Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
Practice article

Scene perception based visual navigation of mobile robot in indoor environment

T. Ran \textsuperscript{a}, L. Yuan \textsuperscript{a,b,∗}, J.b. Zhang \textsuperscript{a}

\textsuperscript{a} School of Mechanical Engineering, Xinjiang University, Urumqi, China

\textsuperscript{b} Beijing Advanced Innovation Center for Soft Matter Science and Engineering, Beijing University of Chemical Technology, Beijing, China

A R T I C L E  I N F O

Article history:
Received 24 June 2019
Received in revised form 28 July 2020
Accepted 4 October 2020
Available online 12 October 2020

Keywords:
Indoor mobile robot
Visual navigation
Convolution neural networks
Scene perception
Obstacle avoidance

A B S T R A C T

Only vision-based navigation is the key of cost reduction and widespread application of indoor mobile robot. Consider the unpredictable nature of artificial environments, deep learning techniques can be used to perform navigation with its strong ability to abstract image features. In this paper, we proposed a low-cost way of only vision-based perception to realize indoor mobile robot navigation, converting the problem of visual navigation to scene classification. Existing related research based on deep scene classification network has lower accuracy and brings more computational burden. Additionally, the navigation system has not yet been fully assessed in the previous work. Therefore, we designed a shallow convolutional neural network (CNN) with higher scene classification accuracy and efficiency to process images captured by a monocular camera. Besides, we proposed an adaptive weighted control (AWC) algorithm and combined with regular control (RC) to improve the robot’s motion performance. We demonstrated the capability and robustness of the proposed navigation method by performing extensive experiments in both static and dynamic unknown environments. The qualitative and quantitative results showed that the system performs better compared to previous related work in unknown environments.

© 2020 ISA. Published by Elsevier Ltd. All rights reserved.

1. Introduction

The ability of autonomous navigation in artificial environments which are hot places, such as medical scenes, offices, and airports is important for indoor mobile robots [1,2]. Indoor mobile robots can be used in indoor service applications especially during a major public health emergency, such as the recent outbreak of 2019 novel coronavirus (COVID-19) in the world. The deployment of such robots can to a great extent help humans perform dangerous and high-intensity work.

In the literature, part of the indoor navigation approaches mainly focused on path planning with prior knowledge like grid map, which is built by laser radar [3,4]. Although mobile robot can navigate in known environment, it will become unfamiliar when the global information is disorganized and lost which is similar to the outages of GPS for outdoor navigation [5]. Moreover, the artificial environment is usually changing in popular places both including stationary structural variations and dynamic objects appearing. Some other approaches addressed the problem of navigation under unknown environments, such as obstacle avoidance [6,7]. Expensive lidar along with other auxiliary sensors are usually combined for exploring of unknown conditions [8]. However, the high cost of navigation systems will hamper the spread of mobile robots in practical applications. Vision-based related research has become a hotspot in recent years with the characteristics of low-cost and the ability to obtain rich environmental information [9–13].

Most vision-based mapless navigation applied the main vision technique that includes optical flow, environment appearance feature detection and tracking from an image in earlier studies [14]. Some researchers extract optical flow to keep the robot traveling in the center of the corridor which emulates the bees’ flying behavior [15], and detect obstacles according to the optical flow intensity [16]. The main problem of this technique is that the environment around the robot need to be textured enough to optimize the optical flow computation. The combination of potential fields and appearance-based navigation method was proposed for vision based navigation and obstacle avoidance [17]. Appearance-based navigation also utilizes color vision to distinguish between the ground and obstacles [18]. Similarly, segmentation method was used for ground classification to detect obstacles [19–21]. Detection results were used as an input of the artificial potential field. Appearance-based environmental perception seems to work for vision-based local navigation. However, finding an appropriate environment representation and the corresponding
algorithm is the major challenge. The illumination change of the environment also makes the feature extraction algorithm not robust enough. Some researchers utilize rich visual geometric features for local navigation. Parallel guidelines were extracted from the bird’s eye views of the floor to guide the robot to follow the center of the corridor [22]. Guillaume et al. proposed an image-based nonholonomic constraint which guides the robot alone the reference visual route [23]. Lines in consecutive images from the smartphone are used for calculating vanishing points that detect heading change for the navigation system and the heading accuracy is related to the extraction precision of the vanishing point [24]. Similarly, the line segment matching method between the current acquired image and nearby reference images is presented for visual navigation in the structured environment without any specific localization [25]. Map-less navigation has been researched earlier, and some of them have been tested in real robots and proved to be of good performance of local navigation and obstacle avoidance. However, this kind of conventional methods is highly controversial with its low robustness and poor generalization capability. Some disadvantages such as the heavy computational load and the insecure algorithm (interference from the environment e.g. viewpoint, illumination, occlusion, noise) indicate that it may not be an optimal solution.

