Measuring Unsupervised-Supervised Real-Life Training of an Autonomous Model Car

Sascha Hornauer¹, Karl Zipser² and Stella Yu³

I. ABSTRACT

In autonomous driving, imitation learning of driving behavior is an active area of research as it enables generalization to many different environments where current handcrafted approaches are limited in their application. Steering and motor commands are trained i.e. as an output of a deep neural network, based on expert examples. However, this approach makes it often hard to judge the actual real-life performance, beyond loss-function and validation values. Current used metrics are often weakly expressive as they count the number of crashes or human interventions while driving [1]. Furthermore, by learning from the example of a human expert, the performance of the human instructor will heavily determine the performance of driving. We propose a novel way to train obstacle avoidance and navigation using a multi-task network with minimal human interaction and show a detailed way of measuring the actual driving performance of the network after training. A path planner with perfect information about the environment trains a model-car to act on vision input alone to navigate in a narrow office space. For different sub-spaces of that scenario, where the same trajectories have to be driven by the car in each run, an in-depth analysis of the trajectories before and after training is performed. This makes it possible to compare the actual driving behavior to training factors as the distribution of the input data and values of the loss- and validation-function in great detail. Furthermore, it is possible to measure how well chosen parameters of the desired behavior, as set in the path planner, are preserved in the resulting trajectories.

II. INTRODUCTION

In academia and industry, research on autonomous driving aims to solve two problems whose current solutions seem to impede each other. On one hand, the goal is to produce safe, predictable and deterministic behaviors to adhere to safety standards. On the other hand, the goal is to handle all possible, sometimes rarely encountered driving situations with suitable, purposefully generalized actions [2].

A. Measuring Robustness and Reliability

One of the major concerns for safety is the robustness and reproducibility of learned behavior in End-to-End learning. Approaches which classify the environment and afterwards execute hard-coded actions, dependent on a detected state, make the possible actions predictable, though at the cost of having to hand-craft the action sequences [3]. In our approach we allow for an almost continuous choice of steering angles and motor speeds, which makes it hard to predict exact driving behavior.

Several metrics to measure autonomous driving performance have been used in the research community, while safety requirements in the automotive industry call for more stringent measures. The ISO 26262 standard requires a level of functional safety for electrical and electronic systems within an car [2]. As part of the risk analysis of each electric and electronic system, Automotive Safety Integrity Level (ASILs) are defined based on the severity, exposure and controllability of a possible safety hazard. When using fully connected neural networks for direct End-to-End driving it is equally hard assessing those factors as tracing the development of the neural network based algorithm. Even though a new standard in development, the ISO/AWI PAS 21448, might accept for some ADAS applications systems, which are not entirely fault-free [4] though assessing, understanding and interpreting the performance of neural networks will call for further metrics than those used in research today. Even though in imitation learning existing functions can measure how close the output of a network is to state-action examples given by an expert, it is still unclear how well trained networks generalize to new situations when deployed.

B. Learning General Navigation

Current research appreciates the need for neural nets to handle complex sensory inputs in real driving situations, [5]. Especially in situations where skills for general navigation are needed yet hard to express in formal algorithmic ways, convolutional neural nets (CNNs) are trained to imitate human behaviors. It is essential for imitation learning to collect data which reflects good human driving behavior while ideally human errors should be excluded [6]. Our preliminary work on imitation learning of steering and motor signals of model cars shows that we could train a shallow CNN to learn navigation in rough terrains, in snow conditions and even at night. However, collecting the needed data is not only time-consuming but it is also hard to ensure exhaustive collection of all corner-cases. In order to learn general navigation capabilities and obstacle avoidance specifically we train a Multi-Task network to drive End-to-End to map from incoming images directly to steering and motor commands. Our main contribution is to train in a realistic and measurable way, to make the correlations in between training data and resulting driving behavior more transparent. We propose a novel approach training the network automatically by a
path planner in a real environment and propose a way of 
measuring and interpreting the behavior of the network in 
real life.

III. RELATED WORK

[7] uses a path planner in a simulation to navigate to 
a number of goals with several obstacles. They fused an 
external goal position and extracted features from a con-
volutional neural network into their final fully connected layers 
to produce steering and motor commands towards that goal. 
A motion planner is also used as expert to train the network 
though this is performed in a deterministic simulation.

