstacle image formation	0.001			0.002		

Note: Bold values indicate average results.

worst in rainy and snowy circumstances. This could be due to the quality of the point matching, which is usually degraded in such situations. Moreover, it was observed that even in the sunny images, when the object's surface was too shiny, the quality of the results was, once again, reduced.

In another experiment, we evaluated the results obtained using video frames (30 frames/second), taken with the LG 360 Cam. The camera movement speed was around 1 m/s. In this experiment, there were various obstacles, like, humans, buildings, cars, and trees. The video was recorded while the objects were approached from different directions, that is, from left to right and back to front. In total, 3108 frames with dimensions of 1536 × 1152 pixels were captured, from which 310 frames (one in 10) were selected for the evaluations. Using the selected frames (310 frames in total), a 3D model was constructed using AgiSoft's Metashape photogrammetric module (Figure 8). To scale the model, several scale bars with millimeter accuracy were used.

Then one in 10 frames was used to evaluate the quality of the obstacle detection. Overall, 31 pairs of frames were processed for obstacle detection, the result of which is shown in Table 6. First, the

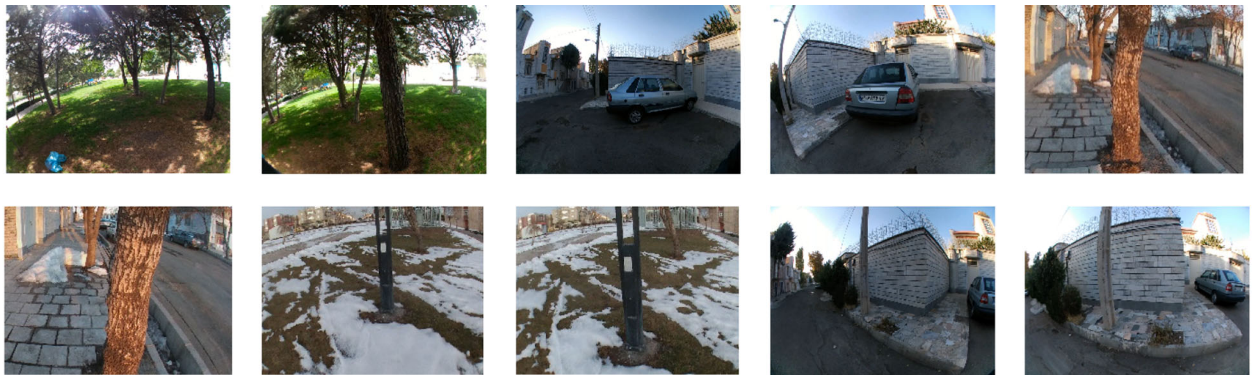

FIGURE 7 A sample of images taken in different weather and lighting conditions. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 5 The suggested algorithm's results for different weather and lighting conditions.

	Different weather and lighting conditions				
Parameters	Windy	Cloudy	Rainy	Snowy	Sunny
Recall (%)	30	83	17	18	95
Precision (%)	50	80	23	25	96
Overall accuracy (%)	90	78	89	89	93

frames were processed to obtain the figures in Table 6, and the corresponding obstacle binary images (see Figure 5e) were created. Then, for any identified obstacle, its distance to the corresponding camera was measured within the constructed 3D model. If the distance was compatible with that used by our algorithm, that object was considered a true sample. Otherwise, it was considered a false sample.

As can be seen, the overall accuracy of the algorithm ranges between 88% and 94% with an average of 92%, which is better than those obtained by our fisheye camera still images (see Table 3). This suggests that the camera movement has not caused much problem, perhaps because of its stability and small speed motion. Also, in this video experiment, the results were obtained mainly in day-light stable conditions, which, as shown in the previous experiment, was shown to produce the best results (see Table 5; the values under the “Sunny” column).

4.3 | Comparative evaluation

In this part, a comparison is performed between the proposed method and the algorithm presented by Badrloo et al. (2022b), which we have improved in our present study. Unlike previous algorithms (Aguilar et al., 2017; Lee et al., 2021; Mori & Scherer, 2013; Padhy et al., 2019), the method by Badrloo et al. (2022b) detects obstacle regions. The algorithm presented in this paper is partly based on that of Badrloo et al. (2022b). The main difference is the way the object regions are identified. In 2022, we used a region-growing algorithm, while in the current study, we employed the panoptic deep lab network to identify the objects.

To implement their method, Badrloo et al. (2022b) used the second image's regions extracted using the SRG algorithm with a gray-level threshold value of 10. Additionally, regions with fewer than 300 pixels were eliminated. Then, the closing morphological operation (Said et al., 2016) with a threshold value of 20 was used to fill the regions' gaps. In addition to extracting the regions, the Affine Scale Invariant Feature Transform (ASIFT) algorithm (Yu & Morel, 2011) was used to extract the match points from both the first and second frames. In the absence of at least three noncollinear match points in the regions, additional match points were extracted using corner matching. Therefore, matched corner points were extracted using the LSM algorithm (Wang et al., 2017) with a threshold value of 0.5 and a 15×15 pixel window size. Then, the area of the regions was calculated using the convex hull of match points. If the area ratio of the region is more than the threshold value of 1.20, that region is considered an obstacle. This process was carried out for all regions. In the end, a binary image of the obstacle was obtained in which the holes were closed using the closing morphological operation (Said et al., 2016) with a threshold value of 20. The findings of implementing the proposed method steps and the Badrloo et al. (2022b) technique are illustrated in Table 7.

