A Few-shot Dynamic Obstacle Avoidance Strategy in Unknown Environments

Xiaoxiao Li, Sheng Bi, Yongxing Wang, Min Dong*, Zhangshao Chen
Dept of Computer Science and Engineering
South China University of Technology
Guangzhou, China
li.xiaoxiao@mail.scut.edu.cn, picy@scut.edu.cn, kingyouxing@outlook.com, *hollymin@scut.edu.cn

***Abstract*—Obstacle avoidance is one of the basic capabilities of intelligent mobile robots. With the diversification of the application environment, mobile robots are required to avoid obstacles with higher generality. Benefit from the development of mobile platform and deep learning algorithm in recent years, we conceive a few-shot dynamic obstacle avoidance strategy to meet this higher generality demand. Under this metric-based meta-learning method, mobile robots can quickly adapt to unknown environments by learning from several samples. In order to verify its effectiveness, we use this strategy to train a model and deploy it to the mobile robot and run multiple obstacle avoidance recognition tests in the real-world environment. The results of experiments performed on the mobile robot platform illustrates a good performance and verifies our proposed strategy. In addition to analyzing the experimental results, the advantages, disadvantages as well as application potential of the proposed strategy as a decision aid are also discussed.**

***Index Terms*—few-shot learning, obstacle avoidance, mobile robot, unknown environment**

I. Introduction

A. Background

Obstacle avoidance is the basic requirement for autonomous navigation of the mobile robots. With the increasing complexity of application scenarios, the reliability and universality of obstacle avoidance for mobile robots are becoming higher and higher. Traditional obstacle avoidance algorithms, such as grid method and visibility graph [1] have good performances in dealing with known obstacles, but they often fail to work well when there are unknown or dynamic obstacles. Therefore, a series of intelligent obstacle avoidance algorithms, such as genetic algorithm [2] [3], artificial neural network [4], fuzzy control [5], etc., are generated, and the performance of these algorithms depends on the design of the geometric model. In recent years, with the update of the processor and the application of various advanced algorithms, it becomes easier to carry out complex operations on the mobile robot platform. In order to promote the intelligence of mobile robots, deep learning algorithms are also being applied to mobile robots to avoid obstacles [6] [7]. However, there is no any kind of algorithm can achieve relatively effective obstacle avoidance in any environment. How to improve the obstacle avoidance algorithm to make it obtains generality is still an important research direction at present. By referring to human abilities, obstacle avoidance can be carried out effectively in different environments according to the previous experience. Human-like learning is a feasible way to solve the general problem of obstacle avoidance algorithm. The generation of few-shot learning [8] is to build a bridge between traditional deep learning and human-like learning [9].

This research work is supported by National Natural Science Foundation of China (61703168, 61703284). Guang dong province science and technology plan projects (2017A030310163, 2017A010101031). Shen zhen basic research project (JCYJ20160429161539298). Shen zhen peacock project (KQTD20140630154026047).

*Corresponding author is Min Dong. Email: hollymin@scut.edu.cn

B. The need of few-shot learning in robotics

Mobile robots are often arranged to work in diverse environments. Traditional deep learning algorithms are driven by large amounts of data, but training models by collecting large numbers of labeled samples in privacy and dangerous environments is expensive or even impossible. How to adapt to the new environment as quickly as humans has become a research hotspot. Few-shot learning solves the problem of lack of samples by referring to prior knowledge and achieves the goal of learning new tasks quickly, which is exactly the defect of obstacle avoidance of mobile robots. From the perspective of few-shot learning, the ability of mobile robots to adapt to the new environment can be improved, which can greatly save the learning cost and achieve fast learning.

C. Contributions

Based on the embedded learning [10] method, we proposed a few-shot obstacle dynamic avoidance strategy to solve the problem of mobile robots quickly adapting to unseen environments. The main contributions of this paper are as follows:

1) Quickly adapt to the unknown environment: only a few labeled samples are needed to achieve the ideal obstacle avoidance effect without retraining the learning model.