End-to-end learning was successfully used in different 
domains as driving tasks [1] and robotic manipulation [8], [9] 
designed a learning architecture for a robot in a simulation, 
which builds a cognitive abstract map and plans actions in 
that map at the same time. The map is built and updated 
from vision and knowledge about the robot’s own motion. 
In their work they exploit the fact that a turn of the own 
robot will lead to a known rotation in their derived map 
and consequently they rotate their belief about the world 
depending on the egomotion. Successful navigation and also 
semantic task performance, as finding an office chair, could 
be shown in simulation though the application is limited by 
the assumption of perfect odometry.

IV. PRELIMINARY WORK

A testbed of several autonomous model cars has been built. 
Fig. 2 shows our model car scenario, where an ultrasonic 
based system provides ground truth localization. Several car 
sensor configurations were tested in the past as 2D Lidar, 
an inertial measurement unit, a stereo camera and a GPS. 
The current version of our neural net is able to learn and 
reproduce generic driving from stereo vision alone while 
avoiding other cars successfully.

V. METHOD

Opposed to [7] we see advantages in real world training: 
The state progression of a model car while driving is proba-
bilistic allowing for a natural exploration of the state space. It 
is not necessary to add artificial noise to the control or input 
signal as when wanting to explore a larger state-space in a 
simulation. Using a path planner in this real-world setting

Fig. 2: Sketch of scenario with to emergent behaviors, full 
turning along the room a) and complex turning instead of 
simple steering b).

still minimizes the need for human intervention. In order to 
ev even improve state-space exploration the path planner can be 
used to track the network task execution and intervene only 
to avert a collision, similar to the technique used in [1].

A. Ground-Truth Dataset

A dataset for training was collected in a fixed scenario with 
one car and two objects with printed patterns attached, which 
makes them easily distinguishable against the background 
from the front-camera of the model car. They form a gate 
in the middle of a room which is passed in several runs at 
random either on the left or right side or in the center. One 
fixed center point on the ground at a distance of the gate is 
chosen from where the car starts to ensure the patterns are 
always in the cameras field of view before it chooses one 
of the goals. The exact trajectory is recorded based on an 
ultrasonic localization system with an accuracy of 2 ± cm, 
as used in [10]. In addition to passing the gate, the cars are 
steered through the whole size of the room to train general 
navigation and add data of the environment around the gate.

The original behavior was trained into the network with 
perfect knowledge about the environment and the car, based 
on the localization system and a model-based path planner. 
This behavior was tracked and the average trajectories while 
going through each gate were measured. After several stages 
of network training, an execution phase was conducted to 
observe the networks performance. In this phase a human 
instructor or another higher-level planner signals which lane 
the car should take. The network is designed with an addi-
tional input to receive the desired lane as meta-information 
during training and when the network is driving. This aims 
to make the driving behavior of the End-to-End trained car

Fig. 1: Two cars of our model car platform with visible ZED-
camera.
selectable during network execution.

B. Path Planner

Path planning is performed based on a localization system and a fixed map. The approach aims to train the network to reproduce the input data based on its vision system alone, without external information apart from a desired lane. The trajectories can be parameterized in detail to set i.e. turning radius, distance to obstacles and the boundaries of the map to match the manoeuvrability of the used car. The model was trained in a narrow though not too complex space to be able to trace in detail the differences of the produced trajectories, resulting from driving under network control, to the path planner trajectories, used at training time. It is based on a Dubins car model with a chosen turning radius of 1.2 meters and plans a feasible path to one of three positions: left of, right of and in the center of the gate, as seen in figure fig. 1. Once one of these positions is reached there are further positions along the room which are planned depending on the chosen lane. After two turns the car steers back to the original point. The exact trajectory towards the planned path is tracked with a simple PID-controller and additional correction via the cross-track error. The system failed only when parts of the localization system failed.

C. Multi-Task Network

In robot manipulation tasks, [11] were able to train a network with task-specific and robot-specific modules to enable transfer learning in between tasks. In our work we define going through the three different lanes as different tasks, and our research interest is to being able to choose in between tasks with the same trained network during the execution phase. The used network architecture is based upon SqueezeNet as developed by [12], a network designed for image classification which was adjusted with an additional meta-layer for task selection. The meta-layer, as seen in fig. 3, adds a variation of a one-hot vector after the first average pooling and is during training set to encode the destination lane. This supplies a more abstract notion of a goal to the network, relative to objects in the scene and avoids giving Cartesian or polar coordinates directly, which would bias any learned internal representation of the environment. At the last layer, a vector of steering and motor commands over 10 time points is predicted while only the command furthest in time will be used for steering. This is motivated by earlier results which indicate that driving behavior improves when not only commands for a fixed point in the future are predicted but also commands in smaller steps before [13]. It favours the prediction of entire trajectories over single points of control through the car.

