Model car architecture for education in Robotics and Deep Neural Networks

Anonymous Author(s)
Affiliation
Address
`email`

Autonomous driving is a challenging task in which Artificial Intelligence techniques can be successfully applied. This paper describes a robotic platform and its neural network software capable of addressing this problem. AutoMiny is an autonomous model vehicle (scale 1:10) developed at Freie Universität Berlin for educational and research purposes. The vehicle's architecture supports the development of Advanced Driver Assistance Systems (ADAS) based on Deep Neural Networks using a stereo camera and a Virtual Force Field (VFF) path planner during the learning process.

1 Autominy

Our car is an onboard complete system instrumented with an infrared stereo camera, a rotating LIDAR, feedback sensors (servo steering angle and motor odometry), a CPU, an NVIDIA GPU, as well as LEDs for the car lights. A picture of the robotic platform is shown in Fig. 1a.

AutoMiny has been designed as a self-sufficient system that requires no external sensors or cloud computing so that we can simulate our full-size electric car e-Instein [1]. NVIDIA has also developed some projects with car-like robots and GPU [2], nevertheless, these models rely on the CPU power of the NVIDIA GPU board. In the case of a GPU Jetson Nano, the CPU leaves a small room to mix classical algorithms with neural networks to control the car or run parallel processes, on the other hand, Autominy counteract these drawbacks using an Intel NUC i7. The architecture of the car is shown in Fig. 1b.

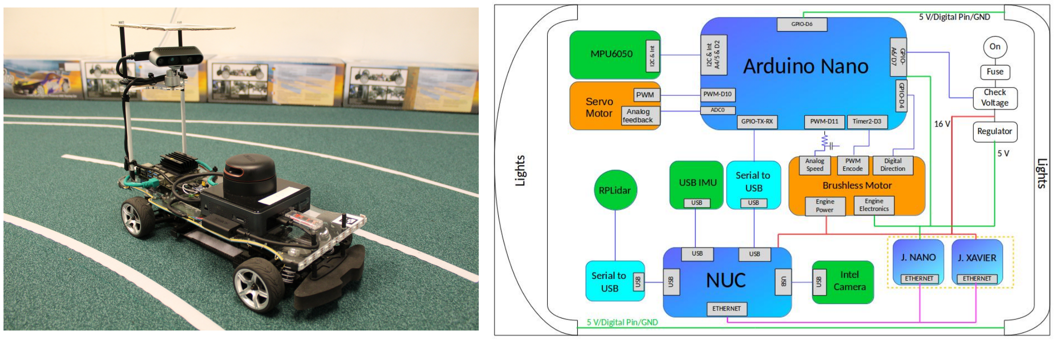
The Operating System of Autominy, as well as the control computer of e-Instein are based on Linux Ubuntu 18.04 and the Robot Operating System (ROS) Melodic. Details about Autominy can be found in `https://autominy.github.io/AutoMiny/`

2 Driving Assistant

Cameras and deep neural networks in an autonomous driving framework are normally applied to perception tasks, such as obstacle [1] and lane detection [2], traffic signal recognition [3], computation of the optical flow [4] or depth estimation [5]. A direct connection between the perception modules and the car hardware using neural networks is possible (it is called sometimes pixel-to-drive)[6]. Development of neural networks can be sped up in the lab using a simulator, in this case, the Autominy platform. The model car and the full-size car share the same robotic framework, so that the experiments in the lab are transferable, through fine-tuning, to the full-size platform.

We have developed a deep network which is able to drive the car on the lab track using sequences of video images. Ground truth is provided by cameras on the ceiling which track the Aruco marker on the car.

[1] `https://autonomos.inf.fu-berlin.de/vehicles/e-instein/`
[2] `https://developer.nvidia.com/embedded/community/jetson-projects`



(a) Autominy with NVIDIA Jetson Nano

(b) Architecture of Autominy

Figure 1: Autominy hardware and architecture

The localization algorithm is used to get the positional ground truth $X = [x, y, \theta]$ and to estimate the desired position $X_d = [x_d, y_d, \theta_d]$ using a VFF path planning algorithm. The algorithm generates a map with all the necessary poses to drive the car within the lanes. The car can, therefore, know the error in position at real time. VFF has some conceptual limitations for the control of the vehicle [7], since it is only used to find the desired position of the car, those limitations are eliminated by the neural network model.

In order to create the labelled data necessary to train the network, we run Autominy manually through the track to saving the video images and the difference between the actual and the desired position given by the VFF. The network learns to steer according to the environment the camera is perceiving.

Since the steering control cannot be unstable or non-smooth, a sequential analysis of pasts decisions should be done by the network, We obtained the best results using two sequential stacked video images as net input. Several convolutions layers are used to find special features that can help the network to decide according to the lab zone. We have found that the use of a LSTM Recurrent neural network in this application helps to connect the history of steering commands and produces smoother decisions. We use two LSTM layers, and finally a couple of dense layers to generate the steering output.

We use mean squared error between X and X_d to calculate the loss function and train the network with the Adam optimizer. The pixel intensities are normalized from zero to one and batch normalization between convolutional layers is used.

We show that the net can find the optimal steering angle regardless of the position of the car on the field to return to the lane and drive on it.

References

[1] Kocić, J., Jovičić, N., and Drndarević, V. (2019). An End-to-End Deep Neural Network for Autonomous Driving Designed for Embedded Automotive Platforms. Sensors (Basel, Switzerland), 19(9), 2064. doi:10.3390/s19092064

[2] Zou, Qin et al. "Robust Lane Detection from Continuous Driving Scenes Using Deep Neural Networks." ArXiv abs/1903.02193 (2019): n. pag.

[3] Tabernik, Domen, and Danijel Skocaj. "Deep Learning for Large-Scale Traffic-Sign Detection and Recognition." IEEE Transactions on Intelligent Transportation Systems (2019): 1–14. Crossref. Web.

[4] Fischer, Philipp and Dosovitskiy, Alexey and Ilg, Eddy and Häusser, Philip and Hazırbaş, Caner and Golkov, Vladimir and van der Smagt, Patrick and Cremers, Daniel and Brox, Thomas. (2015). FlowNet: Learning Optical Flow with Convolutional Networks.

[5] Wofk, Diana et al. "FastDepth: Fast Monocular Depth Estimation on Embedded Systems." 2019 International Conference on Robotics and Automation (ICRA) (2019): n. pag. Crossref. Web.

[6] Bojarski, Mariusz and Testa, Davide and Dworakowski, Daniel and Firner, Bernhard and Flepp, Beat and Goyal, Prasoon and Jackel, Lawrence and Monfort, Mathew and Muller, Urs and Zhang, Jiakai and Zhang, Xin and Zhao, Jake and Zieba, Karol. (2016). End to End Learning for Self-Driving Cars.

[7] Koren, Yoram and Johann Borenstein. “Potential field methods and their inherent limitations for mobile robot navigation.” Proceedings. 1991 IEEE International Conference on Robotics and Automation (1991): 1398-1404 vol.2.