Recent years, there has been renewed interest in neural networks that widely used in the field of computer vision (such as image classification, object detection, and localization and semantic segmentation [26–29]). While some research has been carried out on this advanced technology, there are still few studies of neural network classifiers based visual navigation. To the best of my knowledge, LeCun applied end-to-end learning to mobile robot navigation as a pioneer in which the heading angle predicted for guiding in real time [30]. Subsequently, researches on visual navigation based on deep learning gradually increased [31–33]. Wei Chen et al. had realized door recognition with the application of a multilayer CNN [34]. Visual detection results could be used for robot navigation in an indoor environment. Giusti et al. applied a neural network classifier with four convolution layers for path following in forest creatively [35]. Although the deeper network of AlexNet that performs well is used for the classification of indoor scenes, the higher efficiency of the system is limited by the bulky structure of the net [36,37]. In these related studies, the authors did not carry out sufficient navigation performance evaluation, and in our research we found that the deeper network that previously used may not lead to higher scene classification accuracy. Consequently, improving both the classification accuracy and efficiency and generalization ability of the scene perception model presents a significant challenge for mapless local navigation.

In this paper, we consider an only vision-based navigation approach under unknown environments and location information is not required, which can be regarded as local navigation and deployed on mobile robots for service application (see Fig. 1). The characteristic of our approach is the design of the lightweight perception network in classifying the camera frames with both better accuracy and efficiency. Additionally, utilization of AWC algorithm makes the control of our mobile robot more robust. Simply put, we take advantage of the fact that CNN has satisfactory abstract ability of image features and that the shallow structure reaches higher accuracy when there is less outputs of the network. As a result, video frames from the camera that fix on the robot can be classified accurately and efficiently which guides the robot to travel in the strange environment safely. Consequently, we can reduce the layers of the perception network without compromising the classification accuracy and navigation performance.

Our main contributions are:

- The proposed method of our low-cost scene perception based local visual navigation for indoor mobile robot which does not need the location information during task execution.
- The design of a shallow CNN structure for scene perception which can reach higher classification accuracy and computation efficiency compared to previous related researches.
- The AWC algorithm we applied combined with our scene perception model has improved the local motion capability of the navigation system significantly compared to previous work.
- The qualitative and quantitative description of the comparison experiments on a real mobile robot in complex structured environments demonstrate the effectiveness of our proposed navigation method. Extensive evaluation under corridor environments and dynamic conditions shows the generalization ability of our navigation system.

The remaining part of the paper proceeds as follows: Section 2 illustrates the approach to collect data. Section 3 is concerned with the main methodology used in this study. Section 4 presents the findings of the research, focusing on the four key themes that the system architecture (both hardware and software), the performance of the network, navigation in a structured environment, and dynamic obstacle avoidance respectively. Finally, Section 5 provides brief conclusions. The open-source implementation, CNN models, and videos are available at https://github.com/rantengsky/Visual-Perception-based-Navigation.

2. The method of data acquisition

As inspired by the UAV trail following training [35], the viewpoint of our camera is adjustable by spinning a specific angle. The robot is controlled by human moves in our lab to collect three classes of images at the speed of 30 fps. A similar approach has also been proposed by [37], but the main advantage of ours is that the camera is responsible for both offline data collection and online navigation without additional sensors. As depicted in
Fig. 2. Definition of camera view.

Fig. 2, we assume that \( \vec{c} \) is the direction of the camera’s optical axis and define \( \vec{n} \) as the heading angle of the robot while moving (pedestal of a camera fixed in robot chassis and the camera is rotatable). We defined that the clockwise is positive.

\( \alpha \) is the angle between \( \vec{c} \) and \( \vec{n} \). Three classes of \( \alpha \) are defined as follows, which dovetail with three viewpoints of the camera when the robot move around the lab by an operator to collect images:

- Go Left (GL): \( 30^\circ < \alpha < 60^\circ \), available area is on the left and avoid obstacles on the right side;
- Go Straight (GS): \( -10^\circ < \alpha < 10^\circ \), there is no obstacles ahead and keep moving;
- Go Right (GR): \( -30^\circ < \alpha < -60^\circ \), available area is on the right and avoid obstacles on the left side.

Consider that there are too much rectangular in an indoor environment we give a more narrow range for the \( \alpha \) of going straight compared with the other two classes. That is to say, the robot is more decisive to navigate when place at rectangular instead of falling into a corner. The camera captures frames that contribute to the network as input data. After training, the real-time scene can be classified into one of these three class accurately.

3. Visual perception based indoor navigation

We carry out the task of scene perception using deep learning technology. In fact, there are plenty of challenges to be faced with this problem. On the one hand, lighting conditions, and the cluttered environment with kinds of chairs, desks and other things raise more difficulties. On the other hand, dynamic obstacles seem to be another significant challenge for the robot. In this section, navigation model, training set, network architecture and robot control will be described to solve these problems mentioned above.