2) Human-like ability to avoid obstacles: decisions making are based on monocular vision information without using accurate distance and azimuth.

D. Organization

The follow-up part of this paper is organized as follows: Section II provides an overview of the recent related works in

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

few-shot learning and visual based obstacle avoidance. Section III is an introduction to the proposed obstacle avoidance scheme. Experimental results are given in Section IV, follow by the discuss and analysis in Section V. The conclusion and supplementary reference of related materials are provided at the end.

II. Related works

At present, there are few studies on the practical application about few-shot learning, and there is no application in mobile robot. Therefore, the related works are respectively illustrated from the two aspects of few-shot learning and deep learning in obstacle avoidance.

A. Few-shot learning

As an emerging hot topic of deep learning, a variety of solutions of few-shot learning have emerged. The mainstream of existing few-shot learning methods is embedded learning [9]. The samples are mapped to a smaller embedded space to distinguish between similar and different samples. This method is mainly used for classification tasks by measuring the similarity between samples and embedded Spaces. The ProtoNet [11], Matching Nets [12], Relation Net [13] are all realized based on this method. Embedded learning usually adopts meta-learning [14] [15] training process, and the model learned can be quickly transferred to other tasks. There are also methods based on data augmentation. The method in [16] is to learn a set of encoders from similar classes to perform intra-class transformations to synthesize new samples for an unseen class. [17] augments the dataset by migrating from a large number of unlabled or weakly labled datasets. The effectiveness of such methods depends on external data and is not always applicable. Handcrafted rules are commonly used as pre-processing step, such as mapping [18], rotation [12], flip [19] [20], etc. Santoro et al. [19] proposed the memory-augment strategy based on external memory [21] to solve few-shot learning tasks. Such methods rely on human experience to design rules for memory updating and storage. Other methods such as fine-tuning [22] [23], Model-agnostic meta-learning (MAML) [14], Meta-Learner LSTM [24] are focused on how to optimize parameters to reduce overfitting caused by lack of samples. Our strategy is similar to the ProtoNet [11], adopting the metric-based meta-learning method.

B. Deep learning in obstacle avoidance

Deep learning-based obstacle avoidance algorithms [7] [25] have gained attention in recent years because they do not need to establish geometric models and adjust a large number of parameters, and can directly benefit from large data sets and continuous use. L. Tai et al. [6] proposed a deep-network solution to treat obstacle avoidance as a depth image classification task. It combines the hierarchical structure with the convolutional neural network (CNN) [26] to realize the model-free obstacle avoidance with the raw depth image as the input and the control command as the output. When perception is limited to monocular vision without three-dimensional information, L. Xie et al. [7] proposed a deep reinforcement learning algorithm based on deep double-Q network (D3QN), which avoids obstacles by predicting depth information from RGB images. There are similar methods like [27] which achieve indoor visual navigation without additional map information and manual guidance through distributed deep reinforcement learning in different regions. Kovacs et al. [28] proposed the method of image region classification based on relative focus map, which does not require prior training. Based on the advantages of these methods, we solve the problem of few-shot obstacle avoidance as an image classification task.

III. Few-shot Dynamic Obstacle Avoidance

In this section, we will introduce the proposed few-shot dynamic obstacle avoidance strategy, which is used to adapt a mobile robot to an unknown environment with a few samples and without retraining the learning model. It comprises a feature extractor, metric-based decision making process and meta-training process. The feature extractor generates a set of feature vectors of the raw monocular visual data, which are embedding to a high dimensional space. A similarity measurement function is used to calculate the similarity between the test data and the embedded space for decision making. This whole end-to-end process framework is shown as Fig. 1. With the method of meta-learning, the above process can be rapidly generalized to new tasks.

