# 5.1

# AI-Based Vehicle Systems for Mobility-as-a-Service Application

**Mikko Tarkiainen[1], Matti Kutila[1], Topi Miekkala[1], Sami Koskinen[1], Jokke Ruokolainen[2], Sami Dahlman[2] and Jani Toiminen[2]**

[1]VTT Technical Research Centre of Finland Ltd., Finland
[2]Vaisto Solutions Ltd, Finland

## Abstract

Achieving sufficient safety measures is among the major challenges in developing automated vehicles that can operate safely in an urban environment. Data fusion between an in-vehicle camera and a LiDAR sensor can be used for detection and tracking of other road users in an automated vehicle. In addition, simulated environments together with high-level deterministic, supervised and reinforcement learning-based autonomous control could provide traffic safety benefits in the future. These AI-based technologies have been studied in the AI4DI project to enable the Mobility as a Service (MaaS) operators fleet management of automated vehicles. The development and testing of these methods are presented in this chapter with the first promising results. The Camera - LiDAR fusion algorithm provided very good results with the accuracy evaluation using the KITTI dataset. The real-time applicability of the fusion algorithm was also successfully verified.

**Keywords:** automated driving, sensor data fusion, 3D object detection and tracking, reinforcement learning, simulation, CNN, training, real-time systems, neural networks.

### 5.1.1 Introduction and Background

This article focuses AI solutions to be used in Mobility as a Service (MaaS) with fleet management of automated driving “last mile” vehicles. Automated vehicles could bring improved efficiency to the MaaS, but road safety is a must. There are several factors affecting the decision making of the automated vehicle. The automated vehicles must be aware of the surrounding road users and obstacles and possess quick reaction times in case unexpected movements or behaviour should occur. This presents the problem of 3D object detection and tracking, which is a major topic of research with automated or autonomous vehicles. Having knowledge of the accurate physical locations of other road users is integral to decision making. In addition, estimation of the speeds and headings of other road users is required to predict possible dangerous situations. Therefore, accurate 3D detection and tracking are needed.

Coming up with the best possible predictions and consequent actions is mission critical requirement for the automated vehicle. A mission critical system is a system that is essential to the survival. The selected action depends on many factors in complex traffic situations. The faster the vehicles are moving, the quicker the cycle of prediction and action selection must be. This creates a critical role for system components of the automated vehicle system including the software components.

This paper presents a novel method for data fusion between an in-vehicle camera and a LiDAR sensor, which enables the vehicle to map 2D image coordinates to a 3D environment and vice versa. This is utilised for detection and tracking of other road users in an automated vehicle. In addition, this paper compares the deterministic, supervised and reinforcement learning-based autonomous control development possibilities on a high level. An autonomous control solution blueprint for a control pipeline that can be trained in a simulation environment is presented. This is done by combining reinforcement learning control planning capability with complementary supervised, learning-based observation metadata detection collection and deterministic safety measures for avoiding collisions and casualties.

### 5.1.2 AI-Based 3D Object Detection and Tracking for Automated Driving

With the increased computing power, there are more possibilities of implementing AI-based solutions for automated driving systems, which require real-time processing rates. A convolutional neural network (CNN) is one popular solution for 2D object detection. While the CNNs for image processing are getting more and more accurate, some additional methods are required to make use of these networks in accurate 3D object tracking.

### 5.1.2.1 Camera and LiDAR Sensor Data Fusion

Data fusion between a camera and a LiDAR sensor enables the vehicle to map 2D image coordinates to a 3D environment and vice versa. With accurate inter-sensor calibrations, the 2D detections provided from image data by a CNN can be transformed into the 3D coordinate system of the LiDAR. This enables 3D location estimation of objects, while taking advantage of the accuracy of a 2D CNN. The 3D points provided by a LiDAR sensor can be projected onto a corresponding 2D image, if the Lidar and the camera providing the images are calibrated to each other.