3.1. Navigation model

The visual perception based navigation model is designed as a function approximator for (1) and (2), where velocity command \( u \) at time \( t \) can be described by:

\[
P_{GL}, P_{GS}, P_{GR} \sim D(\alpha_t, \mathbf{p}),
\]

\[
u \sim \pi_{AWC}(D(\alpha_t, \mathbf{p}), \mathbf{b}) \text{ if } |P_{GL} - P_{GR}| \in [A, B],
\]

\[
u \sim \pi_{RC}(D(\alpha_t, \mathbf{p}), \mathbf{b}) \text{ if } |P_{GL} - P_{CR}| \notin [A, B],
\]

Function \( D \) in (1) is the CNN-5 model used for scene perception which output three probabilities of \( P_{GL}, P_{GS} \) and \( P_{GR} \). \( \mathbf{p} \) is the perceptron model parameter, \( \alpha_t \) is the image of the current observation. Function \( \pi_{AWC} \) and \( \pi_{RC} \) in (2) are two control modes. \( \mathbf{b} \) is the baseline velocity that is manually set. Here, the velocity command \( u = (v, w) \) includes linear and angular velocity. The outputs of CNN-5 model are first used for control mode decision and then computing the specific velocity.

Fig. 3 gives a specific navigation model. As we can see, the model consists of perceptron, control mode decision and controller. The input to the navigation model is the robot’s current observation. Perceptron is responsible for image classification and generates three probabilities of \( P_{GL}, P_{GS} \) and \( P_{GR} \) respectively as inputs of the control mode decision. The value of the above probabilities correspond to three probable directions of robot moving (GL, GS and GR). CNN-5 model hyper-parameters are inherited from AlexNet [38] and refined in our training process to achieve higher classification accuracy which is described in network architecture subsection in detail. Three outputs of \( P_{GL}, P_{GS} \) and \( P_{GR} \) determine both the control mode and velocity command which provides a detailed description in robot control subsection. The linear and angular velocity are used for real-time robot control finally.

3.2. Training set

A great dataset is necessary to boost perception performance. Following the approach in Section 2, we use a joystick to control the robot moving along the path in our lab. The robot needs to avoid both static and dynamic obstacles. When collecting three classes of images, each category corresponds to a specific camera angle, and the sampling frequency keeps at 30 fps.

To enhance the robustness of illumination conditions, we expand dataset by capturing images covering three periods of morning, noon, and evening (Lighting conditions include natural light and incandescent light). Data stores depend on its category, that is to say when the camera is facing forward (\( -10^\circ < \alpha < 10^\circ \)), the images are labeled GS. Accordingly, when the camera is facing right forward (\( 30^\circ < \alpha < 60^\circ \)) and left forward (\( -30^\circ < \alpha < -60^\circ \)), the images are labeled GL and GR respectively. Table 1 shows clearly for the category. We have augmented our dataset by flipping that calls data augmentation [39]. Also, encoder recorded the odometer data for trajectory display.

We use one-hot encoding to label training (43,776 images) and testing (3678 images) sets and writes them as two TFRecord files in random order. This kind of file is generated by TensorFlow and can improve stability in training.

| \( \alpha \) range | Category |
|-------------------|----------|
| \( 30^\circ < \alpha < 60^\circ \) | GL       |
| \( -10^\circ < \alpha < 10^\circ \) | GS       |
| \( -30^\circ < \alpha < -60^\circ \) | GR       |


3.3. Network architectures

Prior to the work of [35–37], researches use relatively deep networks. However, the more layers will bring about the heavy
computing load. Therefore, we use the open-source framework Tensorflow [40] to design a shallow CNN architecture after the studying of AlexNet and LeNet [41]. As shown in Fig. 4, our network composes of three convolutional layers, two pooling layers, and two fully connected layers. All of the parameters set and adjust to improve training performance.

Feature maps of each layer visualize in Fig. 5. Raw images are resized to 64×64 as the input to feed the network. Every convolutional and pooling layer output 32 feature maps. The first fully connected layer is also the output layer with three values which are resized to 64×64 as the input to feed the network. Every layers, and two fully connected layers. All of the parameters set optimizes after each iteration. To avoid overfitting as much as possible, we use dropout noisy (4) in the first fully connected layer [42]. We also use the cuDNN accelerated library to improve GPU performance and take just about half an hour for training.

\[ x_{\text{norm}} = (x_{c.g.b} - \text{Min}) / (\text{Max} - \text{Min}) - 0.5, \]  

(3)

\[ \text{dropout noisy} \begin{cases} R_i^{(t)} & \sim \text{Bernoulli}(p) \\ \tilde{y}_i^{(t)} &= R_i^{(t)} \times y_i^{(t)} \\ z_i^{(t+1)} &= \tilde{w}_i^{(t+1)} \tilde{y}_i^{(t)} + \text{bias}_i^{(t+1)} \end{cases}, \]  

(4)

for (3), \( x_{c.g.b} \) is the color value in three channels; Max and Min is the extremum of the current image which generally set to 255 and 0. For (4), \( R_i^{(t)} \) is a random vector generated by the Bernoulli function; \( \tilde{y}_i^{(t)} \) is the value of a batch of neurons, \( z_i^{(t+1)} \) is a single neuron value with the operation of weights \( \tilde{w}_i^{(t+1)} \) and bias \( \text{bias}_i^{(t+1)} \).