The number of regions extracted by Badrloo et al. (2022b) is significantly greater than the number of objects extracted by the method suggested in this paper, as indicated in the first row of Table 8. Consequently, the proposed method will require fewer match points for obstacle detection. Therefore, a large number of the required match points (between 84% and 88%) are provided by the interest point matching algorithm, while a small number are obtained by corner matching or remain in the state without match points. Due to a large number of extracted regions in the Badrloo et al. (2022b) method, only 35% and 33% of the regions have match points extracted from the ASIFT algorithm.

Consequently, in our new approach, the need for match points to calculate the expansion rate has decreased by 49% and 55%. In addition, in the Badrloo et al. (2022b) method, the match points are extracted for 32% and 27% of the regions using corner matching. Also, 33% and 40% of the regions have no match points, so these are

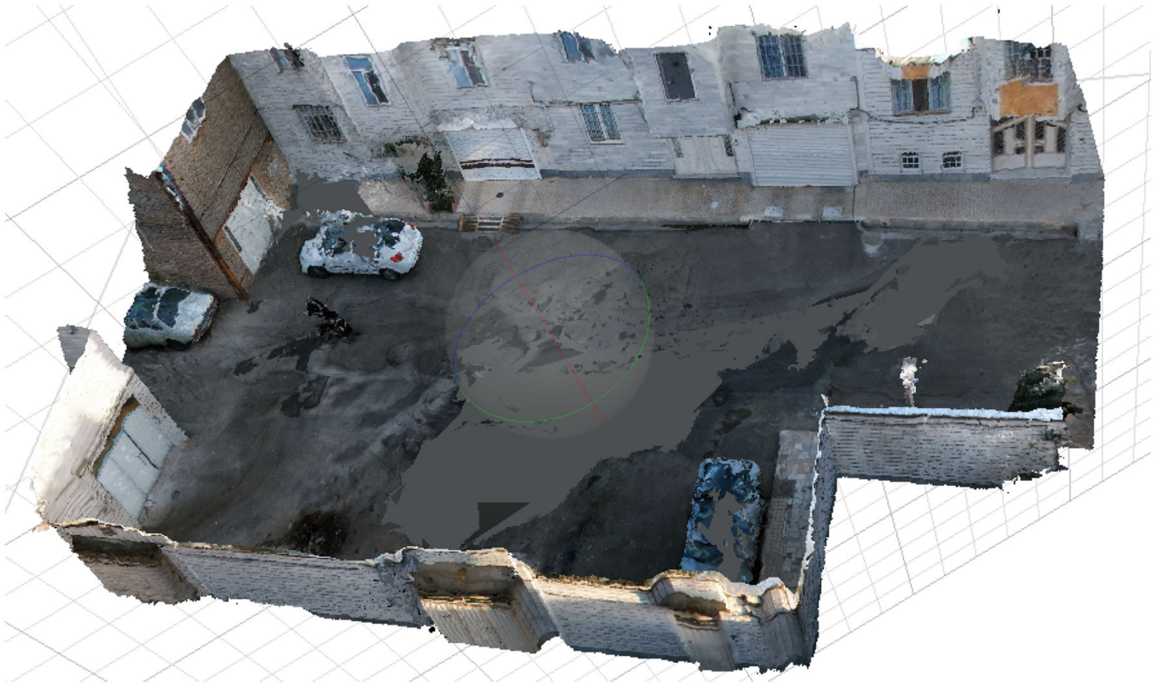

FIGURE 8 Model made of video frames. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 6 The suggested algorithm's results for video.

	Data					
Parameters	People	Building	Tree	Pillar	Other obstacles	Total
Recall (%)	99	97	95	90	97	**96**
Precision (%)	88	97	96	95	96	**94**
Overall accuracy (%)	93	94	93	88	93	**92**

Note: Bold values indicate average results.

not checked for obstacle detection. Also, Table 8 compares the assessment of the findings acquired from the proposed technique and the method of Badrloo et al. (2022b).

According to the first and fifth lines of Table 8, the proposed technique has improved the recall accuracy by 10% in forward motion and 18% in rightward motion compared with the Badrloo et al. (2022b) method. As a result of the complete extraction of objects in the proposed algorithm, recall, precision, and overall accuracy all have improved. However, in Badrloo et al. (2022b) setting the threshold value of the gray levels to a fixed value for the SRG algorithm may result in mixing the distance of nearby regions or the generation of numerous small regions. This, in turn, means the reduction of the accuracy of obstacle detection. Fortunately, in panoptic segmentation (Kirillov et al., 2019), obstacles are extracted accurately and without merging.

In the proposed algorithm, compared with the method of Badrloo et al. (2022b), just a limited number of objects are tested for obstacle detection (first row of Table 7). For example, it extracts a vehicle or a person as an object. However, in the method of Badrloo et al. (2022b), it is possible to extract multiple regions for humans or cars. As a result, the obstacle-detection time is significantly decreased. So, the suggested method decreases the time required to detect obstacles by 15.71 times during forward travel and 25.5 times during rightward movement. Table 9 shows the runtime of different steps.