D. Network Training

The network was trained over 13 epochs with approximately 4200 iterations at a batch size of 500. Each sample consists of two consecutive images, at two different points in time, 33ms apart. The network is trained to produce steering and motor signals as shown by the path planner, although in our first approach with this measuring technique the motor speed is fixed to ease comparison of the resulting trajectories. Information about the chosen lane by the path planner is input as a one-hot channel vector for each frame. Steering and motor commands are compared against the ground truth with a Mean Squared Error-loss. Learning is performed with an adaptive learning rate method, as described in [14], implemented in PyTorch.

VI. EXPERIMENTAL EVALUATION

In a first phase the path planner was used over several sessions to create the database. It was possible with minimal human intervention to create the data and the model car only had to be retrieved from a stuck state when the localization system failed to report the correct location and a collision with a wall occurred. In a second phase the path planner and localization system were turned off and the car was driven based on the learned network output, while saved weights from different epochs were loaded. It was clearly visible how, at different epochs, manoeuvres emerged and the general navigation became more proficient. At the same time it was visible exactly when obstacle avoidance and navigation failed.

A. Pre-Training Trajectory Driving

The following results are based on measured trajectories, created from driving using the path planner to steer the car. Even though trajectories were created which span the whole room to drive continuously through the gate, trajectories in fig. 4 are shown in the region where lane information
was available through the meta-layer. As sketched in figure 1, the path planner leads the car through one of the gates and makes a full turn along the side of the room. When going through the center gate, a complex turning behavior has to be performed, which is not easy for the car to learn. It has to perform a slight turn to one side and then towards the other side to overcome the limitations of its turning radius. No successful complex turning without the path planner could be observed which motivates changes in the network architecture in the future. Presumably constant scene understanding over a longer timespan is needed though apart from presenting input images at different timesteps to our used network, it has no capabilities to keep a consistent state over time yet.

Apart from being able to avoid the obstacles at the very beginning, one successful run consists of turning towards other side of the room and making a second turn to return to the original position, as shown in fig. 1 c). The original position is reached at different, well distributed angles as seen in fig. 4. Since during the time of approaching the gate the goal-lane is known to be either left, center or right, clustering of trajectories was performed depending on the lane. Distances and averages among trajectories of unequal length were calculated using Dynamic Time Warping (DTW).

B. Post-Training Network Driving

The model car was set to the origin point at the center of the room where it was led to by the path planner during training. During several runs of the experiment, it was manually rotated in the direction straight towards each of the lanes of the gate. For each orientation, each possible destination lane was given as an input at the meta-layer before the autonomous mode of the car was started. The aim was to give the signal to go to each lane from each of the different orientations. It was clearly seen that going through the right and the center lane was being trained in between 3-4 epochs though going left failed for unknown reasons and the network steered after 6 epochs the car almost always through the other lanes. Even though the network could be trained to performed better than a network without the meta-layer lane information, the most decisive factor for predicting the chosen lane was the original orientation.

Presumably, constant scene understanding over a longer timespan is needed though apart from presenting input images at different timesteps to our used network, it has no capabilities to keep a consistent state over time yet.

A. Early epochs 0-4

It could be observed that in early epochs, obstacles were hit in 10% of all trials even though it was possible in 73% to avoid the obstacles and complete one round through the space. It can be seen in fig. 5 that the distance from the wall, which is vertical at −0.5m at the x axis, is not always kept. When the model car passed through the center it was not able to navigate back since the complex turn was not yet learned.

D. Mid epochs 5-8

At the end of the 7th epoch, as seen in fig. 6 the car began to reproduce the complex turning behavior when going through the center lane, though not enough to turn. Collisions were still existing in 3% of all trials. Nevertheless, even though there is a strong bias towards the right lane, depending on the angle going through the left lane was stable and the full turn was performed in opposite direction.

E. Late epochs 9-13

In the end several effects emerged which we attribute to overfitting effects. As it can be seen in fig. 8 the validation value is unstable and rising for late epochs. This suggests that learning of the underlying tasks is reduced during training.
in favor of a better fit to the training data. In our driving experiments this is visible as in 20% of all trials the car collides with one of the obstacles. Furthermore, when trying to turn, in about half of the cases the car is not able to. However, only after the 10th epoch a more closer fit to the return behavior as explained in fig. 1 c) was observed.

Fig. 7: Trajectories during execution of epochs 9-13 with degrading quality. Collisions are rising, shown as trajectories going through a black circle.