A. Feature extraction

The feature extractor is implemented based on the EfficientNet-B0 [29]. The baseline model, shown as Fig. 2, is built with mobile inverted bottleneck convolution (MBConv) [30]. It is a network model for mobile applications with 5.3M parameters. We feed the raw RGB image with a resolution of 224x224 into the model and normalized it before training, and obtain its feature map through multi-layer convolution. To improve the speed of calculation without affecting the accuracy, there is a dropout layer with dropout rate equals 0.2 after a batch normalization and 2x2 average-pooling layer. Then squeeze its feature map to feature vector for later embedding.

B. Metric-based decision making

As a few-shot learning problem, we will introduce our strategy in terms of general problem definitions. A task with N categories and K labeled samples per categories is called N-way K-shot task. The sample is expressed as <x, y>. K samples in each category constitute the support set S of corresponding category. The feature vectors F in support set of category C are embedded into a low dimensional space to obtain its embedded representation E:

$$E_C(x) = \frac{1}{K} \cdot \sum_{i=1}^{K} F_{x_i} \tag{1}$$

Equation (1) is to compute the average vector as an embedded representation of category C, F_{x_i} is feature vector of sample x_i in the support set of C.

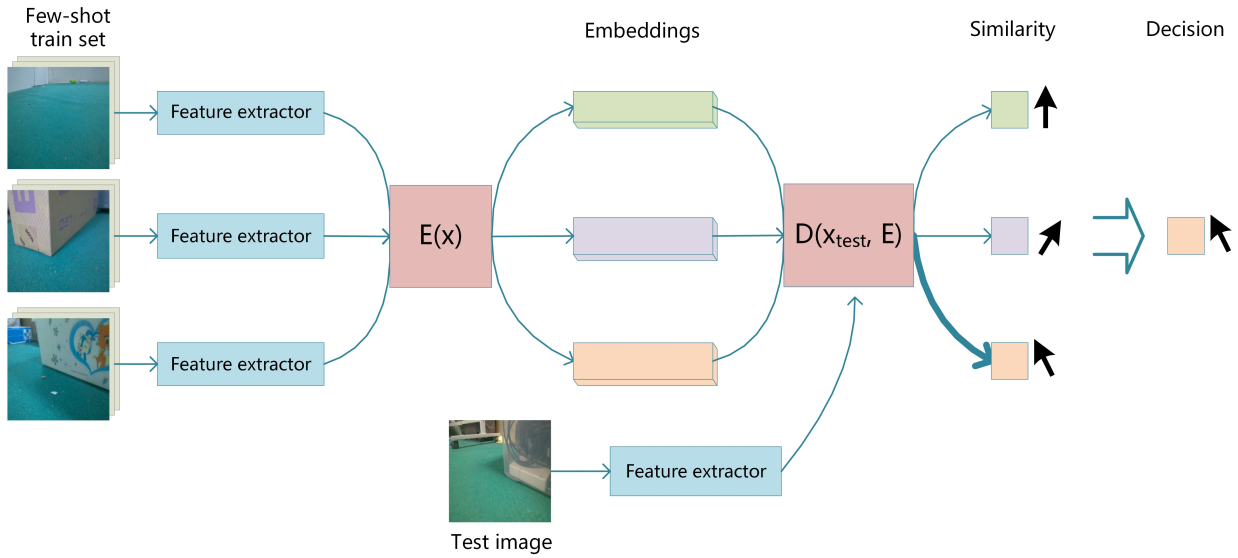


Fig. 1. **Few-shot Dynamic Obstacle Avoidance Framework.** E(x) is an embedding function (Equation (1)), D(x_{test}, E) denotes the similarity measurement (Equation (2)), and black arrows stand for the different decisions.