As shown in Table 9, all steps have longer implementation times than those of the proposed method (see Table 5). Region extraction and image matching have the longest implementation times, respectively. Therefore, the proposed method in this paper has significantly reduced the implementation time by replacing these two components with (1) the extraction of objects using the Panoptic-deepLab method and (2) the employment of the FAI image-matching algorithm. It should be reminded that even though the remaining steps in Table 9 require a significant amount of implementation time, this time is impacted by a large number of extracted regions, so time is dependent on the number of regions.

Figure 9 also depicts the comparison of the results for several used image examples. According to Figure 9, it can be seen that the obstacles detected using the method of Badrloo et al. (2022b) have not been fully revealed, and in the majority of cases, far and near regions have been combined. In contrast, in the majority of instances, the proposed technique has presented a complete form of obstacles.

5 | LIMITATIONS

In Section 4.2, the results of two experiments were analyzed. In the initial analysis, the assessments were carried out in a forward motion. In the second analysis, assessments were carried out in a rightward

TABLE 7 The implementation findings of the proposed algorithm compared with those of Badrloo et al. (2022b).

Parameters	Proposed in this study		Badrloo et al.'s (2022b)	
	Forward motion data	Right motion data	Forward motion data	Right motion data
Average number of extracted objects or regions	**8**	**12**	**150**	**196**
Objects or regions with a minimum of three interest match points (%)	**84**	**88**	**35**	**33**
Objects or regions with a minimum of three corner match points (%)	**16**	**9**	**32**	**27**
Objects or regions without three match points (%)	**0**	**3**	**33**	**40**

Note: Bold values indicate average results.

TABLE 8 Results of the suggested algorithm compared with those of Badrloo et al. (2022b).

Motion direction	Parameters	The method proposed in this study	Badrloo et al.'s (2022b)
Forward	Recall (%)	**91**	**81**
	Precision (%)	**85**	**56**
	Overall accuracy (%)	**87**	**77**
	Time (s)	**7.71**	**121.10**
Rightward	Recall (%)	**76**	**58**
	Precision (%)	**62**	**66**
	Overall accuracy (%)	**85**	**82**
	Time (s)	**6.64**	**169.32**

Note: Bold values indicate average results.

motion. Then, the new method was compared with the existing algorithm (see Section 4.3). In the forward movement test (first test), all obstacles are identified in detail. Whereas the tree and column data sets had a lower average obstacle-detection accuracy than the other data sets because these obstacles were not properly extracted due to inadequate training of the Panoptic-deepLab network. In addition, the results of the second test demonstrated that the data sets of the tree, column, and other obstacles have a lower average recall accuracy than the other data sets. Due to the high distortion of the fisheye images at the edges, it is difficult to identify these objects using the Panoptic-deepLab network, leading to their low recall accuracy. Also, the missing sufficient match points in the image edges also reduces the accuracy of obstacle detection.

There are two options for improving the segmentation stage. First, train the segmentation network model with fisheye photos as input data to learn the semantic properties unique to fisheye images. The model could also incorporate fisheye augments to improve its resistance to distortion and perspective issues. This method, however, necessitates a huge data set of fisheye photos containing objects of various sorts. Another solution is to analyze the fisheye photos at multiple scales at the same time. This method can help capture finer details of objects in the image's center while also taking into account contextual information from the distorted edges. Provided that the implementation of this technique is not time-consuming, it may potentially improve segmentation results.

Although our aim in this study was to reduce the time for accurate and omnidirectional obstacle detection, and it has reduced the time significantly, it has not yet achieved the desired time. The high obstacle-detection time is considerably affected by the point-matching element. If faster methods are employed for the matching part in future studies, the obstacle-detection time will be significantly reduced. In addition, a comparison of our method with that of Badrloo et al. (2022b) revealed that our strategy required significantly less time to detect the obstacle in all experiments. So that object extraction reduces the time required for obstacle detection and the number of required match points. In addition, the suggested method has relatively good recall, precision, and overall accuracy due to the use of objects.

Overall, the goal of this paper has been to present a solution for fast, precise, and omnidirectional obstacle detection. The objects can be static or moving. The technique is relatively quick and efficient, requiring only a pair of fisheye images and a simple algorithm. It is mostly suited for small MAVs, which can only carry light loads and are vulnerable to approaching obstacles from all directions. Nevertheless, the presented approach can be used equally on larger vehicles and worked in indoor and outdoor environments. However, the results' accuracy depends on the segmentation quality and the matching process. These two, in turn, are dependent on the strength of the underlying deep-learning network and the matching algorithms, respectively. Also, the method does not work in real-time. Therefore, the technique may not be scalable in some practical applications, especially when the relative camera-to-object(s) speed is high. A solution could be to employ parallel processing units/algorithms to cope with such conditions.

Moreover, although increasing the speed of obstacle detection has been the main concern, we have tried to increase the accuracy of object detection in all directions, as much as possible. The procedure is precise and reliable, as it does not require depth information or stereo vision, which can be costly and complex operations. Provided that the surrounding objects carry appropriate texture, they can handle different obstacle sizes and shapes. An exception is the detection of very small or narrow objects. Indeed, the detection of

TABLE 9 The average runtime of each algorithm step.

	Runtime in the forward direction for an average of 150 regions (seconds)			Runtime in the right direction for an average of 196 regions (seconds)		
Steps	Parts	Steps	Total	Parts	Steps	Total
First						
Data acquisition and preparation	0	0	**121.10**	0	0	**169.32**
Second						
Region extraction	87.114	116.373		135.406	163.128	
Closing the holes	0.150			0.196		
Image matching	15.387			11.726		
False match points elimination	1.000			1.150		
Corner points extraction	0.570			0.680		
Considering the presence of three match points in regions	9.152			11.970		
Corner points matching	3.000			2.000		
Third						
Considering the presence of three match points in regions	4.575	4.727		5.994	6.192	
Creating a convex hull in both images and calculating its area	0.150			0.196		
Obstacle image formation and closing the holes	0.002			0.002		

Note: Bold values indicate average results.

