Training networks separately on static and dynamic obstacles improves collision avoidance during indoor robot navigation

Viktor Schmuck and David Meredith

Aalborg University, Aalborg, Denmark
{vsch,dave}@create.aau.dk

Abstract. Autonomous robot navigation and dynamic obstacle avoidance in complex, cluttered, indoor environments is a challenging task. A robust solution would allow robots to be deployed in hospitals, airports or shopping centres to serve as guides and fulfil other functions requiring safe human–robot interaction. Previous studies have explored various approaches to selecting sensor types, collecting data, and training models capable of safely avoiding unmapped, possibly dynamic obstacles in an indoor environment. In this paper we address the problem of recognizing and anticipating collisions, in order to determine when avoidance manoeuvres are required. We propose and compare two sensor-fusion and neural-network-based solutions, one in which models are trained separately on static and dynamic samples and another in which a model is trained on samples of collisions with both dynamic and static obstacles. The measured accuracies confirmed that the separately trained, ensemble models had better recognition performance, but were slower at calculation than the models trained without taking the obstacle types into account.

1 Introduction

The research conducted on object avoidance allows for new approaches to the development of way-finding robots, that can be deployed to perform various tasks in densely populated environments. Such robots could be used in complex indoor environments, such as hospitals or airports, where a guide-robot can help users navigate to their destinations within the building. This guidance also entails that the robot is capable of navigating an environment in an optimal, safe and predictable manner. However, there is a gap between the perception of the environment and the decision-making required for such tasks.

This paper addresses the implementation of danger recognition required to create a neural-network-based controller which can handle the detection of dynamic and static obstacles with which the robot might collide. This is achieved by collecting data on scenarios leading to collision, and by training and combining convolutional and deep neural networks (CNNs and DNNs) for the recognition of situations that are likely to result in collisions. Our solution is designed to classify static and dynamic obstacles in the environment, and estimate the likelihood of the robot colliding with these obstacles. This situation assessment system can then be used in combination with various avoidance policies to allow a robot to avoid collisions with both static and dynamic obstacles while navigating in an indoor environment.

In section 2 we provide an overview of related work. The proposed implementation is described in section 3. Section 4 describes the evaluation and results, and section 5 presents a brief discussion and the main conclusions of this paper.

2 Related work

Several solutions have been proposed for navigating in dynamic environments. Most recent approaches use fuzzy logic controllers, which have four main steps [1]: (1) defining linguistic variables for input and output systems; (2) defining fuzzy set; (3) defining rules of the set; and (4) defuzzification. In such systems, the linguistic variables for the inputs are robot sensor readings and the outputs are values controlling the robot's movement. In steps 2–3, the defined rule set is applied on the transformed sensor data which produces one of the expected outputs after step 4 [2]. The fuzzy logic model in such systems can be defined by algorithms such as Minguez and Montano's [3] nearness diagram model. In their method, the robot re-analyses its environment after each movement and determines the next action accordingly. In their approach, the robot performs pre-programmed behaviour based on its ability to recognize a defined set of scenarios. In another approach, Large *et al.* [4] use an NLVO algorithm to predict future obstacle positions and whether collisions will occur. Use of the A* algorithm [5] allowed their robot to react quickly to changes in the environment and avoid high-risk situations. In the domain of *neuro-fuzzy control* [1], neural networks compute the outputs used for navigation. Lecun *et al.* [6] propose a neuro-fuzzy solution for off-road obstacle avoidance based on a CNN, which outputs a set of steering angles computed from input from two cameras. Data collection was performed in various off-road environments, during which the steering angle was modified only when necessary and the distance from the avoided obstacle was kept consistent. The developed avoidance model was reliable in multiple off-road environments. Wang *et al.* [7] describe a similar solution for flying unmanned aerial vehicles (UAVs) in formation. Their solution used a modified Grossberg neural net [8], trained to identify unmapped obstacles. When such obstacles were recognized, a buffer-zone was assigned to them, and the UAVs performed avoidance manoeuvres before realigning themselves into formation and continuing towards their destination. In Gandhi *et al.*'s [9] approach to aerial dynamic object avoidance, deep neural networks were trained on pre-existing negative (collision involving) and positive (collision free) flying data, allowing a robot to be trained on real-world data without subjecting it to a dangerous environment. They gather data from various environments by flying in random directions to acquire footage that can be used to classify safe and non-safe states and determine the optimal avoidance action. Their solution permitted navigation in cluttered and confined spaces with both static and dynamic obstacles. Milde *et al.* [10] present a system implemented on neuromorphic hardware for identifying situations requiring avoidance manoeuvres. Their camera-based system could navigate in cluttered office environments and demonstrated the speed-ups achievable by using neuromorphic hardware, over other neural-network-based solutions. Nasrinahar and Chuah [2] present another fuzzy logic controller system that divides the problem into two recognition–action sub-problems by constructing different behaviours for static and dynamic obstacle avoidance. Their simulation of an indoor environment contained static and dynamic obstacles of various shapes and sizes and their simulated robot had forward-facing distance sensors. Recognition of situations requiring avoidance was based on bearings and distances to detected obstacles. Obstacle type was determined by tracking changes in these bearings and distances. The proposed avoidance behaviours were fuzzified, pre-programmed actions. Finally, Huang *et al.* [11] proposed an obstacle recognition solution based on computing bounding boxes around point clouds generated from LIDAR readings. To handle occlusion, a second classification was proposed using the mean and variance of laser intensity of the identified obstacles. The system could identify single pedestrians, crowds, vehicles and other types of obstacles.