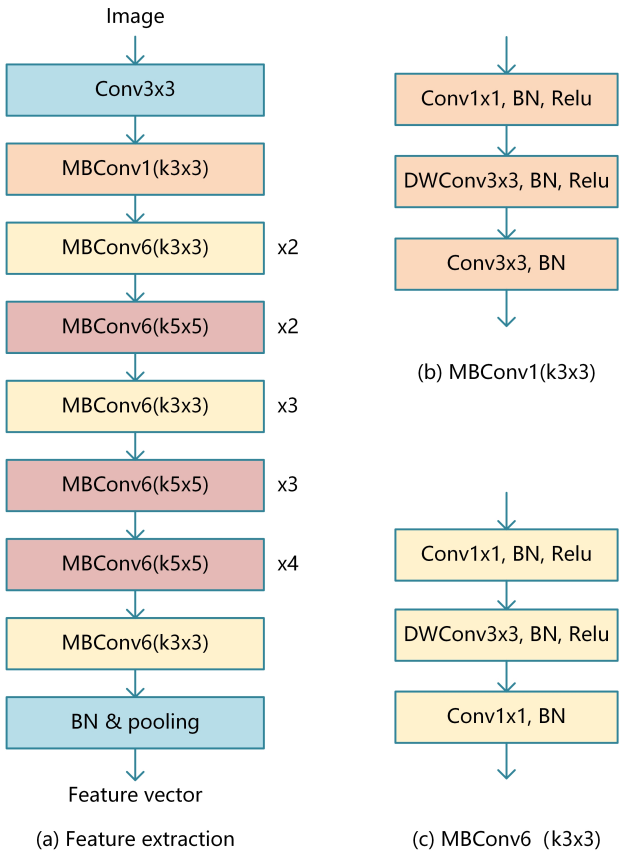


Fig. 2. **Feature extraction network.** (a) is the model architecture; (b) and (c) are structures of middle layers. Conv stands for standard convolution, DWConv is depthwise convolution, 1x1/3x3/5x5 means kernel size, BN is batch normalization, Relu is ReLU activation function, and x2/3/4 is the repeated number of its left layer.

We use the distance from the test sample to different embedded spaces to measure its similarity with each category which is expressed as D(x_{test}, E) in Fig. 1. Similar to the ProtoNet [11], we adopt Euclidean distance as similarity measurement:

$$D(x_{test}, E_C) = \sqrt{(F_{x_{test}} - E_C)(F_{x_{test}} - E_C)^T} \quad (2)$$

The smaller the distance, the more similar the test sample is to the category. In general, the similarity is numerically reflected as s $\in$ [0, 1] and the closer s gets to 1, the more similar it is. Then the similarity can be written as:

$$s(x_{test}, E_C) = \frac{1}{1 + D(x_{test}, E_C)} \quad (3)$$

We calculate the similarity between the test samples and N categories, and select the category corresponding to the maximum s as the prediction result to make the decision. This decision-making process is based on metric information, so in the new environment that also deals with obstacle avoidance tasks, only few labeled samples are needed to build the support set shown in Fig. 3 without retraining the model in order to learn the environmental information into the model.

C. Meta-learning process

The learning process adopts the episode-based meta-learning methods proposed by [24]. It is a meta-learner which contains two nested learning procedures. Fig. 3 illustrates the data composition in the learning process. The learning process consists of two phases, meta-training for learning few-shot learning ability and meta-validation for determining hyper parameters. The meta-training procedure involves many few-shot learning tasks, and each task contains a train set and a test set sampled from meta-training dataset. In each task, we use the same train set to predict a set of test images for evaluation, then we obtain a predicted distribution after feature

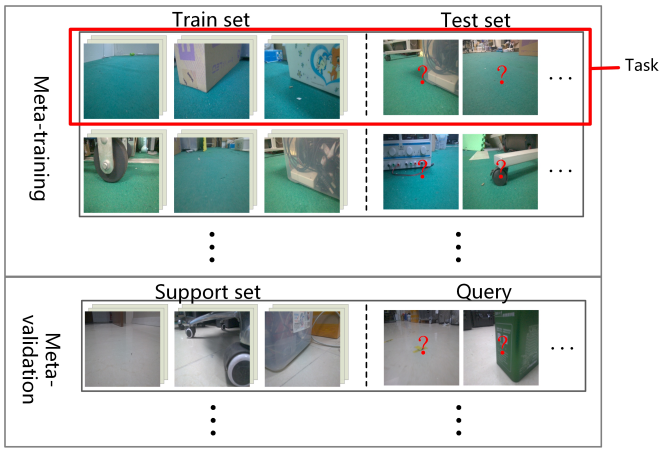


Fig. 3. **An epoch in meta-learning process.** In the red box is a single few-shot learning task. Train set and support have the same structure and are different names for different phases. Query works the same as test set.