The method of combining the point cloud and 2D detection boxes to obtain object clusters is described in Figure 5.1.1. The output of a 2D image object detector defines the locations of the objects in the 2D space using rectangular bounding boxes. The point cloud provided by the LiDAR sensor can be projected to the same 2D image, and the pixel coordinates of the projected points can be compared to the detection boxes. The point projections that are inside a detection box boundary can be tagged so that the original 3D point is marked as residing inside a detection box in the 2D perspective. This allows the examination of 3D spatial information of the 2D detection box. A challenge in this method, however, is raised by the fact that the 2D detection boxes are usually drawn so that the entire object is contained inside it, which results in the background area being included especially in the corner areas of the detection box. This means that many of the 3D points that are projected and considered to be inside a detection box, actually originate from the background terrain, and falsify the 3D spatial information related to the actual object. This can be solved by altering the 2D detection box size to make it smaller, and focus it on a certain area of the original detection box. This detection box focusing can eliminate the false point projections originating from the background. This process results in accurate 3D spatial information of the 2D detected objects. With this information, the original 3D point cloud from the LiDAR can be processed to obtain the 3D point cluster representation of the 2D object mapped onto the image by an image object detector.

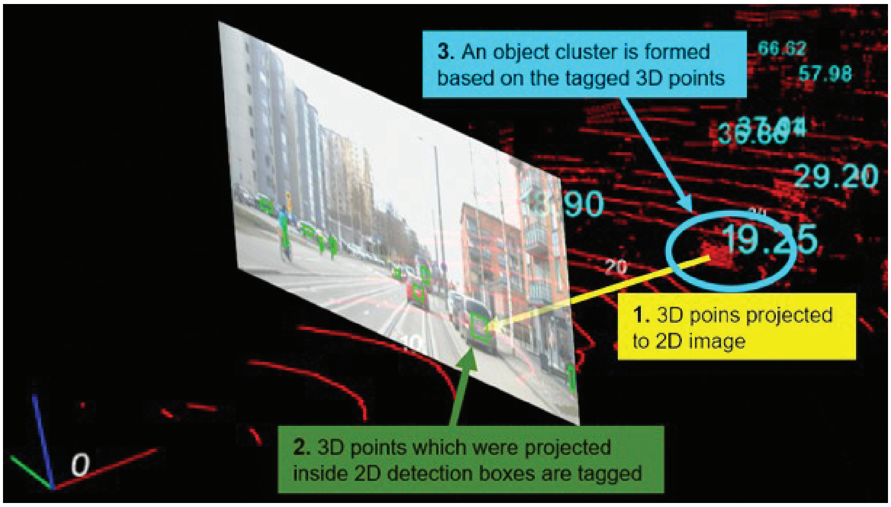
3. An object cluster is formed based on the tagged 3D points
66.62
57.98
29.20
90
19.25
20
1. 3D poins projected to 2D image
0
2. 3D points which were projected inside 2D detection boxes are tagged

**Figure 5.1.1** 3D object clustering using a point cloud and 2D detection boxes.

### 5.1.2.2 Experiments and Results

The presented method of 3D object detection was implemented as a functional real-time system into one of the test vehicles of the VTT automated vehicles research team. The vehicle was equipped with a 32-beam RoboSENSE LiDAR and a 16-beam Velodyne LiDAR for point cloud capturing. The image capturing was performed on a Basler Ace2-series RGB camera, and the images were processed on an Nvidia Jetson Xavier AGX embedded deep learning device. On an actual vehicle integration, the time delays between the data capture events of the LiDAR and the camera must be addressed. To synchronise the point clouds to the captured image more precisely, the odometry data of the vehicle was also captured using a combination of an inertial measurement unit (IMU) and a Global Positioning System (GPS) sensor. The velocity and angular turn rate of the vehicle were used together with the time delays between the camera and the LiDAR to rectify the point cloud to better match the 2D image, and therefore keep the point projections more accurate, even with the vehicle in motion.

This sensor setup was integrated into the vehicle as three separate data capture and distribution modules running on separate computers. The LiDAR-capturing computer collected point clouds, transformed them into the vehicles' common coordinate system, and the modified point cloud was published on an OpenDDS (Open Data Distribution Service) network. The

Jetson device processed the images captured by the Basler camera using the YOLO v4 network [1], and published the images and the neural network 2D detections on the in-vehicle OpenDDS network together with the movement data of the vehicle from the odometry module.

All the processed data was received from the OpenDDS network by the fourth computer, performing the sensor data fusion. The algorithm first received the latest 2D camera-based detections, transformed the LiDAR point cloud and the odometry data. The point cloud was filtered using an Approximate Progressive Morphological Filter (APMF). It is a simplified version of the Progressive Morphological Filter [2], which removes the ground points of the cloud in real-time. The ground points are unneeded, and they are even likely to add error to the later calculations.