There remain several vital components to discuss still. ReLU function (5) is used for nonlinearity activation, which can overcome the vanishing gradient problem and improve training efficiency [38]. Although stochastic gradient descent is applied to minimize the training loss in [35], it is easy to produce the severe concusion and get into the “saddle point”. The descent update so frequent that cannot reach the global optimum. We use the Adam optimizer (6) for cross entropy loss (7) optimization. This kind of optimizer can solve the problems of sparse gradients and very noisy based on adaptive estimates [43].

\[ R(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}, \]  

(5)

\[ \theta_t = \theta_{t-1} - \gamma \times \hat{m}_t / \left( \sqrt{\hat{v}_t} + \varepsilon \right), \]  

(6)

\[ L = \sum_{i=1}^{N} y_i \log \hat{y}_i + (1 - y_i) \log (1 - \hat{y}_i), \]  

(7)

for (6), \( \theta \) is the gradient parameter; \( \gamma \) is the learning rate; \( \hat{m}_t \) is the bias-corrected first-moment estimate; \( \hat{v}_t \) is the bias-corrected second raw moment estimate. \( \varepsilon \) is set to avoid the denominator from being zero. For (7), \( i \) is the sample size, \( \hat{y}_i \) is the actual label of sample \( i \), \( \hat{y}_i \) is the predicted label of sample \( i \).

3.4. Robot control

The model we have trained aims to classify a real-time image correctly. Three classes are corresponding to three motion directions. The output of the classifier is the probability of each class which is used to calculate the specific velocity to navigate the robot [17]. As mentioned above, the robot is equipped only with a monocular camera as the sensor. The blind area on the side of the robot will be potential danger. In some special cases, the robot could perceive the obstacles beside its body with a small probability of \( P_{GL} \) and \( P_{GR} \). Such a probability is not enough to drive the robot to keep away from the objects nearby. In view of this problem, AWC algorithm has applied to the navigation system.

The flow diagram of the robot control is shown in Fig. 6. We adopt two ways of RC and AWC algorithms to control the robot. The results of model classification are the input parameters for the control system. We define that \( P_{GL} \), \( P_{GS} \), and \( P_{GR} \) are the probability of GL, GS and GR respectively. The range of \([A, B]\) is given by experience. Subtraction of \( P_{GL} \) from \( P_{GR} \) decides to use the specific control algorithm. The full implementation of RC and AWC is described using pseudocode shown in Algorithm 1.

In the robot control algorithm, two if conditional statements are used to determine the control mode. The robot motion trend is estimated in step 2 and step 3 by \( P_{GL} \) and \( P_{GR} \). For AWC, amplification coefficients are defined in step 4 and step 5. These two parameters are used for lowering the forward speed and increasing turning speed. \( \lambda \) is the empirical value and we make it greater than 1. For RC, only the base velocity is employed as the following functions (8) and (9):

\[ V_{\text{linear}} = P_{GS} \cdot C_1, \]  

(8)
V_{angular} = (P_{GL} - P_{GR}) \cdot C_a, \quad (9)

the product of \( P_{GS} \) and \( C_1 \) (base linear speed) as the linear velocity; the product of \( (P_{GR} - P_{GL}) \) and \( C_a \) (base angular speed) as the angular velocity.

The user interface is shown in Fig. 7. The background is the real-time scene from the single camera of the robot. There are three classes of probability in the upper left corner, and which is to determine the trend of robot movement. Velocity command that calculated according to the functions above is shown in the lower left corner. When there is AWC mode, the interface will provide a system prompt.

4. Experiments

The desktop and mobile robot platform are responsible for model training and navigation deployment respectively. The CPU of our desktop is Intel Core i5-7500 and GPU is Nvidia GTX 1060. As shown in Fig. 8, a laptop is placed on the mobile robot platform and equipped with Intel core i5-3210M and Nvidia Geforce GT 610M. The CNN model that we have trained before is loaded in the laptop to classify the images captured by the monocular camera. The base linear and angular velocity is set to 0.2 m/s and 0.1 rad/s.

The experiment includes three parts: model performance, navigation in the structured environment, and the reaction to dynamic obstacles. The sites that we experiment at are our lab (9 m^2 8 m) and a long corridor (28 m^2 3 m) which are very similar to the medical scenario. Besides the original stuff in the lab, several artificial roadblocks are added by us. By doing this, the environment for navigation is different from that the data collection. Moreover, the scene of the corridor does not provide data for model training. As up to now, the single monocular camera is the sole sensor both for collecting training data and testing the algorithm. Odometer is used to record the trajectory (assuming that odometer data is the ground truth) and the actual speed recording. These data are used to analyze navigation performance. To describe more visually, we use a 2-D lidar applying gmapping algorithm [44] and build grid maps of the experimental sites. Maps will be shown later.

4.1. The system architecture

Fig. 8 shows the essential components of the mobile robot, equipped with a single camera, a laptop for image processing, STM32 microcontroller for motion control and motor drivers. Besides, the chassis is driven by couple of DC motors which are powered by 24 V DC battery. Monocular camera is installed right in front of the mobile robot that captures raw images from its view. Laptop equipped with GPU runs our deep neural network and produces velocity commands according to the classification results. The commands are addressed to STM32 microcontroller via USB port. STM32 is a core control unit which is responsible for parsing the velocity command and generating PWM signals to control motors via the motor drivers. The entire hardware platform is about 450 mm long, 420 mm wide (wheel interval), 300 mm high and weights 6.2 kg.