extraction and decision making to calculate the single task loss and accuracy. The loss is calculated using the cross entropy loss function:

$$Loss = -\sum_{i=1}^{N} p(y_i) log(p(\hat{y_i})) \tag{4}$$

where $p(y_i)$ is the probability of the ground-truth and $p(\hat{y_i})$ is the probability of the prediction. We update the model parameters by using an Adam optimizer to optimize loss.

After we the meta-training process, there is a series of test tasks of a new environment in the meta-validation process to verify the learning ability of the model which is evaluated with accuracy.

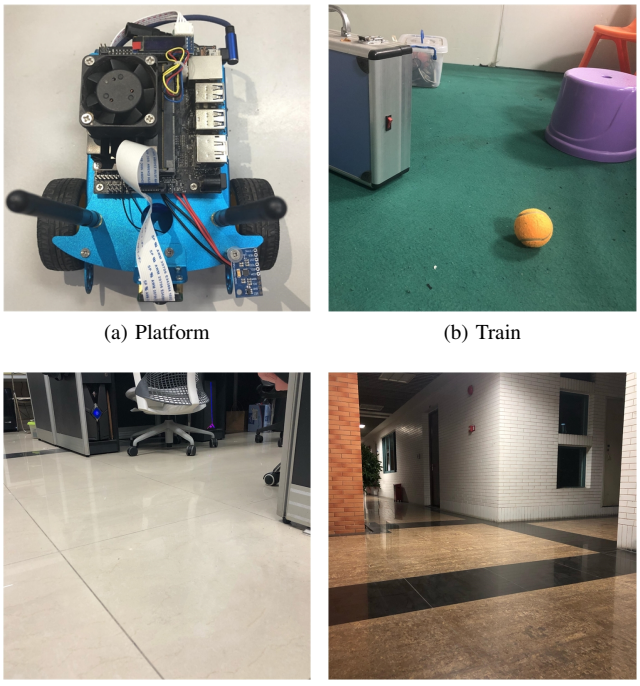
IV. Experiments and Results

A. Platform and environments

In order to verify the effectiveness of our strategy, the experiment is carried out on NVIDIA Jetbot, a mobile robot platform shown in Fig. 4(a) consisting of Jetson Nano development board for data processing, vehicle frame, steering gear, power supply, Raspberry Pi camera V2.1 for RGB image acquiring, and other components. The Jetson Nano is equipped with a 4 core Cortex-A57 CPU, an 128 core Maxwell GPU and 4GB LPDDR4 memory. It is a a powerful development board for edge computing. Fig. 4(b) is the environment we use to collect the dataset for training and Fig.4(c) is the environment to collect dataset for validation. The test environment is shown as Fig.4(d). We also collect a dataset in test environment to evaluate model capability by accuracy.

B. Data collecting

In our experiment, we collect three datasets from three different environments for training, validation and testing. And the ground-truth labels are manually marked. To be specific, we control the mobile robot to walk in different environments



(a) Platform (b) Train

(c) Validation (d) Test

Fig. 4. **Platform and environments.** (a) the mobile robot platform for experiments, (b) and (c) are environments to collect dataset for training and validation, and (d) is the test environment.

and execute various instructions to avoid collisions. Meanwhile, samples are collected by the monocular camera on the mobile robot and labeled as “free”, “left-blocked” and “right-blocked” categories corresponding to “go forward”, “go right-forward”, “go left-forward” three decisions. So we have three datasets that each have three categories and each category contains 300 RGB images. The training dataset is from the artificial environment, and the validation and test dataset are from the real-world environment. Two of them are used to train the obstacle avoidance model in the section III, and one is used to test and construct the support set required for actual obstacle avoidance.