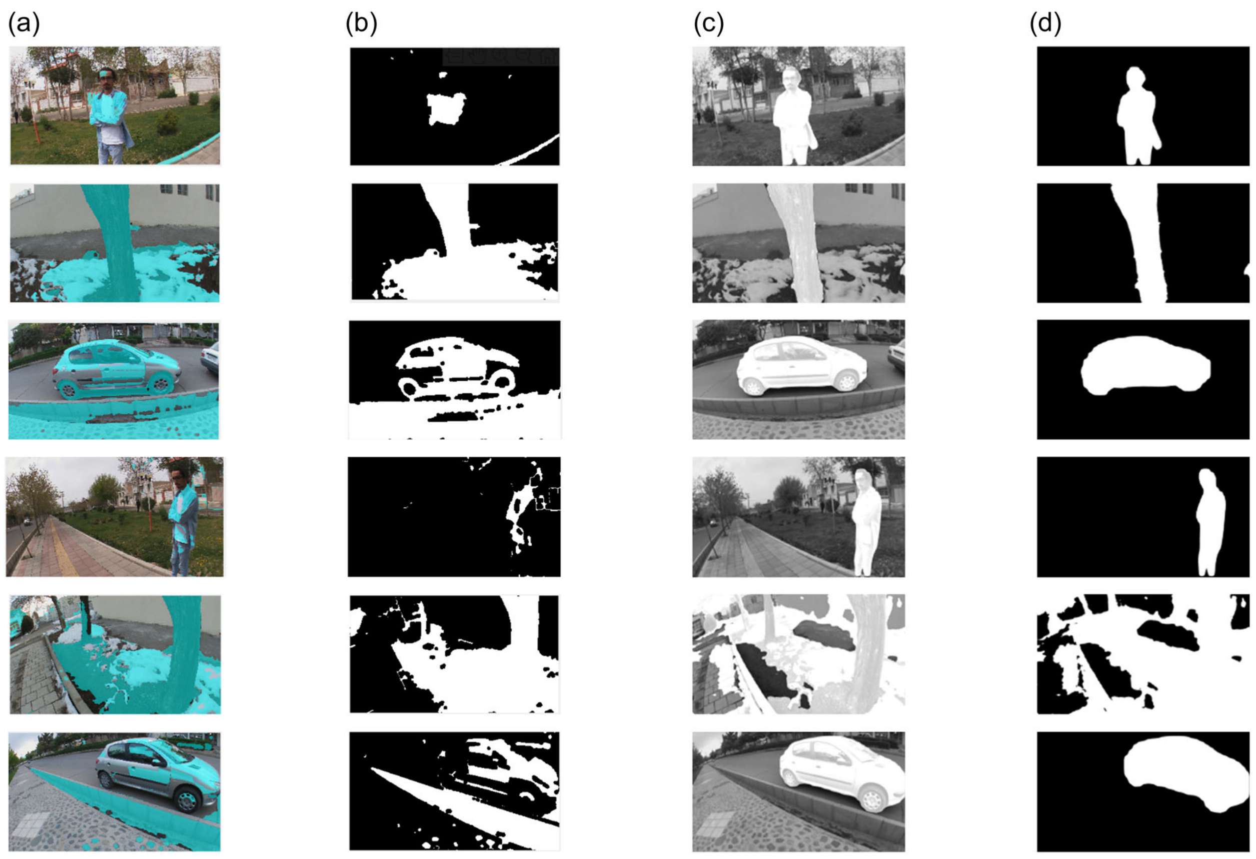

FIGURE 9 Our findings compare to those of an existing method (Badrloo et al., 2022b). (a) The existing method displays the results with cyan regions, (b) the binary image acquired by the existing method, (c) the obstacle detected by our method, and (d) the final binary image of the obstacle produced by our method. [Color figure can be viewed at wileyonlinelibrary.com]

very small and/or narrow objects is still an open problem and should be addressed in future research.

6 | CONCLUSION

Among the several approaches for detecting obstacles, expansion-based methods have simpler computations. However, they still face challenges in the speed and accuracy of obstacle detection. Recently, to enhance the accuracy and completeness of obstacle identification, a method (Badrloo et al., 2022b) has been introduced that uses the image regions and the match points within them to calculate the expansion rate between sequential images. The method mentioned above requires an appropriate threshold value to extract image regions and at least three noncollinear matching points for each image region. In addition, it takes time to evaluate a large number of regions in images containing multiple objects. Therefore, in the proposed algorithm, image objects are extracted. This is achieved by extracting objects from the latest image and matching points from both images. Then, using the match points, the objects' area is calculated in both images; if the area ratio of the object is more than the threshold value, it is termed an obstacle. Several tests with images taken from the front and sides were carried out to assess the proposed technique. The findings indicated that the proposed technique is between 76% and 91% accurate at detecting obstacles.