The algorithm first applied the detection box focussing, and processed each of the point cloud 3D points. For each 3D point, the delay correction was applied using the odometry and capture time delays, and then the projection was performed onto the 2D image. Then it was checked whether the projected point was placed inside a detection box. This point cloud processing operation was multi-threaded with the sensor fusion computers' processor cores to significantly decrease the computing times.

After matching the 3D points to the 2D detection boxes, the algorithm processed each of the detection boxes to find the objects' 3D location from the LiDAR point cloud. This was done by sorting the points tagged to a detection box based on distance, and choosing the median 3D point as the estimated location of the object. Choosing the median point helps remove any possible noise that might still be caused by some background 3D points, and even occluding obstacles of smaller sizes, which may partially cover the detection box in the image. Another option instead of choosing the median point is to average all of the 3D points which have been tagged to the detection box. This, however, leads to more inaccuracies and makes the estimation much more susceptible to noise from occluding obstacles, for example.

With the estimations of the locations of the objects in 3D, they can be extracted from the ground-filtered point cloud. This was done by cropping the approximate point cloud area containing the object. The 3D crop dimensions depend on the predicted class of the object. For a pedestrian, the cropping is much smaller than for a car, for example. This operation was performed on every object detected by YOLO v4, optimally resulting in the true 3D locations and point representations of the objects, see Figure 5.1.2 and Figure 5.1.3.

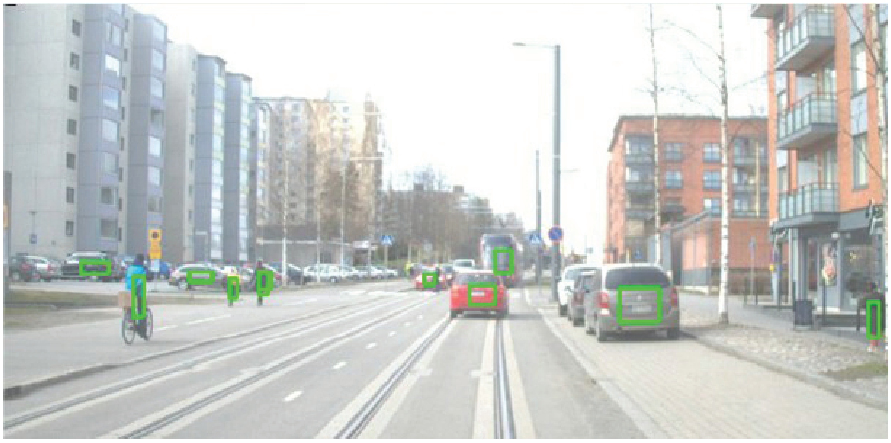

**Figure 5.1.2** 3D detections on camera view.

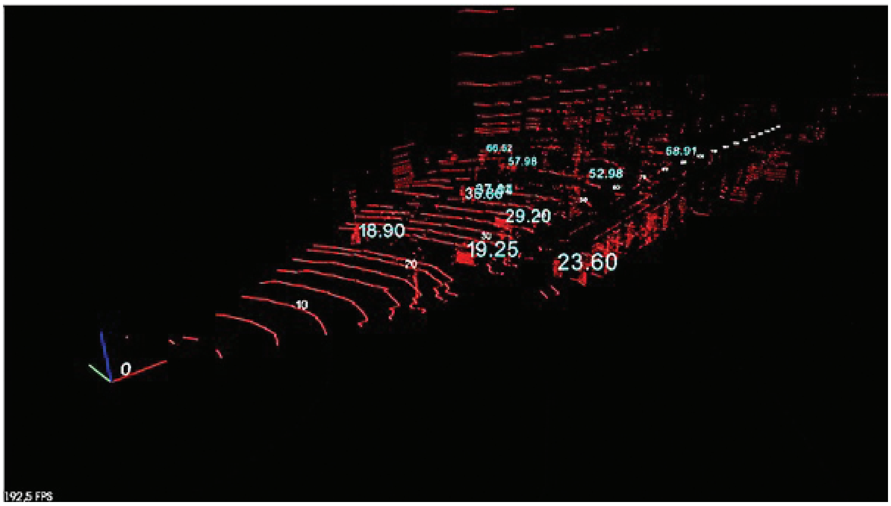

**Figure 5.1.3** 3D detections in LiDAR point cloud.

### 5.1.2.3 Evaluation of the Algorithm and Vehicle Integration

The performance of the algorithm was evaluated on the KITTI 3D object detection dataset [3]. Instead of using the standard KITTI evaluation threshold, a custom method of accuracy evaluation was used to focus more on the accuracy of the projection-based location matching. The evaluation was done by estimating the 3D point clusters representing the objects, and