Fig. 9 shows the software framework of the navigation system. Our framework is developed based on Python3.6 which deploys in Window 10. Tensorflow [40], OpenCV2, and other third parties provide the full-fledged API for neural network training, data preprocessing, and online image processing. MySQL is responsible for receiving and storing the odometry data from the chassis. CPU is working on multithreaded mode to improve efficiency and GPU use the cuDNN library [45] to accelerate the computation.
4.2. Model performance

To ensure the model performance, we first trained our dataset using different networks, and then the outstanding one was compared with the related researches. The more comprehensive assessment helps us to choose a more appropriate model.

We use the other two different structure of networks to train our dataset. AlexNet (CNN-8) with eight layers that was used in [36] and [37] is one of them in which most of the parameters have been modified to match our data. LeNet (CNN-4) with parameters been modified is another network which is four layers less than AlexNet. Three networks use the same activation function of Relu, optimizer of Adam, dropout noise of 0.75 and dataset of ours. Training accuracy of three models is shown in Fig. 10 with 600 iteration steps. The training of our network and LeNet is more stable than AlexNet. Furthermore, our model accuracy is slightly more accurate than LeNet.

The details is described in Table 2. It also shows the highest accuracy of each model. Surprisingly, CNN-4 from LeNet with the most simple structure is larger than ours in model size. We found that the model size is decided by the output feature map size of the last convolution layer to a great extent. The classification time is closely related to the mode size. Our CNN-5 model has a smaller size than the other two and it is faster to be loaded and classify a single image. With such high performance, we can build a real-time navigation system.

We develop the navigation system not only deploys the model that have trained but also exploits multithreading techniques to read and store data. With the operations above, the system enabled efficient processing and could satisfy real-time navigation. We selected a period randomly which represented the time variation to process a frame of image (included image processing and classifying). As depicted in Fig. 11, the average time that required was just 53 ms. Some peaks occur regularly which caused by data storage termly through our analysis.

4.3. Navigation in structured environments

The following experiments were conducted using the real robot in our lab and a long straight corridor outside the lab. Environmental conditions of the lab for experiments were different from those in which data were collected for model training. For example, there were added more artificial roadblocks in the lab which had great changes. Furthermore, the corridor was completely alien for the robot, although it had no obstacles. The maps we had built will be shown with the robot trajectories.

The first experiment was carried out in the lab. The robot needs to traverse through narrow indoor environments. During the experiment, the robot starts at the beginning and goes
Fig. 12. Comparison of motion trajectories that deploy different perception models and control methods in the mobile robot, including: (a) CNN-8 model, (b) CNN-5 model without AWC algorithm, (c) CNN-5 model with AWC algorithm.

Fig. 13. Comparison of: (a) changes of linear velocity: navigation system without AWC algorithm (green) and with AWC algorithm (blue); (b) changes of angular velocity: navigation system without AWC algorithm (green) and with AWC algorithm (blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

through an area with several chairs on both sides. Then it enters a narrow curve and passes through an area with a lot of clutter. Finally, the robot goes out through a narrow “gate”. Snapshots of several key areas are posted next to the trajectory. Experiments were conducted using CNN-8 model, CNN-5 model without AWC and CNN-5 model with AWC respectively, and their performances were evaluated and compared. We repeated each set of experiments for four times that are shown
in Fig. 12. Fig. 12(a) shows the trajectories of the mobile robot deploying the CNN-8 model which was used in [36]. The robot collided after moving some distance because of the inaccurate scene classification in the bend, and it could not make it through the corner. Fig. 12(b) shows the trajectories of the mobile robot deploying the CNN-5 model. With this model, the robot could classify the camera frames more accurate and through the curve successfully. However, it still not robust enough that the robot was in a state of wag without collision sometimes when the probabilities of GL and GR are very close. Fig. 12(c) shows the trajectories of the mobile robot deploying the CNN-5 model with AWC algorithm. The AWC algorithm can help the robot make better decisions especially at the turning. Because the difference between the probability of GL and GR will be amplified appropriately to drive the robot more decisively. As we can see, the trajectories are longer and smoother with fewer inflection points compared to the first two methods, and which is conductive to human–machine friendly interaction. To further evaluate the performance of our method, we quantified the experimental results that presented in Table 3. The quantitative comparison shows that our approach has a longer traveling distance and time for the mobile robot. Meanwhile, classification speed of our CNN-5 model is faster than that deployed with CNN-8 model. So the part of the traveling time of the Y.H. Kim’s method is the consumption of scene classification rather than the effective time.

Additionally, Fig. 13 shows part of the velocity changes with and without AWC algorithm. As we can see, there is a slight trend of the velocity changes with AWC, and in contrast, navigating without AWC has more sharp variations both in linear and angular velocity. When adding the AWC algorithm, the variance of the overall linear and angular velocity change had decreased by 48.5% and 56.5% respectively as shown in Table 4. The reason why the proposed algorithm performs better is that the AWC has an adaptive process for velocity jump that comes from the change of scene.