Moreover, in a separate experiment, our results were compared with a robust region-based obstacle-detection method presented by Badrloo et al. (2022b). Compared with our previous work (Badrloo et al., 2022b), the proposed technique improved recall accuracy by 10% for forward movement and 18% for right movement on average. Moreover, the proposed strategy has reduced the obstacle-detection time by 15.71 times at forward motion and 25.5 times at rightward motion.

High distortion in fisheye images at the edges makes it difficult to identify objects using the Panoptic-deepLab network, resulting in low recall accuracy. The missing match points in the image edges also reduce obstacle-detection accuracy. The network model can be trained with fisheye photos to improve segmentation and learn unique semantic properties. Incorporating fisheye augments can improve resistance to distortion and perspective issues. Finally, analyzing fisheye photos at multiple scales can capture finer details and contextual information simultaneously from distorted edges, and potentially improving segmentation results.

ACKNOWLEDGMENTS

The authors appreciate the Shaoxing Institute of Artificial Intelligence, China and K.N. Toosi University of Technology (KNT), Iran, for technical research support. This work is the outcome of a joint research study.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request. Data and code are available upon request.

ORCID

Saied Pirasteh http://orcid.org/0000-0002-3177-037X
Masood Varshosaz http://orcid.org/0000-0002-2703-1515

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J. et al. (2016) TensorFlow: a system for large-scale machine learning. In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, Savannah, GA, USA, 2–4 November, pp. 265–283.

Aguilar, W., Casaliglla, V. & Pólit, J. (2017) Obstacle avoidance based-visual navigation for micro aerial vehicles. *Electronics*, 6, 10.

Aharchi, M. & Kbir, M. (2022) *Localization and navigation system for blind persons using stereo vision and a GIS, WITS 2020*. Singapore: Springer, pp. 365–376.

Al-Kaff, A., García, F., Martín, D., De La Escalera, A. & Armingol, J. (2017) Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for UAVs. *Sensors*, 17, 1061.

Asmussen, P., Conrad, O., Günther, A., Kirsch, M. & Riller, U. (2015) Semi-automatic segmentation of petrographic thin section images using a "seeded-region growing algorithm" with an application to characterize wheathered subarkose sandstone. *Computers & Geosciences*, 83, 89–99.

Badrloo, S., Varshosaz, M., Pirasteh, S. & Li, J. (2022a) Image-based obstacle detection methods for the safe navigation of unmanned vehicles: A review. *Remote Sensing*, 14, 3824.

Badrloo, S., Varshosaz, M., Pirasteh, S. & Li, J. (2022b) A novel region-based expansion rate obstacle detection method for MAVs using a fisheye camera. *International Journal of Applied Earth Observation and Geoinformation*, 108, 102739.

Ball, D., Ross, P., English, A., Milani, P., Richards, D., Bate, A. et al. (2017) Farm workers of the future: vision-based robotics for broad-acre agriculture. *IEEE Robotics & Automation Magazine*, 24, 97–107.

Ball, D., Upcroft, B., Wyeth, G., Corke, P., English, A., Ross, P. et al. (2016) Vision-based obstacle detection and navigation for an agricultural robot. *Journal of Field Robotics*, 33, 1107–1130.

Barry, A.J., Florence, P.R. & Tedrake, R. (2018) High-speed autonomous obstacle avoidance with pushbroom stereo. *Journal of Field Robotics*, 35, 52–68.

Beul, M., Krombach, N., Nieuwenhuisen, M., Droeschel, D. & Behnke, S. (2017) Autonomous navigation in a warehouse with a cognitive micro aerial vehicle. In: Koubaa, A. (Ed.) *Robot Operating System (ROS)*. Cham: Springer, pp. 487–524.

Bharati, S.P., Wu, Y., Sui, Y., Padgett, C. & Wang, G. (2018) Real-time obstacle detection and tracking for sense-and-avoid mechanism in UAVs. *IEEE Transactions on Intelligent Vehicles*, 3, 185–197.

Bradski, G. (2000) The openCV library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 25, 120–123.

Chang, C.-Y., Chang, S.-E., Hsiao, P.-Y. & Fu, L.-C. (2020) EPSNet: efficient panoptic segmentation network with cross-layer attention fusion. In: Ishikawa, H., Liu, C.L., Pajdla, T. & Shi, J. (Eds.) *Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December*. Cham: Springer.

Chen, H.-C. (2019) Monocular vision-based obstacle detection and avoidance for a multicopter. *IEEE Access*, 7, 167869–167883.

Chen, X., Wang, J. & Hebert, M. (2020). *PanoNet: real-time panoptic segmentation through position-sensitive feature embedding*. arXiv preprint arXiv:2008.00192.

Cheng, B., Collins, M.D., Zhu, Y., Liu, T., Huang, T.S., Adam, H. et al. (2020). Panoptic-deeplab: a simple, strong, and fast baseline for bottom-up panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June, IEEE. pp. 12475–12485.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R. et al. (2016) The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June, pp. 3213–3223.

Dairi, A., Harrou, F., Senouci, M. & Sun, Y. (2018) Unsupervised obstacle detection in driving environments using deep-learning-based stereovision. *Robotics and Autonomous Systems*, 100, 287–301.

de Geus, D., Meletis, P. & Dubbelman, G. (2020) Fast panoptic segmentation network. *IEEE Robotics and Automation Letters*, 5, 1742–1749.

Du, L., Chen, X., Pei, Z., Zhang, D., Liu, B. & Chen, W. (2022) Improved real-time traffic obstacle detection and classification method applied in intelligent and connected vehicles in mixed traffic environment. *Journal of Advanced Transportation*, 1–12.