C. Model training and deployment

The acceleration environment can be configured on Jetbot for training, but in order to reduce the training time, we trained and tuned the model on the server with GeForce RTX 2080 Ti GPUs, and then transplanted the model to Jetbot. We have a PyTorch [31] framework configured on both the server and Jetbot for training and evaluation. The original RGB image size collected from camera is 300x300, and we resize the samples to a resolution of 224x224 before training to reduce the computational cost. Then put them into the meta-learning cycle introduced in Section III. We set up 100 epochs for training, meta-training batchsize is 100 and meta-validation batchsize is 200. It means that there are 100 tasks in meta-training phase and 200 tasks in meta-validation phase in each epoch. Each task is a 3-way 5-shot problem with query equals

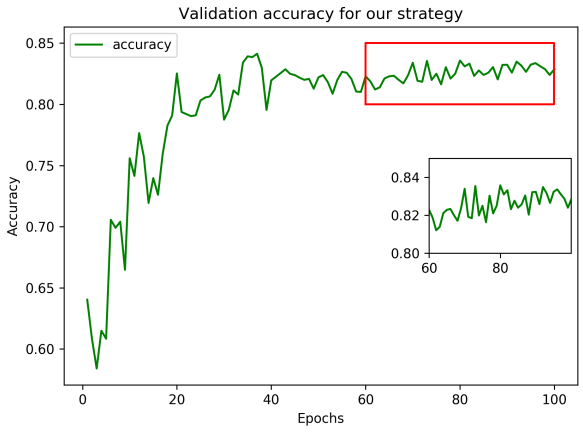


Fig. 5. **Accuracy on validation dataset during training.** The subgraph on the right magnifies the final validation result of the model.

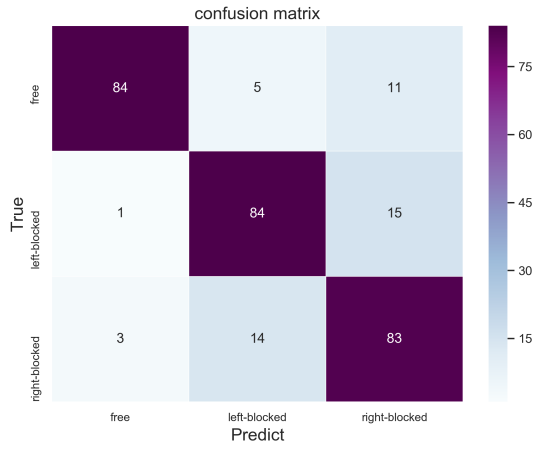


Fig. 6. **Confusion matrix on test dataset.** The values on the map represent the distribution of the prediction.

15. The learning rate of training process is 1e-3, and we load 'efficientnet-b0' as the pre-training model and take its weights as the initial point of training.

When deploying the model to Jetbot, we prepared a support set with five images for each of the three categories from the test environment. After loading the model, the real-time video frames are predicted according to the support set, and the predicted results are respectively corresponding to the instructions to be taken.

D. Results and evaluation

In the actual test environment shown in Fig. 4(d), the mobile robot successfully avoids all obstacles, including static tables, chairs and posts and dynamic pedestrians. The average processing time is 0.131s per frame. The whole reaction time from reading the video frame to making the decision is about 0.15s. This processing speed is enough for robot to get the next instruction before the decision is completed. As a task-driven rather than precision-driven problem, our strategy reach the ideal results in the experiment. To improve the persuasiveness, we also evaluate the effectiveness of the strategy numerically.

Firstly, we show the validation result of model training as Fig. 5. The way the model learns determines that its results in the validation phase will be similar to those during the test, and the results in Fig. 6 also confirm this statement. We could see that the initial accuracy is 64.2%. During the first 40 epochs of learning, the fluctuation is strong, but it is in the normal fluctuation rising state. This indicates that the 3-way 5-shot learning ability of the model is gradually enhanced. After that, the accuracy gradually stabilized and finally reached 83.1% by the time the training was completed.