The second experiment was carried out in the long straight corridor. Panoramagram of the corridor is shown in Fig. 14, and the left side is the beginning of the navigation. To test the performance navigating in the corridor, we arranged for the robot to enter the corridor from $-22^\circ$, $0^\circ$, $30^\circ$ (see Fig. 15). The robot successfully navigated in different initial angles which are presented in Fig. 16. The trajectory was sometimes undulating because of the characteristics of local planner and the uneven brightness of artificial illumination. Even so, the robot could navigate safely without collisions.

### 4.4. Navigation in unstructured environments

The last four experiments show the performance of the proposed method when meeting dynamic obstacles. The system computed real-time steering commands according to the location of the dynamic obstacles.
The first experiment was carried out to evaluate the reaction of the robot that was deployed with different algorithms when the distance between the robot and the obstacle changed. We consider people as dynamic obstacles. According to Fig. 17, our CNN-5 model with AWC algorithm has a smoother change in speed compared to the CNN-5 model without AWC algorithm. When the robot is deployed with the CNN-8 model, there is a wrong change in linear and angular speed which is caused by the incorrect scene classification. This situation is dangerous and unfriendly for autonomous navigation especially when the robot is very close to the obstacle. As we can see from the results, when the robot is more than about three meters away from the obstacle, it maintains the maximum linear speed and the minimum angular speed. When the distance is less than three meters, the robot actively reduces the linear velocity and increases the angular velocity to avoid the obstacle. When the distance is less than about one point five meters, the robot maintains the minimum linear speed and the maximum angular speed.

In the experiment, the distance between the robot and the obstacle ranges from four to zero meters. From the results, we know that when the obstacle is too far and more than about three meters, the robot thinks that it is safe and keeps moving forward, hence there is no need for avoiding. Otherwise, the robot avoids the obstacle by adjusting speed. We obtain our safe distance of three meters when the dynamic obstacle occupies about a quarter of the camera’s field of view. Therefore, the closer the obstacle is, the larger the proportion, and the more intensified the tendency of the robot to avoid it.

The second experiment was carried out when the dynamic obstacle approaches the moving robot at the speed of 1 m/s, 2 m/s and 3 m/s respectively. We also consider people as dynamic obstacles. The moving obstacle is coming from four meters away...
from the robot. As shown in Fig. 18(a), when the obstacle moves at 1 m/s, the robot moves forward normally in the first second and adjusts the movement speed during the second and the third seconds to avoid the dynamic obstacle gradually. In Figs. 18(b) and 18(c), when the obstacle is at the speed of 2 m/s and 3 m/s respectively, there is almost no time to avoid in advance gradually for the robot. Instead of changing its speed step by step, the robot stops moving and performs pivot steering to avoid collision when the obstacle is soon. Therefore, when the obstacle moves less than about 1 m/s, the robot can avoid during its traveling. When the obstacle moves faster, the robot almost stops moving forward and steers clear of it. The experimental results show that the navigation approach is robust and safe for the robot when the obstacle is very soon.

The third experiment is dynamic obstacle avoidance in our lab which is shown in Fig. 19. The obstacle is about one point five meters away from the robot. When the robot went straight normally (Fig. 19(a)), it was faced with a dynamic obstacle blocking its path (Fig. 19(b)). The local planner reacted to the obstacle by reducing the linear velocity dramatically and swerving to the right (Fig. 19(b)). After the obstacle disappeared, the robot corrected its orientation (Fig. 19(c)) and moved on (Fig. 19(d)). The velocity change of the above process is presented in Fig. 20. Angular velocity and linear velocity rise and fall alternately with the appearing and disappearing of the dynamic obstacle.

We conducted the similar experiment in the corridor which is shown in Fig. 21. As we can see, after the beginning of straight on (Fig. 21(a)), a person suddenly appeared in its right (Fig. 21(b)). The system made a quick reaction and turned left (Fig. 21(c)). When the person left, the robot altered its heading and went through the corridor continuously (Fig. 21(d)). The homologous velocity change is shown in Fig. 22, that is an alternation variation to avoid the dynamic obstacles.

The fourth experiment was carried out when the dynamic obstacle passed quickly in front of the robot. The moving obstacle is about one point five meters away from the robot. As shown in Figs. 23 and 24, initially, the robot keeps moving forward. When the object suddenly appears from the left side, the robot stopped. The linear velocity drops sharply to a minimum and the angular velocity increases rapidly to its maximum which is try to steer clear of obstacles. As the obstacle moves quickly from one side to the other, the angular velocity reverses and the linear velocity is still at a minimum. In such process, the robot has almost no real displacement and rotation and just tries to turn because the obstacle travels in a very short time. After that, the speed returns to the initial state, and the robot keeps going. Therefore, our navigation system can handle such emergency situation successfully.

5. Conclusions

In this paper, we have developed a low-cost visual navigation method without prior map assistance and the location information is not required which can be thought of as local navigation and deployed on indoor mobile robots for service application.