Elharrouss, O., Al-Maadeed, S., Subramanian, N., Ottakath, N., Almaadeed, N. & Himeur, Y. (2021) *Panoptic segmentation: a review*. *Arxiv*. [Preprint] Available from: https://doi.org/10.48550/arXiv.2111.10250

Escobar-Alvarez, H.D., Johnson, N., Hebble, T., Klingebiel, K., Quintero, S.A.P., Regenstein, J. et al. (2018) R-ADVANCE: rapid adaptive prediction for vision-based autonomous navigation, control, and evasion. *Journal of Field Robotics*, 35, 91–100.

Gayathri, R., Uma, V. & O'Brien, B. (2023) *Implementing robotic path planning after object detection in deterministic environments using deep learning techniques, machine learning, image processing, network security and data sciences*. Singapore: Springer, pp. 157–169.

Gharani, P. & Karimi, H.A. (2017) Context-aware obstacle detection for navigation by visually impaired. *Image and Vision Computing*, 64, 103–115.

Goodin, C., Carrillo, J., Monroe, J.G., Carruth, D.W. & Hudson, C.R. (2021) An analytic model for negative obstacle detection with lidar and numerical validation using physics-based simulation. *Sensors*, 21, 3211.

Grinberg, M. & Ruf, B. (2021) *UAV use case: real-time obstacle avoidance system for unmanned aerial vehicles based on stereo vision, towards ubiquitous low-power image processing platforms*. Cham: Springer, pp. 139–149.

Häne, C., Heng, L., Lee, G.H., Fraundorfer, F., Furgale, P., Sattler, T. et al. (2017) 3D visual perception for self-driving cars using a multicamera system: calibration, mapping, localization, and obstacle detection. *Image and Vision Computing*, 68, 14–27.

Haseeb, M.A., Guan, J., Ristic-Durrant, D. & Gräser, A. (2018) DisNet: a novel method for distance estimation from monocular camera. In: *10th Planning, Perception and Navigation for Intelligent Vehicles (PPNIV18), IROS*.

Hatch, K., Mern, J.M. & Kochenderfer, M.J. (2021) Obstacle avoidance using a monocular camera. In: AIAA Scitech 2021 Forum, 0269.

He, D., Qiu, Y., Miao, J., Zou, Z., Li, K., Ren, C. et al. (2022) Improved mask R-CNN for obstacle detection of rail transit. *Measurement*, 190, 110728.

Ho, H.W., De Wagter, C., Remes, B.D.W. & de Croon, G.C.H.E. (2018) Optical-flow based self-supervised learning of obstacle appearance applied to MAV landing. *Robotics and Autonomous Systems*, 100, 78–94.

Hong, W., Guo, Q., Zhang, W., Chen, J. & Chu, W. (2021) LPSNet: a lightweight solution for fast panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June, IEEE. pp. 16746–16754.

Hong, C.S. & Oh, T.G. (2021) TPR-TNR plot for confusion matrix. *Communications for Statistical Applications and Methods*, 28, 161–169.

Hou, R., Li, J., Bhargava, A., Raventos, A., Guizilini, V., Fang, C. et al. (2020) Real-time panoptic segmentation from dense detections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June, IEEE. pp. 8523–8532.

Hu, J.-w, Zheng, B.-y, Wang, C., Zhao, C.-h, Hou, X.-l, Pan, Q. et al. (2020) A survey on multi-sensor fusion based obstacle detection for intelligent ground vehicles in off-road environments. *Frontiers of Information Technology & Electronic Engineering*, 21, 675–692.

Huang, M., Wu, J., Zhiyong, P. & Zhao, X. (2022) High-precision calibration of wide-angle fisheye lens with radial distortion projection ellipse constraint (RDPEC). *Machine Vision and Applications*, 33, 44.

Ji, S., Qin, Z., Shan, J. & Lu, M. (2020) Panoramic SLAM from a multiple fisheye camera rig. *ISPRS Journal of Photogrammetry and Remote Sensing*, 159, 169–183.

Jiang, W., Song, C., Wang, H., Yu, M. & Yan, Y. (2023) Obstacle detection by autonomous vehicles: an adaptive neighborhood search radius clustering approach. *Machines*, 11, 54.

Kim, D., Woo, S., Lee, J.-Y. & Kweon, I.S. (2020) Video panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June, IEEE. pp. 9859–9868.

Kirillov, A., He, K., Girshick, R., Rother, C. & Dollár, P. (2019) Panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June, IEEE. pp. 9404–9413.

Kumar, V.R., Milz, S., Simon, M., Witt, C., Amende, K., Petzold, J. et al. (2018) Monocular fisheye camera depth estimation using semi-supervised sparse velodyne data. *Arxiv*. [Preprint] Available from: https://doi.org/10.48550/arXiv.1803.06192

Kumar, V.A., Raghuraman, M., Rashid, M., Hakak, S. & Reddy, P.K. (2022) Green-tech CAV: Next generation computing for traffic sign and obstacle detection in connected and autonomous vehicles. *IEEE Transactions on Green Communications and Networking*, 6, 1307–1315.

Lambert, R., Chavez-Galaviz, J., Li, J. & Mahmoudian, N. (2022) ROSEBUD: a deep fluvial segmentation dataset for monocular vision-based river navigation and obstacle avoidance. *Sensors*, 22, 4681.