We randomly selected 5 samples from each categories of test dataset to construct a support set. The mobile robot is arranged to run in the test environment shown in Fig. 4(d), which is tested with different obstacles. Each category is tested 100 times and the model's performance is shown in Fig. 6. It can be seen that the overall test accuracy is 83.6%, and the correct amount of decision for three categories of "free", "left-blocked" and "right-blocked" are 84, 84 and 83. The accuracy is in line with the training results and expectations. From the wrong decisions, 1 sample in"left-blocked" and 3 in "right-blocked" are predicted to be "free" which means that the probability of making a "go forward" decision is very small when there are obstacles. 5 samples in "free" and 14 in "right-blocked" are predicted to be "left-blocked", and 11 and "free" and 15 in "left-blocked" are predicted to be "right-blocked". The error rate is not high but it means that the robot is likely to make a turn where it needs to go straight, and turn in the wrong direction when it needs to turn. We will analyze the possible causes of this type of error in section V.

V. Analysis and Discussion

In our strategy, the model treats obstacle avoidance as a classification problem, and is trained in a supervised way. It enables a mobile robot to complete a new task using a few samples. In this section, we are going to analyze the experimental results and discuss the advantages and disadvantages of our strategy as well as its application potential.

A. Experimental analysis

Considering going straight, turning left-forward and right-forward are the three most basic forward movements and one of the two movements rather than going straight would be made when there is an obstacle right in front. Therefore, the task was set as a 3-way problem in experiments. The probability of random guessing was 33.3%, but the initial accuracy of model learning was more than 60%. We tried to replace the feature extraction network and cancel the use of the pre-trained model then observe the corresponding learning curve. It was found that the initial accuracy using the pre-trained model was significantly better than the unused one, while the model using another feature extraction network is not different in this respect. It explains the importance of

prior knowledge in few-shot learning task, just like human-beings who's ability to complete new tasks is influenced by accumulated experience.

According to the experimental results, it can be concluded that the robot could avoid all the obstacles in the real-world environment, but the accuracy of the model was only about 83%. From the distribution of prediction results, our strategy can accurately predict the need for turning. We have observed samples of decision-making errors, and found that the errors occurred when the obstacles accounted for a relatively small proportion in the image, that is, the robot still had some moving space. So the mobile robot will not collide even if it is wrong. And when making a wrong decision on the free area, the car will deviate from the path rather than collide. The main reason for this is that our test set came from a dark ceramic tile environment, and some test samples had obvious reflective conditions. This also reflects that our strategy has the same limitations as other monocular visual obstacle avoidance methods, which are greatly affected by light, visibility and other factors. It is not robust enough as a single sensor obstacle avoidance scheme.

In addition to the above experiments, we also tried some methods to improve the accuracy without adjusting the framework of the strategy. The most intuitive and effective method is to use operations such as scaling or rotation to to amplify the data of the support set, or to construct multiple support sets for streaming processing of the support set to make voting decisions but this method will obviously affect the processing speed.

B. Discussion

Our strategy works well in common environments, but it is only a feasible but imperfect idea to solve the generality problem of obstacle avoidance of mobile robot. Due to some inherent defects and advantages of the vision sensor, the obstacle avoidance scheme based on monocular vision is often used to integrate with other sensors. Our strategy can realize the integration with other schemes at the decision-making level to deal with some extreme situations, for example, the 2D-laser cannot detect the stair edge below the plane and the visual information is obviously different from the free area. Because of its low learning cost, it has an advantage over other deep learning programs. And its human-like learning ability is worthy of further study in the process of intelligent mobile robots. For example, how to make the obstacle avoidance route of the robot more stable and less affected by environmental factors, and how to use this strategy to achieve a complete autonomous navigation system and how to apply it in a more complex environment.