F. Loss Functions

The loss and validation function show results which are backed from our findings in the experiments. After the 2nd epoch the loss function begins to plateau though the validation loss shows good results for the 2nd and the 10th epochs and bad training results for the 3rd and 11th epoch. As the complex behaviors as complex turning, obstacle avoidance and general navigation emerge and decay at specific epochs, further research is motivated how these behaviors relate to the validation loss.

VII. CONCLUSION AND FUTURE WORK

We were able to train a model car to navigate based on vision input along in a narrow office space using a modified version of SqueezeNet. Apart from investigating how Multi-Task Learning in the autonomous driving domain can be realized we used a path planner to train the desired behaviour in real life with several advantages. The path planner is able to steer the car based on perfect information and with minimal human intervention. The model of the car was considered by the path planner during training while a human expert would have to learn to navigate in the narrow space and show examples of good driving at the same time. It was possible to present consistent good driving examples while gathering diverse data from the inherent probabilistic nature of real driving. Furthermore, the localization system used for the path planner enabled us to measure the resulting trajectories in a second phase, when the network showed its performance at several stages of proficiency. This result can be directly used towards making the behavior of neural networks in autonomous driving more understandable and designing future metrics and risk assessment techniques.

In the future two different lines of research have to be followed. Even though task learning based on a meta-layer showed improved network training in earlier experiments, it was not yet possible to correlate with the actual driving behavior. Varied training and different architecture designs should be evaluated to incorporate more temporal information. Especially when returning after passing the gate in the center knowledge about the position of the car in the room would benefit in choosing an appropriate turning radius. Also, we want to extend the localization and training to several cars which will not only annotate their appearance in the input video data but also introduce moving obstacles. Furthermore we were able to gather a huge amount of data about the execution phase, which is now meta information about the dataset used. With statistical methods we aim to finding a more quantifiable relation in between different emergent behaviors and other data as the training and validation results though also in particular to the input data.
ACKNOWLEDGMENT

This research was supported by the sponsors of the Berkeley Deep Drive program (BDD). The authors gratefully acknowledge NVIDIA for their donation of the NVIDIA TX1 Developer Kits for this project. We would also like to thank our students Sauhaarda Chowdhuri and Tushar Pankaj for their help in assembling and maintaining the model car project infrastructure.

REFERENCES

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, “End to End Learning for Self-Driving Cars,” pp. 1–9, 2016.

[2] P. Koopman and M. Wagner, “Autonomous Vehicle Safety: An Interdisciplinary Challenge,” IEEE Intelligent Transportation Systems Magazine, vol. 9, no. 1, pp. 90–96, 2017.

[3] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, “DeepDriving: Learning affordance for direct perception in autonomous driving,” in Proceedings of the IEEE International Conference on Computer Vision, vol. 2015 Inter, pp. 2722–2730, 2015.

[4] F. Falcini, G. Lami, and A. M. Costanza, “Deep Learning in Automotive Software,” IEEE Software, vol. 34, no. 3, pp. 56–63, 2017.

[5] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue, F. Mujica, A. Coates, and A. Y. Ng, “An Empirical Evaluation of Deep Learning on Highway Driving,” arXiv, pp. 1–7, 2015.

[6] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, “Imitation Learning,” ACM Computing Surveys, vol. 50, no. 2, pp. 1–35, 2017.

[7] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, “From Perception to Decision : A Data-driven Approach to End-to-end Motion Planning for Autonomous Ground Robots,” in 2017 IEEE International Conference on Robotics and Automation (ICRA), (Singapore, Singapore), pp. 1527–1533, IEEE, 2017.

[8] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-End Training of Deep Visuomotor Policies,” vol. 17, pp. 1–40, 2015.

[9] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, “Cognitive Mapping and Planning for Visual Navigation,” 2017.

[10] M. Novák, “Mobile robot indoor navigation,” in 2016 17th International Carpathian Control Conference (ICCC), pp. 151–155, IEEE, May 2016.

[11] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, “Learning Modular Neural Network Policies for Multi-Task and Multi-Robot Transfer,” pp. 2169–2176, 2016.

[12] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size,” Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9351, pp. 324–331, Feb 2016.

[13] R. Caruana, “Learning Many Related Tasks at the Same Time With Backpropagation,” Advances in Neural Information Processing Systems, vol. 22, no. 4, pp. 657–664, 1995.

[14] M. D. Zeiler, “ADADELTA: An Adaptive Learning Rate Method,” 2012.