3 Implementation

We propose a collision anticipation system which could serve as the basis of a neural-network-based navigation controller. Our solution combines a number of models, trained separately for static and dynamic collision recognition. The system is implemented on a Pepper robot,[1] a humanoid robot used in human–robot interaction. This platform was selected for its range of available sensors and integration possibilities. The Pepper robot can be programmed via a graphical interface provided by its manufacturer. However, in order to access and utilise its full range of sensors, either its provided Python SDK or ROS must be used. Our proposed system is implemented using the SDK, however a basic navigation extension would require ROS in order to integrate simultaneous localisation and mapping (SLAM).

3.1 Data collection

To gather the required data for the solution, a negative, collision-involving sample collection was used, as proposed by Gandhi *et al.* [9]. In addition, the collision data involving dynamic and static obstacles were separately gathered to allow for the training of two types of model. The recorded data holds frames gathered with the robot's two 2D cameras, and distance values gathered from its forward-facing sonar and forward-, left-, and right-facing laser rangers, each covering a 60 degree wide field.

In order to label the dataset entries, the sonar and laser ranger distance measurements were used to identify frames in which obstacles approached too close[2] to the robot, resulting in a collision. The samples describing situations resulting in a collision within one to three seconds were labelled based on the difference from the timestamps of the collision samples.

[1]https://www.softbankrobotics.com/emea/en/pepper, visited: 30/01/2019

[2]Closer than 20cm in any direction, as below that distance the robot performs an emergency stop even with reduced safety settings.

The resulting dataset holds information on over 3000 samples, describing collisions with both static and dynamic obstacles, as well as more than 2000 additional samples providing information on situations preceding collisions by up to three seconds. As a side-product of the negative sample collection method, information about collision-free samples was collected, resulting in over 8900 samples.

3.2 Designed neural networks

Our proposed system compares two approaches to neural network training: one using collisions with both static and dynamic obstacles; and another that trains separate models for static and dynamic obstacles.

For performing sensor fusion, regardless of the samples used for training, the laser and sonar readings have to be used to train a separate model from the ones trained by the two cameras. Therefore, for the distance-measurement-based classification, a two-layer deep neural network (DNN)[3] was constructed, which takes all distance readings and the robot's gyroscope values as input and, after two dropout layers, provides an estimation of the time until a possible anticipated collision, if any. The use of wheel position actuator values – tracked by the gyroscope – contributes to improved danger recognition and better estimation of time to collisions, because they describe the robot's velocity, which correlates with the relative speed of obstacles in the environment. The two camera feeds were used for the training of one convolutional neural network (CNN) each. These networks were modelled on Bojarski *et al.*'s [12] CNN for autonomous virtual vehicle navigation using a camera feed.

In order to use the outputs of all three models, a weighted average ensemble (WAE) was used to calculate the labels assigned to the frames. The chosen weights were in accordance with the individually measured evaluation accuracies of the models. As a result, for all three types of model trained on the different datasets, the sensor-input classifying DNN was assigned the highest weight of 80% while the other two CNNs each contributed to the resulting label with weights of 10%.

For the training of the models, three datasets were used: one containing all samples; one containing samples where the robot collided with only static obstacles; and a third where it only collided with dynamic obstacles. We had a train–test split of 95%–5%, and used a validation split of 10% from the selected training samples. The models were compiled to calculate MSE loss and monitor accuracy as well. The implemented early stopping prevented overfitting if training an epoch did not result in significant improvement. After training a sensor classifier DNN and two camera image classifier CNNs with all samples, the resulting models were combined by calculating a WAE. The same procedure was repeated for only static, and dynamic samples, and the resulting six models were ensembled.

Lastly, to explore the possibility of determining whether the obstacle involved

[3]With both layers having 50 nodes, a dropout rate of 0.4, and 'tanh' activation.

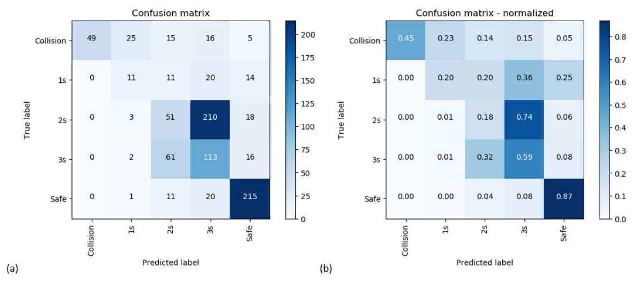

Fig. 1: Separately trained ensemble – Raw (a) and normalized (b) confusion matrices for predicted vs. true labels over the test set.

in a collision is static or dynamic, an altered version of the aforementioned CNN model was trained with smaller convolutional layer node counts and higher dropout rates.