Initially, a shallow CNN network for scene classification was designed which reached a higher accuracy and speed after training compared to previous related work. The scene perception model is end-to-end trainable which avoided building a complex multisensor model. Then the AWC algorithm was applied for the real robot motion control to improve the robustness and smoothness. With the above improvements, the navigation system became more robust and performed better in complicated and cluttered surroundings compared to the existing studies.

Our experiments showed a comprehensive evaluation of our method. On the one hand, we provided the qualitative and quantitative comparison of the proposed method which showed better results in safe traveling distance and time compared to the state-of-the-art scene classification based visual navigation methods. On the other hand, the extensive evaluation in corridor environments and dynamic conditions performed the generalization to different indoor scenarios of our method.

Our future work includes reinforcement learning based visual perception in our navigation system to further improve the capability of self-learning and continuous decision-making. Additionally, we plan to use an embedded board with its low
power consumption and cost, high performance and advanced hardware/software compatibility, instead of the using of a laptop as the upper processor. It will improve the integration, flexibility and micromation of the system.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the National Natural Science Foundation of China [grant number U1813220 and 61662075].

References

[1] Marder-Eppstein E, Berger E, Foote T, Gerkey B, Konolige K. The Office Marathon: Robust navigation in an indoor office environment. In: 2010 IEEE international conference on robotics and automation. 2010. p. 300–7.
[2] Triebel R, Kai A, Alami R, Beyer L, Zhang L. SPENCER: A socially aware service robot for passenger guidance and help in busy airports. In: Field & service robotics. 2015.
[3] Takahashi M, Suzaki T, Shitamoto H, Moriguchi T, Yoshida K. Developing a mobile robot for transport applications in the hospital domain. Robot Auton Syst 2010;58:889–99. http://dx.doi.org/10.1016/j.robot.2010.03.010.
[4] Bačík J, Durovsky F, Biros M, Kyslan K, Perdukova D, Sanjeevikumar P. Pathfinder – development of automated guided vehicle for hospital logistics. IEEE Access 2017;5:26892–900. http://dx.doi.org/10.1109/ACCESS.2017.2767899.
[5] Chen C, Zhang Y, Tang J, Cao H, Liu J. Dual-optimization for a MEMS-INS/GPS system during GPS outages based on the cubature Kalman filter and neural networks. Mech Syst Signal Process 2019;133:106222. http://dx.doi.org/10.1016/j.ymssp.2019.07.003. URL http://www.sciencedirect.com/science/article/pii/S0888327019304303.
[6] Skewes T, Evans J, Lumelsky V, Krishnamurthy B, Barrows B. Motion planning for a hospital transport robot, vol. 1. 1991, p. 58 – 63. http://dx.doi.org/10.1109/ROBOT.1991.131553.
[7] Wang C, Savkin A, Clout R, Nguyen H. An intelligent robotic hospital bed for safe transportation of critical neurosurgery patients along crowded hospital corridors. IEEE Trans Neural Syst Rehabil Eng 2014;23. http://dx.doi.org/10.1109/TNSRE.2014.2347377.
[8] Pfrunder A, Borges P, Rechy Romero A, Catt G, Elfes A. Real-time autonomous ground vehicle navigation in heterogeneous environments using a 3D LiDAR. 2017. p. 2601–8. http://dx.doi.org/10.1109/ROBOT.2017.8206085.
[9] Man J, Li G, Xi M, Lei Y, Lu W, Gao X. MFNN: Position and attitude measurement neural network based on multi-feature fusion. IEEE Access 2019;7:109495–505. http://dx.doi.org/10.1109/ACCESS.2019.2933878.
[10] Gao X, Tang J, Li J, Shen C, Liu J. Attitude measurement based on imaging ray tracking model and orthographic projection with iteration algorithm. ISA Trans 2019;95. http://dx.doi.org/10.1016/j.isatra.2019.05.009.
[11] Davison AJ, Reid ID, Molton ND, Stasse O. MonoSLAM: Real-time single camera SLAM. IEEE Trans Pattern Anal Mach Intell 2007;29(6):1052–67. http://dx.doi.org/10.1109/TPAMI.2007.1049.
[12] Engel J, Schöps T, Cremers D. LSD-SLAM: Large-scale direct monocular SLAM. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. Computer vision. Cham: Springer International Publishing; 2014. p. 834–49.
[13] Mur-Artal R, Montiel JM, Tardós JD. ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Trans Robot 2015;31(5):1147–63. http://dx.doi.org/10.1109/TRO.2015.2463671.
[14] Bonin-Font F, Ortiz A, Oliver G. Visual navigation for mobile robots: A survey. J Intell Robot Syst 2008;53:263–96. http://dx.doi.org/10.1007/s10866-008-9235-4.
[15] Santos-Victor J, Sandini G, Curotto G, Garibaldi S. Divergent stereo for robot navigation: Learning from bees. In: IEEE computer vision and pattern recognition. 1993, p. 434–9. http://dx.doi.org/10.1109/CVPR.1993.341094.
[16] Temizer S, Kaelbling L. Optical flow based local navigation. 2014.
[17] Remazeilles A, Chaumette F, Gros P. Robot motion control from visual information processing systems. 2015.
[18] Wang H, Yuan K, Zou W, Peng Y. Real-time region-based obstacle detection with monocular vision. 2005. p. 615–9. http://dx.doi.org/10.1109/ROBIO.2005.246339.
[19] Kim Y-G, Kim H. Layered ground floor detection for vision-based mobile robot navigation, vol. 1. 2004. p. 13 – 18. http://dx.doi.org/10.1109/ROBOT.2004.1307122.
[20] N. Pears BL. Ground plane segmentation for mobile robot visual navigation. In: Proceedings - IEEE international conference on intelligent robots and systems. 2001, p. 1513–8.
[21] Lenser S, Veloso M. Visual sonar: fast obstacle avoidance using monocular vision. In: Proceedings 2003 IEEE/RSJ international conference on intelligent robots and systems, vol. 1. 2003, p. 886–91. http://dx.doi.org/10.1109/ROBOT.2003.1250741.
[22] Gaspar J, Winters N, Santos-Victor J. Vision-based navigation and environmental representations with an omnidirectional camera. IEEE Trans Robot Autom 2000;16(6):890–8. http://dx.doi.org/10.1109/70.862912.
[23] Blasch A, Mezour Y, Martinet P. Indoor navigation of a wheeled mobile robot along visual routes. In: IEEE international conference on robotics & automation. 2005.
[24] Ruotsalainen L, Kuusniemi H, Chen R. Heading change detection for indoor navigation with a smartphone camera. In: International conference on indoor positioning & indoor navigation. 2011.
[25] Bista SR, Giordano PR, Chaumette FC. Appearance-based indoor navigation by IBVS using line segments. IEEE Robot Autom Lett 2016;1(1):423–30.
[26] Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. In: International conference on computer vision. Cham: Springer International Publishing; 2015, p. 1510–8. http://dx.doi.org/10.1007/978-3-319-24574-4_168.
[27] Redmon J, Farhadi A. YOLOv3: An incremental improvement. 2018.
[28] Fu J, Jing L, Wang Y, Lu H. Stacked deconvolutional network for semantic segmentation, vol. PP. 2017. p. 1.
[29] Wang P, Chen P, Ye Y, Ding L, Huang Z, Hou X, Cottrell G. Understanding convolution for semantic segmentation. 2017.
[30] Lecun Y, Muller U, Ben J, Cosatto E, Flepp B. Off-road obstacle avoidance through end-to-end learning. In: International conference on neural information processing systems. 2005.