**Table 5.1.1** Percentage of estimated object cluster means correctly placed inside KITTI ground truth boxes.

| DIFFICULTY | TOTAL | CAR | PEDESTRIAN |
|---|---|---|---|
| Easy | 97.91% | 99.29% | 93.94% |
| Moderate | 92.28% | 92.58% | 90.72% |
| Difficult | 87.64% | 87.50% | 88.50% |

then comparing whether the mean of the points in a single cluster is found inside a KITTI 3D ground truth detection box. If the average point is inside a ground truth box, it is counted as a success, otherwise it is a failure. Only one match per ground truth box is allowed. The accuracy was evaluated with the 'easy', 'moderate' and 'hard' difficulty thresholds KITTI, with the 'car' and 'pedestrian' classes included, see Table 5.1.1.

The vehicle integration was the main goal of the algorithm implementation. For the practicality of the system, real-time operating speed was critical. The YOLO v4 module was able to operate at a rate of 16 Hz with the Jetson utilising the TensorRT library to accelerate deep learning operations. The amounts of the odometry data were comparably very small, and it was streamed in the OpenDDS network at a rate of 20 Hz. The LiDAR point clouds were captured at 10 Hz, which were the largest data stream in the system. Based on the field of view (FoV) of the Basler camera, the points that were clearly out of the camera image frame were ignored to speed up the algorithm. Additionally, the OpenMP multiprocessing library was utilised to parallelise the data fusion operations, increasing the total speed of the integrated system to real-time levels. The inference times were measured in a 728-second-long test in an urban driving environment, resulting in operation rates of 7-10 Hz for the full system.

## 5.1.3 Autonomous Control Prototyping in Simulated Environments

Autonomous control fascinates technology enthusiasts and engineering teams all over the world. Public focus is on autonomous road vehicles for bringing improved efficiencies and safety on the road. In more controlled and restricted operating environments, autonomous work machines and robots have already been able to tirelessly perform cycles of work under human operator surveillance for some time. The ambition and need for research remain clear as more advanced autonomous control seems achievable.

### 5.1.3.1 Reinforcement Learning Control for Mobile Vehicles

The potential to use simulated environments for purposes of training reinforcement learning (RL)-based control agents for mobile machines has been studied in the project by Vaisto. This topic has been studied actively in recent years, see [4] and [5]. Contrary to supervised learning methods, which cover the potential action state space of the targeted operational domain only partially, RL has the theoretical potential to have comparably higher finite action-value state space coverage [6]. Then again, it is a known challenge that applying RL control in the real world is challenging [7].

This higher state coverage brings with it the promise that RL can handle more corner cases, if the RL training process and reward scheme considers the state space coverage as one key performance indicator. There's no certainty that what the action state coverage is and how well an RL agent can adapt to slightly different environments and observations. Research focus remains on measuring the potential actions and state space in each operational domain and then be able to measure that and identify potential pitfalls.

The RL-agent controlled last-mile pod has been trained for project demonstrators in a simulation environment. A short route from the bus stop to the nearby coffee shop has been highlighted in Figure 5.1.4.

The RL agent can drive the pod along any arbitrary and continuous routes in the simulation environment, but in the case wherein the control model was overfitted to be able to collect statistics related to *reward scheme obedience*. The reward scheme monitors the agent's actions, and based on fitting actions, a reward was granted to the agent. If the agent was violating the reward rules, the training episode was ending. The training for this measurement