VI. Conclusion

In this paper, we proposed a few-shot dynamic obstacle avoidance strategy based on monocular visual information which enables mobile robots to quickly adapt to unknown environment, and accomplished the obstacle avoidance tasks in the real-world environment. Experimental results show that our scheme can successfully accomplish the few-shot obstacle avoidance task. In principle, it allows mobile robots to learn much like humans and develop human-like abilities. Nevertheless, the strategy has its limitations, such as the inability to accumulate knowledge learned in different test environments, so it may not be the most suitable way for practical applications. We will look at how to achieve the accumulation of experience in all the walked environment to omit the step of collecting support sets changing environments in the future research.

References

[1] B. Patle, G. B. L, A. Pandey, D. Parhi, and A. Jagadeesh, "A review: On path planning strategies for navigation of mobile robot," *Defence Technology*, vol. 15, no. 4, pp. 582 – 606, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2214914718305130

[2] W.-G. Han, S.-M. Baek, and T.-Y. Kuc, "Genetic algorithm based path planning and dynamic obstacle avoidance of mobile robots," in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 3. IEEE, 1997, pp. 2747–2751.

[3] P. Shi and Y. Cui, "Dynamic path planning for mobile robot based on genetic algorithm in unknown environment," in *2010 Chinese control and decision conference*. IEEE, 2010, pp. 4325–4329.

[4] R. Glasius, A. Komoda, and S. C. Gielen, "Neural network dynamics for path planning and obstacle avoidance," *Neural Networks*, vol. 8, no. 1, pp. 125–133, 1995.

[5] K. M. Passino, S. Yurkovich, and M. Reinfrank, *Fuzzy control*. Citeseer, 1998, vol. 42.

[6] L. Tai, S. Li, and M. Liu, "A deep-network solution towards model-less obstacle avoidance," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 2759–2764.

[7] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," *arXiv preprint arXiv:1706.09829*, 2017.

[8] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[9] Y. Wang and Q. Yao, "Few-shot learning: A survey," *CoRR*, vol. abs/1904.05046, 2019. [Online]. Available: http://arxiv.org/abs/1904.05046

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[11] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4077–4087.

[12] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," in *Advances in neural information processing systems*, 2016, pp. 3630–3638.

[13] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1199–1208.

[14] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1126–1135.

[15] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," 2018.

[16] E. Schwartz, L. Karlinsky, J. Shtok, S. Harary, M. Marder, A. Kumar, R. Feris, R. Giryes, and A. Bronstein, "Delta-encoder: an effective sample synthesis method for few-shot object recognition," in *Advances in Neural Information Processing Systems*, 2018, pp. 2845–2855.

[17] T. Pfister, J. Charles, and A. Zisserman, "Domain-adaptive discriminative one-shot learning of gestures," in *European Conference on Computer Vision*. Springer, 2014, pp. 814–829.

[18] J. Kozerawski and M. Turk, "Clear: Cumulative learning for one-shot one-class image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3446–3455.

[19] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*, 2016, pp. 1842–1850.

[20] P. Shyam, S. Gupta, and A. Dukkipati, "Attentive recurrent comparators," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3173–3181.

[21] S. Sukhbaatar, J. Weston, R. Fergus *et al.*, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[22] S. Arik, J. Chen, K. Peng, W. Ping, and Y. Zhou, "Neural voice cloning with a few samples," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 019–10 029.

[23] S. Azadi, M. Fisher, V. G. Kim, Z. Wang, E. Shechtman, and T. Darrell, "Multi-content gan for few-shot font style transfer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7564–7573.

[24] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[25] J. A. Caley, N. R. Lawrance, and G. A. Hollinger, "Deep learning of structured environments for robot search," *Autonomous Robots*, vol. 43, no. 7, pp. 1695–1714, 2019.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[27] S. Hsu, S. Chan, P. Wu, K. Xiao, and L. Fu, "Distributed deep reinforcement learning based indoor visual navigation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 2532–2537.

[28] L. Kovacs, "Visual monocular obstacle avoidance for small unmanned vehicles," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016.

[29] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[30] M. Tan, B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le, "Mnasnet: Platform-aware neural architecture search for mobile," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2820–2828.

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf