2020 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics (AIM), Boston, USA
(Virtual Conference), July 6-9, 2020

Training End-to-End Steering of a Self-Balancing Mobile Robot Based on RGB-D Image and Deep ConvNet

Chih-Hung G. Li, Member, IEEE, and Long-Ping Zhou

***Abstract*—In an attempt to build a self-balancing mobile robot, an end-to-end autonomous steering system was proposed based on RGB-D deep learning. An RGB-D camera is installed on the mobile robot to capture real-time depth images of the front environment. By annotating the depth images with the steering angles, a deep convolutional neural network was trained to provide end-to-end steering commands for direction control. The self-balancing mobile robot was built on a commercial self-balancing mobile platform; a data collection scooter was built on the same mobile platform and operated by a human rider to reflect the social-aware behavior. Two types of navigation tasks – corridor cornering and path adjustment were devised and tested. A preliminary performance study was also reported.**

I. Introduction

A mobile service robot navigates its work environment and provides services such as delivery, surveillance, tour-guiding, object-handling, etc. We built a mobile robot on a commercialized self-balancing two-wheeled scooter – Ninebot Mini Pro, which will be referred to as J4.α in this article. To devise the navigation system of J4.α, we divide the system into two parts, a localization module through visual place recognition and a self-driving module through visual direction control. The localization module was reported in [1]; in this paper, we report the self-driving module developed for J4.α. Specifically, a deep learning scheme was adopted to train the robot to properly react to the depth images acquired in real time and control the handlebar for proper steering. When J4.α navigates along indoor corridors, the self-driving system guides the robot through the path without bumping into the wall. Along with the localization module, J4.α can thus perform autonomous navigation from one location to a designated location according to the topological nodal structure depicted in [1].

An RGB-D camera is installed in front of J4.α, 29 cm from the ground, and is used to continuously inspect the environment in front. The depth information of various corridor scenes in relation to a human rider's response is collected to form the training set for the control model that determines the handlebar angle. The relatively vacant environments allow us to focus on the static objects and neglect the passing pedestrians. To collect the training images, a self-balancing scooter of the same kind was modified to install the RGB-D camera and three rotary encoders that monitor the handlebar angle and the rotation of the two wheels. As the human rider rides the vehicle along indoor corridors, the depth images are recorded along with the rider's control actions. We then annotate the depth images with the corresponding handlebar angles. A deep convolutional neural network (ConvNet) is used to train the image input for the angle output end-to-end. As shown in Figure 1, during operation, the robot acquires real-time depth information by the RGB-D camera, the on-board PC executes the ConvNet calculation and outputs the detection result, and the handlebar angle is driven accordingly by a stepper motor. The main contributions of this work are three folds. First, an end-to-end deep learning visual steering strategy was proposed for a self-balancing mobile robot. Secondly, the deep learning scheme allows the system to imitate a human rider's social-aware reactions in indoor corridor environments. Thirdly, we demonstrate the scheme on two types of navigation tasks; preliminary performance study was also reported.

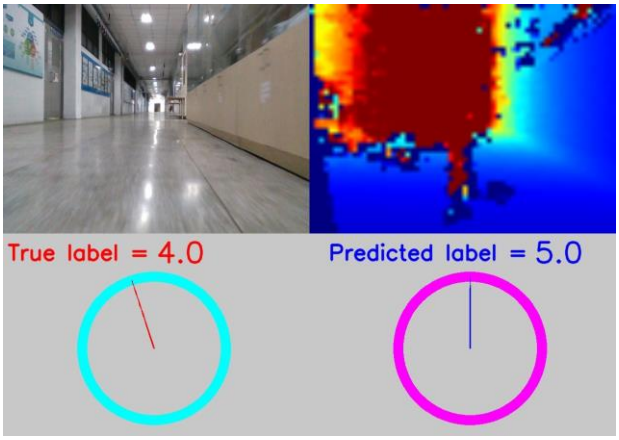


Fig. 1 The proposed end-to-end steering system for a self-balancing mobile robot receives RGB-D inputs of the front view and predicts handlebar angle by a deep ConvNet.

II. Related Work

A. RGB-D Vision

RGB-D cameras capture RGB images along with per-pixel depth information. Applications of the RGB-D camera for detection of the depth information around the robot have been reported previously. Researchers used RGB-D vision for provision of depth information of the targeted objects [2] to perform automated manipulator tasks such as object grasping [3] and clothes handling [4]. Henry et al. [5] introduced RGB-D Mapping, which utilizes RGB-D cameras to generate dense 3-D models of indoor environments. They have concluded

*This work was supported by the National Taipei University of Technology – King Mongkut's University of Technology Thonburi Joint Research Program: NTUT-KMUTT-107-02.

Chih-Hung G. Li and Long-Ping Zhou are with the Graduate Institute of Manufacturing Technology, National Taipei University of Technology, Taipei, 10608 Taiwan ROC (e-mail: cL4e@ ntut.edu.tw).

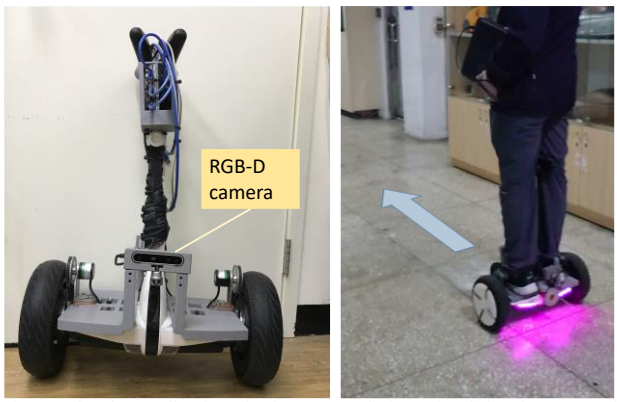


Fig. 2. The data collection rig for accumulation of training images and vehicle signals. An RGB-D camera is installed in front of the scooter at 29 cm from the ground.

that it would be feasible to build complete robot navigation and interaction systems solely based on inexpensive RGB-D cameras [5].

In the context of simultaneous localization and mapping (SLAM), Endres et al. [6] present an approach for concurrent estimation of the camera trajectory and generation of a dense 3D model of the environment; their system can guide a quadrotor through a cafeteria avoiding obstacles such as walls and furniture by a single RGB camera and a trained depth-aware ConvNet. However, the performance near the corners and clear glass windows still needs to be improved. In this study, we link the raw depth input provided by an RGB-D camera directly to the angle output of the handlebar that controls the motion direction of the robot rather than building a 3-D map of the environment being explored.

B. *Deep ConvNet*

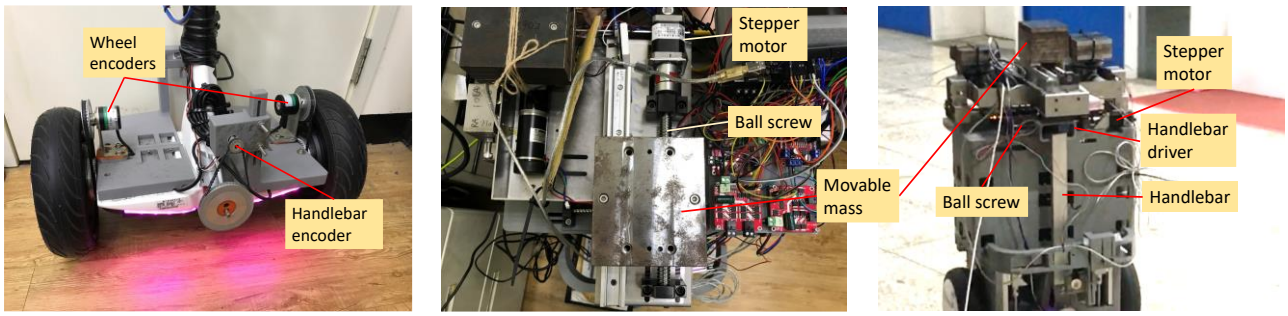
Back in the 90's, Artificial Neural Network (ANN) was already presented as an effective framework for end-to-end training of pixel inputs to steering outputs [7], e.g., Pomerleau's system ALVINN receiving inputs from the pixels in the image and outputting 30 steering options for a self-driving vehicle on public highways [8]. Ever since the success in ImageNet Competition by Krizhevsky et al. in 2012 [9], deep ConvNet has received explosive attentions and been adopted for feature extraction in various jobs such as place recognition [10] and object detection [11]. Tai et al. [12] demonstrated end-to-end training of RGB-D pixels to steering commands of a 3-wheeled mobile robot on a ConvNet fused with the decision process. Five directional control commands are generated as network outputs for a TurtleBot [13] navigation task in a corridor environment. The Microsoft Kinect they used provided a sensing range of 0.8 m to 4 m; with a laptop PC without GPU, their system could complete every visual-command cycle in 247 ms. Their test results show high similarity between robot decisions and human decisions with an overall accuracy of 80.2%; in all trails, the robot did not collide with obstacles.

C. *Autonomous Driving*

To some extent, the mobility characteristics of J4.α are more like street vehicles, although in this study, the target environment where it is operated in are much different from open roads. Nevertheless, experiences in the development of autonomous driving are worth learning. Chen et al. [14] summarized two approaches in autonomous driving – the mediated perception approach [15][16] and the behavior reflex approach [8][12][17]; they proposed a third approach of direct perception, in which a mapping is learned from an image to several meaningful affordance indicators, instead of direct steering angles. They argued that by this way, over representation by the mediated perception approach can be avoided, and the drawbacks of low-level decision-making of the behavior reflex approach can be remedied. Admitting that the behavior reflex approach we adopted here may fall in the problems such as same-image-multiple-choice and low-level decision-making, we divide navigation into more focused and specific tasks such as cornering, path adjustment, obstacle avoidance, etc. In our proposed scheme, these tasks will be trained separately and later be administered jointly under a hierarchical control structure. A similar strategy was demonstrated by [18], which effectively trained socially compliant navigation through raw depth inputs.

D. *Robotic Control Policy*

In the work of Muller et al. for off-road obstacle avoidance [17], end-to-end training from stereo pixels to steering angles on a ConvNet resulted in an error rate of 35.8% on the test set.



(a) (b) (c)

Fig. 3. (a) Encoders for monitoring the handlebar angle and the wheel speed, (b) movable mass and the driving mechanism that provide the forward/backward thrust for the robot; (c) steering mechanism of J4.α including the handlebar, the handlebar driver, a ball screw, and a stepper motor that drives the ball screw and moves the handlebar.

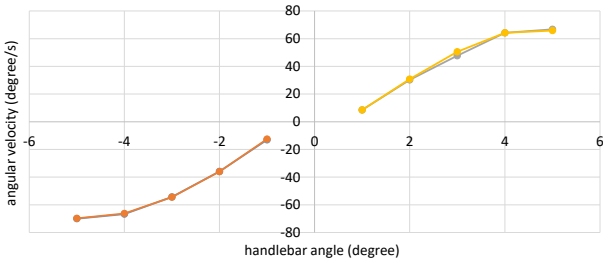


Fig. 4. Measured relationship between the handlebar angle and the angular velocity of the scooter. Within $\pm 1°$, the vehicle does not generate a turning response.

However, the seemingly high error rate does not reflect the fact that mis-avoidance of obstacles by the vehicle is rare. Whereas the error rate presents the deviation between the human labels and the output bins, the mechanical system has a large space for tolerance that allows for completion of the targeted task. Thus, a sophisticated control policy was not needed in such an outdoor off-road task. It is not difficult to imagine that for other precise robotic manipulations, the control policy becomes relatively important. In an attempt of learning robotic control policies, Levine et al. [19] presented a method that represents the policy by a deep ConvNet, which can be trained end-to-end using a guided policy search algorithm.

III. The Proposed System

A. Sensing Units

J4.α reaches a maximum speed of 4.4 m/s, and is commonly operated at a speed of 1 – 2 m/s, which is much higher than that of TurtleBot (0.26 m/s) used in [12]. The higher operation speed makes it necessary for J4.α to see farther in order to react promptly. An RGB-D camera – Intel RealSense D415 was installed in front of the robot, which detects a minimum depth distance of 0.3 m and a maximum range of approximately 20 m [20]. The depth camera is capable of providing a resolution up to 1280×720 pixels and a frame rate up to 90 fps; here we chose 640×480 for output. In order to record the depth images with the control actions of the human operator, we transformed one of the scooters into a data collection rig as shown in Figure 2, in which the depth camera is installed at the same position as in the robot, and a rotary encoder is installed at the handlebar pivot to track its rotating angle. When the operator rides the scooter, the front views associated with the handlebar angles are recorded synchronously. As shown in Figure 3 (a), two extra rotary encoders were adopted to monitor the revolution of the wheels, which can be used to track the linear and rotating speeds of the vehicle.

B. Steering Mechanism

The forward/backward motion of the self-balancing scooter is controlled by the forward/backward tendency of the rider on the vehicle. When a tilting angle is detected by the scooter, it automatically accelerates in the same direction in order to maintain force equilibrium and prevent the scooter from tipping over. To imitate the COG (center of gravity) control of a human rider, a movable-mass system was designed for J4.α, which effectively shifts the overall COG and changes the forward/backward tendency of the robot. As shown in Figure 3 (b), steel blocks were adopted as the movable mass, which is supported by a linear slide and a ball screw driven by a stepper motor. Motion patterns of J4.α have been studied systematically; the results will be reported in a different article to be published.

The heading direction of the scooter is controlled through a handlebar, which can be pushed by the rider's thigh and rotate by a small angle. The handlebar angle is related to the angular velocity of the changing heading direction in a fashion that a larger angle results in a higher angular velocity as shown in Figure 4. Our measurement reveals that the system does not generate a response when the handlebar angle is within $\pm 1°$; outside the range, the angular velocity of the scooter is essentially proportional to the handlebar angle until $\pm 4°$. Upon further increase of the handlebar angle, the angular velocity is not increased as much as that by smaller angles.

On J4.α, we designed a handlebar driving system that propels the handlebar as needed. As shown in Figure 3 (c), the handlebar driver moves horizontally on a ball screw, powered by a stepper motor. Thus, by commanding the pulse count of the stepper motor, the position of the handlebar driver as well as the handlebar angle can be effectively controlled.

C. Task Formulation

As briefly described in II *C*, the vision-guided navigation task can be divided into specific tasks such as corridor cornering, path adjustment, obstacle avoidance, etc. Each task can be trained separately; in operation, only a task is activated at a time. Thus, within a specific task, the visual input is only related to the decision making in that specific task. For example, within the task of corridor cornering, the depth images (what the robot sees) only correlate to the actions that aim to complete the task of cornering. For any specific navigation task, the duration can be discretized into a finite number of steps N. At each time step n, the agent is given a set of sensory data which is the front depth image. These data construct the input $I_n(o_n)$ to the policy function $\pi_n(u_n|I_n)$, which will recommend an action u_n. In this Markov process, the probability of the next step is based on the current state and the action being taken as $p_n(x_{n+1}|x_n, u_n)$. The trajectory τ of a navigation task is defined as a series of states and actions,

$$\tau = \{x_1, u_1, x_2, u_2, \dots, x_n, u_n, \dots, x_N, u_N\} \quad (1)$$

The policy that completes a navigation task can be formulated as,

$$\pi(\tau) = p(x_1) \prod_{n=1}^{N} p(x_{n+1}|x_n, u_n) \int \pi(u_n|I_n) p(I_n|x_n) dI_n \quad (2)$$

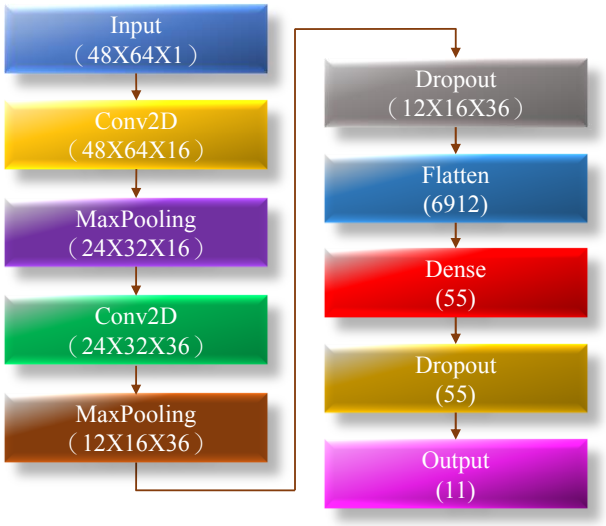


Fig. 5. Architecture of the ConvNet.

In our scheme, the policy function can be further divided into two parts, a deep ConvNet Γ that classifies the depth input, and an action maker $\tilde{\pi}$ that decides on the ConvNet output of ω_n; jointly the policy function is,

$$\pi(u_n|I_n) = \tilde{\pi}(u_n|\omega_n)\Gamma(\omega_n|I_n) \quad (3)$$

D. Convolutional Neural Network

The initial 640 × 480 depth image captured by the RGB-D camera is resized to 64 × 48 before entering ConvNet. The array covers a depth range between 0 m and 20 m. We truncated the maximum depth at 15 m and normalized the results to between 0 and 1. Every image was then annotated with the corresponding handlebar angle, which is divided into 11 categories that represent $-5°, -4°, \ldots, 0°, \ldots, 4°$, and $5°$. The result is used to train a ConvNet whose architecture is shown in Figure 5. Training of the ConvNet was executed on a PC with an Intel i7-7700HQ CPU, 8GB DDR4 RAM, and NVIDIA GeForce GTX 1050 with 2GB DDR5.

IV. Experiments

Two types of navigation tasks were devised, type 1 – corridor cornering and type 2 – path adjustment. Both are assumed in an indoor corridor environment; as depicted in Figure 6, the robot is trained to follow type 1 while making left or right turns at the corner of a corridor, and to follow type 2 for adjusting its direction to be parallel to the wall. For both type 1 and type 2, the policy functions endowed by the human trainer inherit the right-passing social awareness [21], as the training data were generated under human social awareness.

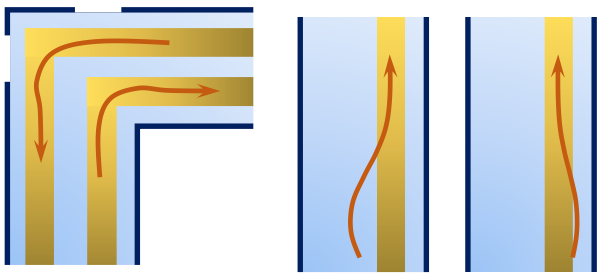


Fig. 6. Two types of the navigation tasks devised and tested, *left*: type 1 – corridor cornering; *right*: type 2 – path adjustment. The robot is trained to follow the arrows and move to the target paths (yellow bands).

To generate the two policy functions, a human operator rode the data collection rig to collect training data. At various sections of the corridor, the operator performed various maneuvers similar to the arrows shown in Figure 6 into the yellow bands indicating the target paths. A task is defined between the beginning and the end of the arrow; data outside the tasks are discarded. The goal was to train the robot to generate similar maneuvers like the arrows while confronting similar front depth views. The corridors used for data collection are inside a university campus building with an average width of approximately 3 m. The duration of each task, namely the length of the arrow is approximately 5 m for both types. The location of the target paths is approximately 1 m from the right wall. Data sampling was executed at a frequency of 16 Hz, which was also the frequency for detection at the later stage. The entire data collection was completed by a single operator in order to maintain the consistency in behavior and data characteristics.

A. Corridor Cornering

Totally, 110k sequence images of type 1 were collected, equivalent to approximately 1000 cornering tasks. Note that in order to maintain an equal number for every detection category, lots of small angle images were discarded, as in a typical task sequence, much more images were associated with small angles. The 110k is the result after removing the excess of small angle images. Training of 110k images took around 150 min.

1) ConvNet prediction

A training accuracy of 97% was obtained for ConvNet training. We then tested the type 1 ConvNet model by a series

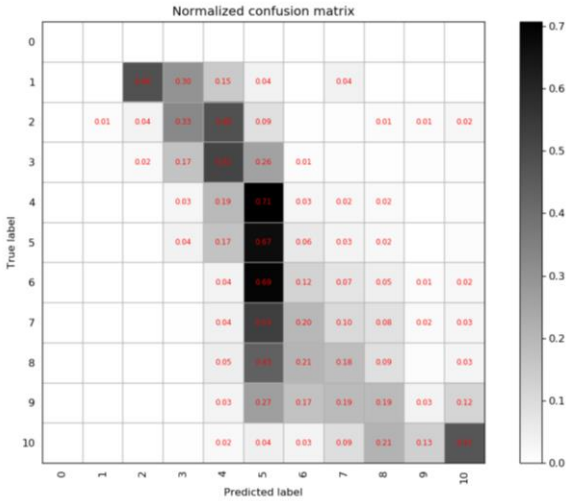


Fig. 7 Confusion matrix of type 1 detection – corridor cornering.

of testing sequences, which includes 19k images and approximately 211 tasks. The results are shown in the form of confusion matrix in Figure 7. The overall accuracy is 35%; however, as a dark diagonal band can be witnessed, predictions of opposite turning directions are rare.

2) System performance

Preliminary evaluation of the system performance was performed by estimating the history of the robot heading direction, assuming the handlebar is controlled by the ConvNet predictions. By integrating the turning speed $\beta(\theta)$ of the vehicle over time while considering the time series of the handlebar predictions θ_n, the history with an initial β_0 and a final heading direction β_T can be obtained as,

$$\beta_T = \beta_0 + \int_0^T \beta(\theta)\, dt \cong \beta_0 + \sum_{n=1}^{n=N} \beta(\theta_n)\Delta t \qquad (4)$$

In Figure 8, estimation for a typical type 1 task was shown; the estimate of the final heading angle arrived at 83° is fairly close to the ground truth of 87°.

B. Path Adjustment

110k sequence images were also collected for type 2 training, equivalent to approximately 1000 path adjustment tasks. Analogously, lots of small angle images were discarded; 110k is the final data size.

1) ConvNet prediction

To test the type 2 ConvNet, 22k images equivalent to 244 tasks were used; the resulting confusion matrix is shown in Figure 9. The overall accuracy is 23%; however, a similar dark diagonal band on the confusion matrix is witnessed. It is worthy to note that both results showed a tendency of underestimate on the handlebar angle, i.e., labels 0 and 1 are more likely to be predicted as 2 or 3, and labels 10 and 9 are predicted as 8 or 7. However, predictions of opposite turning directions are still rare.

2) System performance

The history of heading direction was also estimated following the treatment of (4). It was found that some test cases showed close resemblance between the estimate and the ground truth, while the others exhibit larger discrepancies in the final angles, despite the motion patterns being similar. The result confirms that the handlebar angle determining the heading direction of the robot can be trained from raw pixels of the front depth view. However, accurate maneuver may require further improvement on the ConvNet prediction accuracy.

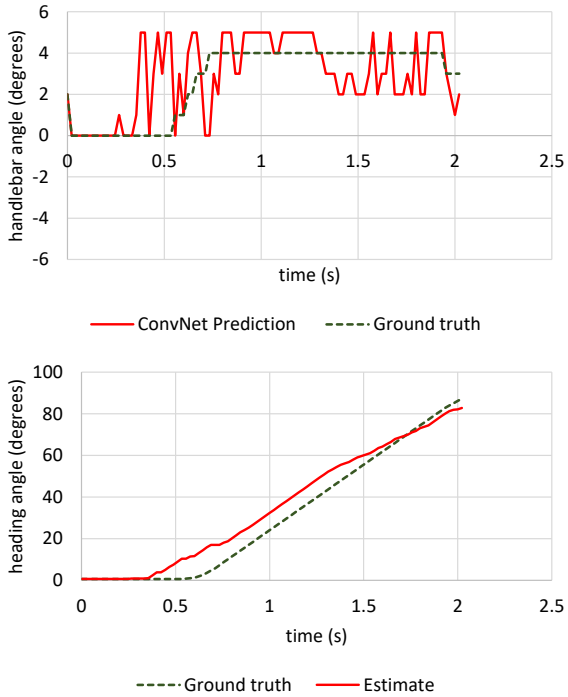


Fig. 8. Comparisons of the ground truth and prediction of a type 1 task (turning right), *top*: the handlebar angle; *bottom*: the heading angle. The ground truth turns right by 87° and the prediction turns right by 83°.

V. Discussion and Conclusion

In this paper, we proposed a control scheme for end-to-end steering of a self-balancing mobile robot based on RGB-D input for direct steering output. Convolutional neural network was adopted to learn from depth pixels and to output 11 categories of the handlebar angle. A human operator rode the data collection rig to capture the front depth images while the handlebar angles are recorded synchronously. The data essentially reflect some social-aware behaviors on operating a vehicle in indoor corridor environments. Two types of navigation tasks were devised and tested – corridor cornering and path adjustment. 110k sequence images were collected for ConvNet training of each type, equivalent to approximately 1000 tasks each. For ConvNet testing, around 20k sequence images were used; confusion matrices were obtained. It was found that larger handlebar angles are likely to be underestimated; however, predictions of opposite turning directions are rare. In practice, a large handlebar angle