LeCun, Y., Bengio, Y. & Hinton, G. (2015) Deep learning. *Nature*, 521, 436–444.

Lee, H., Ho, H. & Zhou, Y. (2021) Deep learning-based monocular obstacle avoidance for unmanned aerial vehicle navigation in tree plantations. *Journal of Intelligent & Robotic Systems*, 101, 1–18.

Lee, J., Jeong, J., Cho, J., Yoo, D., Lee, B. & Lee, B. (2020) Deep neural network for multi-depth hologram generation and its training strategy. *Optics Express*, 28, 27137–27154.

Li, Q., Qi, X. & Torr, P.H. (2020). Unifying training and inference for panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June, IEEE. pp. 13320–13328.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D. et al. (2014) Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B. & Tuytelaars, T. (Eds.) *European Conference on Computer Vision, 6–12 September*. Zurich, Switzerland: Springer, pp. 740–755.

Lin, J., Zhu, H. & Alonso-Mora, J. (2020) Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments. In: *2020 IEEE International Conference on Robotics and Automation (ICRA), 31 May–31 August*. Paris, France: IEEE, pp. 2682–2688.

Liu, J., Li, H., Liu, J., Xie, S. & Luo, J. (2021) Real-time monocular obstacle detection based on horizon line and saliency estimation for unmanned surface vehicles. *Mobile Networks and Applications*, 26, 1372–1385.

Liu, J., Li, H., Luo, J., Xie, S. & Sun, Y. (2021) Efficient obstacle detection based on prior estimation network and spatially constrained mixture model for unmanned surface vehicles. *Journal of Field Robotics*, 38, 212–228.

Mancini, M., Costante, G., Valigi, P. & Ciarfuglia, T.A. (2018) J-MOD 2: Joint monocular obstacle detection and depth estimation. *IEEE Robotics and Automation Letters*, 3, 1490–1497.

Mashaly, A.S., Wang, Y. & Liu, Q. (2016) Efficient sky segmentation approach for small UAV autonomous obstacles avoidance in cluttered environment. In: *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International, 10–15 July*. Beijing, China: IEEE, pp. 6710–6713.

Mendes, C.C.T., Osório, F.S. & Wolf, D.F. (2017) Real-time obstacle detection using range images: processing dynamically-sized sliding windows on a GPU. *Robotica*, 35, 85–100.

Ming, Y., Meng, X., Fan, C. & Yu, H. (2021) Deep learning for monocular depth estimation: A review. *Neurocomputing*, 438, 14–33.

Mori, T. & Scherer, S. (2013) First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In: *2013 IEEE International Conference on Robotics and Automation (ICRA), 10–13 December*. Bangkok, Thailand: IEEE, pp. 1750–1757.

Mu, Z. & Li, Z. (2018) A novel Shi-Tomasi corner detection algorithm based on progressive probabilistic hough transform. In: *2018 Chinese Automation Congress (CAC)*. IEEE, pp. 2918–2922.

Neuhold, G., Ollmann, T., Rota Bulo, S. & Kontschieder, P. (2017) The mapillary vistas dataset for semantic understanding of street scenes. In: *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October*, IEEE. pp. 4990–4999.

Opromolla, R. & Fasano, G. (2021) Visual-based obstacle detection and tracking, and conflict detection for small UAS sense and avoid. *Aerospace Science and Technology*, 119, 107167.

Padhy, R.P., Choudhury, S.K., Sa, P.K. & Bakshi, S. (2019) Obstacle avoidance for unmanned aerial vehicles: using visual features in unknown environments. *IEEE Consumer Electronics Magazine*, 8, 74–80.

Pehlivan, S., Unay, M. & Akan, A. (2019) Designing an obstacle detection and alerting system for visually impaired people on sidewalks. In: *2019 Medical Technologies Congress (TIPTEKNO)*. IEEE, pp. 1–4.

Pestana, J., Maurer, M., Muschick, D., Hofer, M. & Fraundorfer, F. (2019) Overview obstacle maps for obstacle-aware navigation of autonomous drones. *Journal of Field Robotics*, 36, 734–762.

Petrovai, A. & Nedevschi, S. (2022) Fast panoptic segmentation with soft attention embeddings. *Sensors*, 22, 783.

Qiu, Z., Zhao, N., Zhou, L., Wang, M., Yang, L., Fang, H. et al. (2020) Vision-based moving obstacle detection and tracking in paddy field using-improved yolov3 and deep SORT. *Sensors*, 20, 4082.

Rane, M., Patil, A. & Barse, B. (2020) Real object detection using TensorFlow. In: *ICCCE 2019*. Springer, pp. 39–45.

Redmon, J. & Farhadi, A. (2018) *Yolov3: an incremental improvement*. *Arxiv*. [Preprint] Available from: https://doi.org/10.48550/arXiv.1804.02767

Rodríguez, M., Delon, J. & Morel, J.-M. (2018) Fast affine invariant image matching. *Image Processing on Line*, 8, 251–281.

Rusiecki, A. (2012) Robust learning algorithm based on iterative least median of squares. *Neural Processing Letters*, 36, 145–160.

Said, K.A.M., Jambek, A.B. & Sulaiman, N. (2016) A study of image processing using morphological opening and closing processes. *International Journal of Control Theory and Applications*, 9, 15–21.