[31] Zhu Y, Mortaghi R, Kolve E, Liu JJ, Farhadi A. Target-driven visual navigation in indoor scenes using deep reinforcement learning. 2016.

[32] Gupta S, Davidson J, Levine S, Sukthankar R, Malik J. Cognitive mapping and planning for visual navigation. 2017.

[33] Richter C, Roy N. Safe visual navigation via deep learning and novelty detection. In: Amato NM, Srivinasa SS, Ayanian N, Kuindersma S, editors. Robotics: science and systems XIII. Massachusetts Institute of Technology; 2017, http://dx.doi.org/10.15607/RSS.2017.XIII.064, URL http://www.roboticsproceedings.org/rss13/p64.html.

[34] Chen W, Qu T, Zhou Y, Weng K, Wang G, Fu G. Door recognition and deep learning algorithm for visual based robot navigation. In: 2014 IEEE international conference on robotics and biomimetics. 2014. p. 1793–8. http://dx.doi.org/10.1109/ROBIO.2014.7090595.

[35] Giusti A, Guzzi J, Dan C, He FL, Rodriguez JP, Fontana F, Faessler M, Forster C, Schmidhuber J, Caro GD. A machine learning approach to visual perception of forest trails for mobile robots. IEEE Robot Autom Lett 2017;1(2):661–7.

[36] Kim Y-H, Jang J-I, Yun S. End-to-end deep learning for autonomous navigation of mobile robot. 2018. p. 1–6. http://dx.doi.org/10.1109/IICCE.2018.8326229.

[37] Hu S, Cao C, Pan J. Deep-learned pedestrian avoidance policy for robot navigation. In: 2017 IEEE international conference on robotics and biomimetics. 2017, p. 338–43. http://dx.doi.org/10.1109/ROBIO.2017.8324440.

[38] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: International conference on neural information processing systems. 2012.

[39] Inoue H. Data augmentation by pairing samples for images classification. 2018.

[40] Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, Kudlur M, Levenberg J, Monga R, Moore S, G. Murray D, Steiner B, Tucker P, Vasudevan V, Warden P, Zhang X. Tensorflow: A system for large-scale machine learning. 2016.

[41] Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE 1998;86(11):2278–324. http://dx.doi.org/10.1109/5.726791.

[42] Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. Comput Sci 2012;3(4); págs. 212–223.

[43] Kingma D, Ba J. Adam: A method for stochastic optimization. Comput Sci 2014.

[44] Grisetti G, Stachniss C, Burgard W. Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Trans Robot 2007;23(1):34–46.

[45] Cherlur S, Woolley C, Vanderschreft P, Cohen J, Tran J, Catanzaro B, Shelhamer E. CuDNN: Efficient primitives for deep learning. Comput Sci 2014.