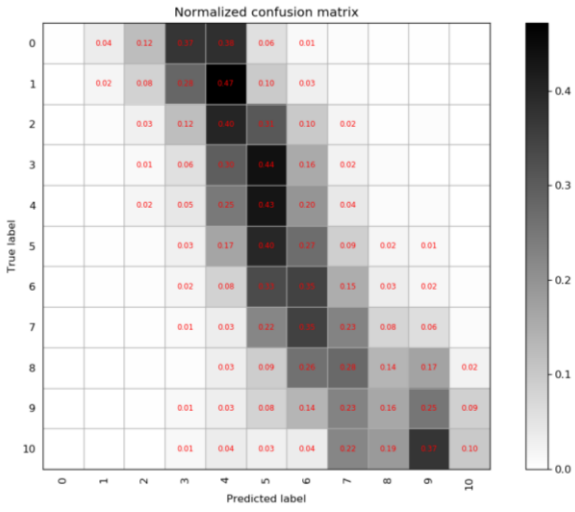


Fig. 9. Confusion matrix of type 2 detection – path adjustment.

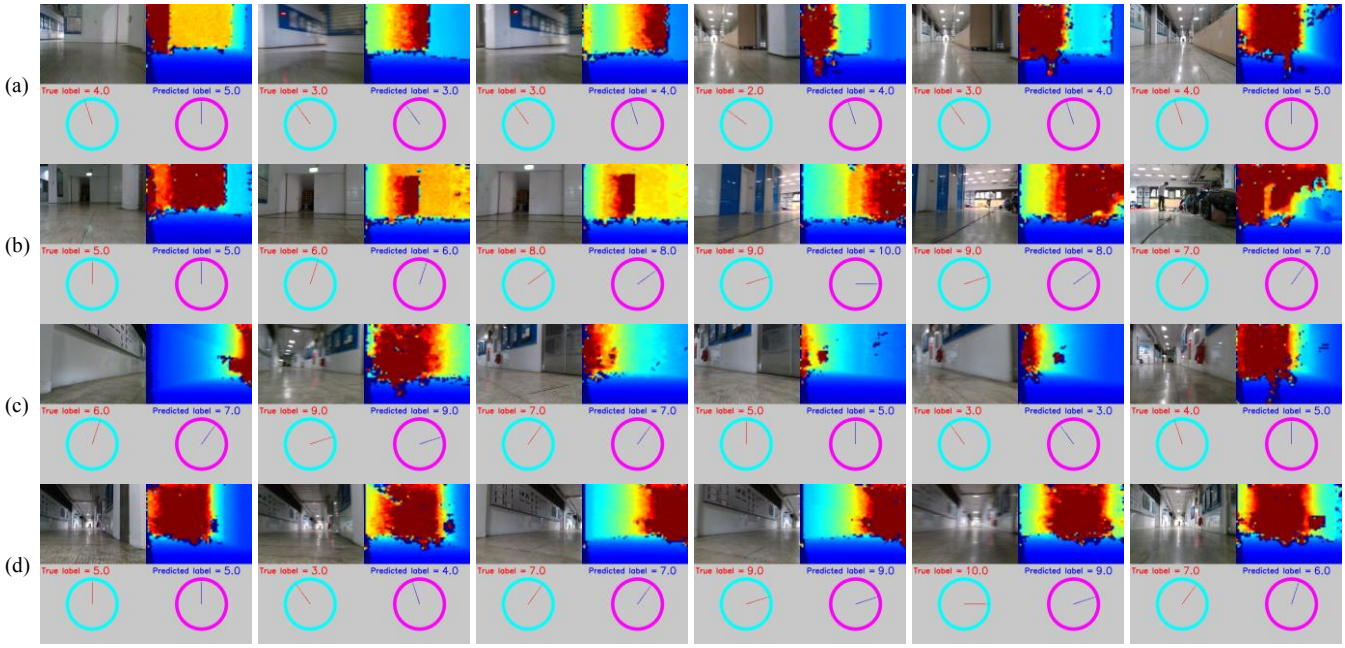


Fig. 10. Sample sequences (left to right) of the consecutive images of various scenarios: (a) Type 1 – left turn, (b) Type 1 – right turn, (c) Type 2 – moving to right; (d) Type 2 – avoiding right wall. https://youtu.be/a481aVdBkJk

has much lower occurrence; thus, underestimate of the handlebar angle may not be a devastating system pitfall. By integrating the turning speed based on the predicted handlebar angles, the history of heading direction in a task was estimated and compared with the ground truth. Some test cases showed close resemblance between the estimate and the ground truth, while the others exhibited similar motion trends but larger discrepancies. The overall performance including the mechanical response is still under investigation.