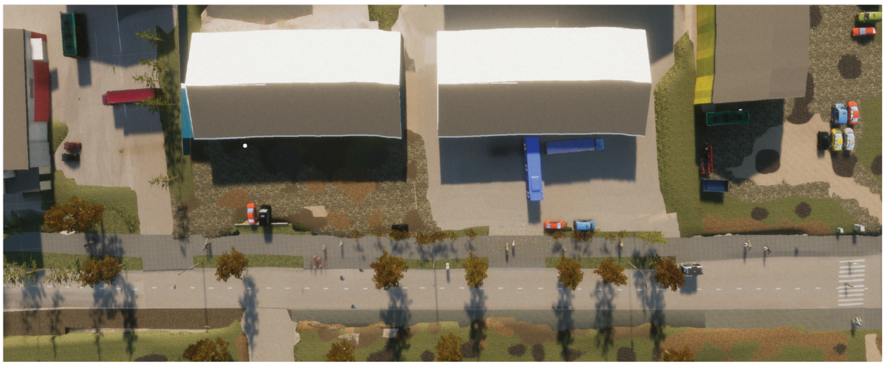

**Figure 5.1.4** Last-mile pod driving scenario.

**Table 5.1.2** The statistics for the agent performance.

| **Episode end reasons:** | | |
|---|---|---|
| Maximum (60m) distance reached | 16601 | 68.68% |
| Episode end checkpoint reached | 4135 | 17.11% |
| Collision with "dummy" car | 2362 | 9.77% |
| Pod was off from the GPS line by 1.5m+ | 617 | 2.55% |
| Collision with pedestrian | 113 | 0.47% |
| The next checkpoint was not reached within 10 sec | 338 | 1.40% |
| The Pod flipped more than 45 degrees | 4 | 0.02% |
| TOTAL | 24,170 | 100.00% |

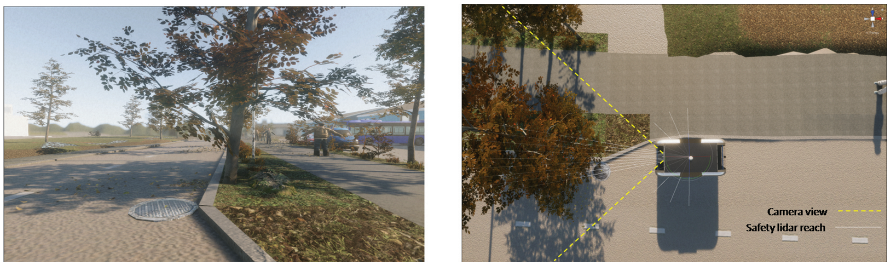

**Figure 5.1.5** Pod sensor view.

was performed on a laptop workstation with Intel i7 CPU. Agent was trained in simulation over 10 nights (~90 hours). The statistics for the agent's performance are shown in Table 5.1.2.

The training for this measurement example is not complete, but the statistics clearly show that "Maximum (60m) distance reached" starts to be in the majority and "Episode end checkpoint reached" has increased to 17.11%, so pod is able to complete the route successfully. Based on our experience the reward obedience continues to improve as training continues. Also, the dummy cars and pedestrian are not naturally behaving at all times and they cause some of the episodes to end. As shown in Figure 5.1.5, the sensors are facing forward and thus they currently leave blind spots around the vehicle.

### 5.1.3.2 The Architecture – Immediate Actions Time-Horizon

Potential solution architecture for autonomous mobile vehicles based on a focus on reinforcement learning was presented above. The main control functions would be handled by a hierarchy of RL agents as shown in Figure 5.1.6. On the top level is a multi-armed bandit agent that would