Scaramuzza, D. (2014) Omnidirectional camera. In: Ikeuchi, K., (Ed.) *Computer Vision: A Reference Guide*. Boston, MA: Springer US. pp. 552–560.

Shi, T.-W., Chang, G.-M., Qiang, J.-F., Ren, L. & Cui, W.-H. (2023) Brain computer interface system based on monocular vision and motor imagery for UAV indoor space target searching. *Biomedical Signal Processing and Control*, 79, 104114.

Shih An, L., Chou, L.-H., Chang, T.-H., Yang, C.-H. & Chang, Y.-C. (2019) Obstacle avoidance of mobile robot based on HyperOmni vision. *Sensors and Materials*, 31, 1021.

Shikishima, J., Urasaki, K. & Tasaki, T. (2022) PMOD-Net: point-cloud-map-based metric scale obstacle detection by using a monocular camera. *Advanced Robotics*, 37, 458–466.

Silva, A., Mendonça, R. & Santana, P. (2020) Monocular trail detection and tracking aided by visual SLAM for small unmanned aerial vehicles. *Journal of Intelligent & Robotic Systems*, 97, 531–551.

Sleaman, W.K., Hameed, A.A. & Jamil, A. (2023) Monocular vision with deep neural networks for autonomous mobile robots navigation. *Optik*, 272, 170162.

Sun, T., Pan, W., Wang, Y. & Liu, Y. (2022) Region of interest constrained negative obstacle detection and tracking with a stereo camera. *IEEE Sensors Journal*, 22, 3616–3625.

Talele, A., Patil, A. & Barse, B. (2019) Detection of real time objects using TensorFlow and OpenCV. *Asian Journal for Convergence in Technology (AJCT)*, 5(1), 1–4.

Tijmons, S., de Croon, G.C.H.E., Remes, B.D.W., De Wagter, C. & Mulder, M. (2017) Obstacle avoidance strategy using onboard stereo vision on a flapping wing MAV. *IEEE Transactions on Robotics*, 33, 858–874.

Tsai, C.-C., Chang, C.-W. & Tao, C.-W. (2018) Vision-based obstacle detection for mobile robot in outdoor environment. *Journal of Information Science & Engineering*, 34, 21–34.

Urban, D. & Caplier, A. (2021) Time- and resource-efficient time-to-collision forecasting for indoor pedestrian obstacles avoidance. *Journal of Imaging*, 7, 61.

Urban, S., Leitloff, J. & Hinz, S. (2015) Improved wide-angle, fisheye and omnidirectional camera calibration. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108, 72–79.

Vargas, J., Alsweiss, S., Toker, O., Razdan, R. & Santos, J. (2021) An overview of autonomous vehicles sensors and their vulnerability to weather conditions. *Sensors*, 21, 5397.

Wang, D., Ba, Y., Nian, L., Pei, D. & Zhang, J. (2017) Face detection based on color template and least square matching method. In: *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. IEEE, pp. 1–5.

Wang, W., Wang, S., Guo, Y. & Zhao, Y. (2022) Obstacle detection method of unmanned electric locomotive in coal mine based on YOLOv3-4L. *Journal of Electronic Imaging*, 31, 023032.

Wilshin, S., Amos, S. & Bomphrey, R.J. (2023) Seeing with sound; surface detection and avoidance by sensing self-generated noise. *International Journal of Micro Air Vehicles*, 15, 175682932211483.

Xiong, Y., Liao, R., Zhao, H., Hu, R., Bai, M., Yumer, E. et al. (2019) UPSNet: A unified panoptic segmentation network. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June*, IEEE. pp. 8818–8826.

Xue, Z., Xue, N., Xia, G.-S. & Shen, W. (2019) Learning to calibrate straight lines for fisheye image rectification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June*, IEEE. pp. 1643–1651.

Yu, Y., Liu, Y., Wang, J., Noguchi, N. & He, Y. (2023) Obstacle avoidance method based on double DQN for agricultural robots. *Computers and Electronics in Agriculture*, 204, 107546.

Yu, G. & Morel, J.-M. (2011) ASIFT: an algorithm for fully affine invariant comparison. *Image Processing on Line*, 1, 11–38.

Zeng, Y., Zhao, F., Wang, G., Zhang, L. & Xu, B. (2016) Brain-inspired obstacle detection based on the biological visual pathway. In: Ascoli, G., Hawrylycz, M., Ali, H., Khazanchi, D. & Shi, Y. (Eds.) *International Conference on Brain and Health Informatics, 13–16 October*. Omaha: Springer, pp. 355–364.

Zhang, Z., Cao, Y., Ding, M., Zhuang, L. & Tao, J. (2020) Monocular vision based obstacle avoidance trajectory planning for unmanned aerial vehicle. *Aerospace Science and Technology*, 106, 106199.

Zhang, J., Yang, X., Wang, W., Guan, J., Ding, L. & Lee, V.C.S. (2023) Automated guided vehicles and autonomous mobile robots for recognition and tracking in civil engineering. *Automation in Construction*, 146, 104699.

Zhou, G., Li, H., Song, R., Wang, Q., Xu, J. & Song, B. (2022) Orthorectification of fisheye image under equidistant projection model. *Remote Sensing*, 14, 4175.

How to cite this article: Pirasteh, S., Varshosaz, M., Badrloo, S. & Li, J. (2024) Developing an expansion-based obstacle detection using panoptic segmentation. *Journal of Field Robotics*, 1–20. https://doi.org/10.1002/rob.22319