REFERENCES

[1] Y. Hong, Y. Chang, C. G. Li, "Real-time visual-based localization for mobile robot using structured-view deep learning," in *Proc. 2019 IEEE Int. Conf. Auto. Sci. Eng. (CASE)*, pp. 1353–1358, 2019.

[2] K. Lai, L. Bo, X. Ren, D. Fox, "A Large-Scale Hierarchical Multi-View RGB-D Object Dataset," in *Proc. 2011 IEEE Int. Conf. Robotics Auto. (ICRA)*, pp. 1817–1824, 2011.

[3] R. Detry, C. H. Ek, M. Madry, D. Kragić, "Learning a Dictionary of Prototypical Grasp-predicting Parts from Grasping Experience," in *Proc. 2013 IEEE Int. Conf. Robotics Auto. (ICRA)*, pp. 601–608, 2013.

[4] L. Twardon, H. Ritter, "Interaction Skills for a Coat-Check Robot: Identifying and Handling the Boundary Components of Clothes," in *Proc. 2015 IEEE Int. Conf. Robotics Auto. (ICRA)*, pp. 3682–3688, 2015.

[5] P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *Int. J. Robotics Res.*, vol. 31, no. 5, pp. 647–663, 2012.

[6] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard, "An Evaluation of the RGB-D SLAM System," in *Proc. 2012 IEEE Int. Conf. Robotics Auto. (ICRA)*, pp. 1691–1696, 2012.

[7] T. Mitchell, *Machine Learning*, ISBN-13: 978-0070428072, McGraw-Hill, 1997.

[8] D. A. Pomerleau, "Knowledge-based Training of Artificial Neural Networks for Autonomous Robot Driving," in *Robot Learning. The Springer International Series in Engineering and Computer Science*, vol. 233, pp. 19–43. Springer, 1993.

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 1097–1105, 2012.

[10] N. Sunderhauf, S. Shirazi, F. Dayoub, B. Upcroft, M. Milford, "On the performance of ConvNet features for place recognition," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, pp. 4297–4304, 2015.

[11] M. Oquab, L. Bottou, I. Laptev, J. Sivic, "Is object localization for free? - weakly-supervised learning with Convolutional Neural Networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recog. (CVPR)*, pp. 685–694, 2015.

[12] L. Tai, S. Li, M. Liu, "A Deep-network solution towards model-less obstacle avoidance," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, pp. 2759–2764, 2016.

[13] ROBOTIS e-manual http://emanual.robotis.com/docs/en/platform/turtlebot3/specifications / last accessed June 10, 2019.

[14] C. Chen, A. Seff, A. Kornhauser, J. Xiao, "DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving," in *Proc. Int. Conf. Comput. Vision (ICCV)*, pp. 2722–2730, 2015.

[15] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. 2008 IEEE Intell. Vehicles Symposium*, pp. 7–12, 2008.

[16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, "Object detection with discriminatively trained partbased models,". *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, 2010.

[17] U. Muller, J. Ben, E. Cosatto, B. Flepp, Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning," in *Adv. Neural Inf. Process. Syst. (NIPS)*, pp. 739–746, 2005.

[18] L. Tai, J. Zhang, M. Liu, W. Burgard, "Socially Compliant Navigation through Raw Depth Inputs with Generative Adversarial Imitation Learning," *Proc. 2018 IEEE Int. Conf. Robotics Auto. (ICRA)*, pp. 1111–1117, 2018.

[19] S. Levine, C. Finn, T. Darrell, P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learning Res.*, vol. 17, pp. 1–40, 2016.

[20] Intel RealSense Depth Camera D415, https://click.intel.com/intelr-realsensetm-depth-camera-d415.html. last accessed Jun 5, 2019.

[21] Y. F. Chen, M. Everett, M. Liu, J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, pp. 1343–1350, 2017.