4 Evaluation and results

The models were evaluated separately and, after applying a WAE to combine them and produce two classification pipelines, their accuracy was evaluated with the Keras model's `predict` function[4] and a 20 segment cross-validation. Based on the individual model accuracy evaluation, the DNNs were weighted to give 80% of the classification, while the CNNs contributed 10% each in case of the all-data-trained model. These values were halved when calculating the final value of the static- and dynamic-data-trained models.

The evaluation showed that the overall accuracy for classifying if a situation is dangerous or not, or if it will be within the next 3 seconds (5 labels in total, see Fig. 1) was on average 54.73% for the ensemble consisting of the separately trained models and 45.8% for the ensemble consisting of the 3 models trained on both types of sample. Based on the calculated confusion matrices, the greatest confusion occurred when determining how much time was left until a collision, the safe state was recognised over 87% of the time on average (ranging between 87.04% and 99.63%). In addition, we achieved an accuracy of 50.12% when using CNNs whose hyperparameters had not been optimized to classify whether the colliding obstacle was static or dynamic.

5 Discussion

In this paper we propose a solution for using sensor and image data for recognizing and anticipating collisions during indoor robot navigation. We compared the performance of a model trained on collisions with both static and dynamic objects with one that combined networks trained separately on dynamic and

[4]Generates a numpy array consisting of output predictions for input samples. https://keras.io/models/model/#predict, visited: 30/01/2019

static obstacle collisions. The comparison showed that even with a small sample size, the split training outperforms the all-sample-based one. Moreover, it was observed that, within the weighted ensemble methods, the individual DNN models contributed with more accurate predictions than the CNNs. We expect that, with more samples, the DNNs could be reliably used for the recognition, with the CNNs supporting classification. However, based on the binary classification CNN created in addition, the CNNs should not be used for the recognition of danger, but only for discriminating between static and dynamic obstacles which are about to collide with the robot. This approach may mean that less data is required for training danger recognition, as the proposed DNNs, unlike the image-based models, did not require a large dataset to achieve a reasonable danger recognition accuracy.

References

[1] Mohammed Faisal, Khalid Al-Mutib, Ramdane Hedjar, Hassan Mathkour, Mansour Alsulaiman, and Ebrahim Mattar. Multi Modules Fuzzy Logic for Mobile Robots Navigation and Obstacle Avoidance in Unknown Indoor Dynamic Environment. page 9, 2013.

[2] Amir Nasrinahar and Joon Huang Chuah. Intelligent motion planning of a mobile robot with dynamic obstacle avoidance. *Journal on Vehicle Routing Algorithms*, January 2018.

[3] Javier Minguez and L. Montano. Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios. *IEEE Transactions on Robotics and Automation*, 20(1):45–59, February 2004.

[4] Frédéric Large, Christian Laugier, and Zvi Shiller. Navigation Among Moving Obstacles Using the NLVO: Principles and Applications to Intelligent Vehicles. *Autonomous Robots*, 19(2):159–171, 2005.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC4*, 4(2):100–107, 1968.

[6] Yann LeCun, Urs Muller, Jan Ben, Eric Cosatto, and Beat Flepp. Off-road Obstacle Avoidance Through End-to-end Learning. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 739–746, Cambridge, MA, USA, 2005. MIT Press.

[7] X. Wang, V. Yadav, and S. N. Balakrishnan. Cooperative UAV Formation Flying With Obstacle/Collision Avoidance. *IEEE Transactions on Control Systems Technology*, 15(4):672–679, July 2007.

[8] Stephen Grossberg. Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks*, 1(1):17–61, 1988.

[9] Dhiraj Gandhi, Lerrel Pinto, and Abhinav Gupta. Learning to Fly by Crashing. *arXiv:1704.05588 [cs]*, April 2017. arXiv: 1704.05588.

[10] Moritz B. Milde, Hermann Blum, Alexander Dietmüller, Dora Sumislawska, Jörg Conradt, Giacomo Indiveri, and Yulia Sandamirskaya. Obstacle Avoidance and Target Acquisition for Robot Navigation Using a Mixed Signal Analog/Digital Neuromorphic Processing System. *Frontiers in Neurorobotics*, 11, 2017.

[11] Jiangliang Huang, Derong Tan, Liang Sun, Jinju Shao, Yaojuan Ma, and Zhangu Wang. Obstacle recognition in front of vehicle based on geometry information and corrected laser intensity. *Artificial Life and Robotics*, 23(3):338–344, September 2018.

[12] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to End Learning for Self-Driving Cars. *arXiv:1604.07316 [cs]*, April 2016. arXiv: 1604.07316.