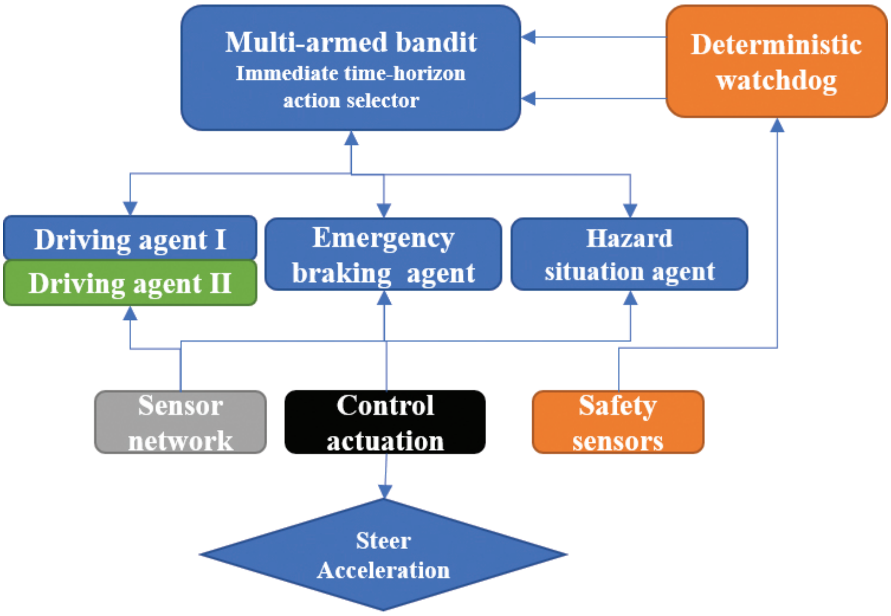

**Figure 5.1.6** Immediate action control research architecture.

have the highest system level control authority. Driving agents would be complementing each other and handling specific parts of driving. Then there would be peer agents performing overlapping primary functions and if the actions proposed by the agents would be equal, then the action is approved for actuation.

The multi-armed bandit would be trained to stochastically select the right agent for proposing action in real-time. The object-detecting environmental sensing would be performed by neural networks trained with finite datasets. The time horizons beyond two seconds can rely on various deterministic and machine learning approaches. A reinforcement learning agent can learn to follow, for example, position breadcrumbs, but the actual planning and optimisation is beyond the scope of this article.

## 5.1.4 Conclusion

This chapter introduced novel methods for data fusion between an in-vehicle camera and a LiDAR sensor for detection and tracking of other road users as well as high-level, deterministic, supervised and reinforcement learning-based autonomous control development possibilities.

The Camera – LiDAR fusion algorithm provided very good results with the accuracy evaluation, being mostly able to locate even the more challenging objects in the KITTI dataset as seen in Table 5.1.1. The real-time applicability of the algorithm was also verified. The developed algorithm makes a valuable contribution to the development of the automated vehicles' environment perception. In addition, the real-time operating speed of the algorithm in the test vehicle was quite fast. However, occasional performance drops also occurred for single frames. In future work, the operating rates could be stabilised by further developing the multiprocessing of the data fusion module.

Reinforcement Learning can be used for developing autonomous driving control in a simulated environment. RL was applied in continuous action space, so the control agents learn to approximate parametric action-value control functions that correspond to real-world needs. A method was presented whereby reinforcement learning is complemented by other machine learning methods or even deterministic safety methods in building a flexible autonomous driving control system. Based on the study, it was concluded that RL potentially plays a role in autonomous control development. Simulation environments are as yet neither visually nor physically on par with the real world, but the gap is getting smaller every year and autonomous control models are already a viable method of product development.

## Acknowledgements

This work is conducted under the framework of the ECSEL AI4DI "Artificial Intelligence for Digitising Industry" project. The project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Czech Republic, Italy, Latvia, Belgium, Lithuania, France, Greece, Finland and Norway.

## References

[1] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, (2020), YOLO v4: Optimal Speed and Accuracy of Object Detection. arXiv 2020, arXiv:2004 10934.

[2] K. Zhang, S.-C. Chen, D. Whitman, M.-L. Shyu, J. Yan, C. Zhang, (2003), A Progressive Morphological Filter for Removing Nonground Measurements From Airborne LiDAR Data. Appl. Opt. 56, 9359-9367 (2017)

[3] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, (2013), Vision meets Robotics: The KITTI Dataset. The International Journal of Robotics Research, 32(11), pp. 1231–1237. doi: 10.1177/0278364913491297.
[4] Praveen Palanisamy (2019): Multi-Agent Connected Autonomous Driving using Deep Reinforcement Learning. arXiv 2019, arXiv: 1911.04175.
[5] Alhawary, Mohammed (2018) Reinforcement-learning-based navigation for autonomous mobile robots in unknown environments. Available from http://essay.utwente.nl/76349/
[6] M. Mitchell Waldrop with Matthew Botvinick: PNAS/What are the limits of deep learning? Proceedings of the National Academy of Sciences Jan 2019, 116 (4) 1074-1077; DOI: 10.1073/pnas.1821594116
[7] Peter Almasi, Robert Moni, Balint Gyires-Toth: Robust Reinforcement Learning-based Autonomous Driving Agent for Simulation and Real World. arXiv 2020, arXiv:2009.11212.