Self-driving cars in Duckietown environment

Önvezető autózás Duckietown környezetben

Endrész Balázs
BME
Budapest, Hungary
endresz.balazs@edu.bme.hu

Monori János Bence
BME
Budapest, Hungary
monoribence@edu.bme.hu

Wenesz Dominik
BME
Budapest, Hungary
dominik.wenesz@edu.bme.hu

Abstract

Our task was to implement a reinforcement learning-based neural network for lane following in the Duckietown simulational environment. *going straight, turning left and turning right*. Thus a *Q-learning algorithm* was implemented. During every step a screenshot is taken which is followed by a segmentation process: the outer white line is filtered to a Blue channel while the yellow line in the middle to the Red channel. After this preprocessing method, a reward function is calculated for the three different actions and the one with the highest possible value is chosen. Our solution was by far not the most efficient approach to the problem, however several promising results are seen.

A feladatunk az volt, hogy egy megerősítéses tanulás (RL) alapú neurális hálózatot hozzunk létre egy útvonalkövetéses problémára a Duckietown környezetben. Ehhez egy Q-hálót építettünk fel. Minden lépésben egy képernyőképet mentünk ki a szimulátorból, a képet szegmentáljuk: a külső folytonos vonalat a kék csatornára, míg a középső szaggatott vonalat a piros csatornára. A feldolgozást követően egy függvénnyel kiértékeljük az aktuális pillanatban legjobb tevékenységet. A mi megoldásunk messzemenőkig nem a legjobb megközelítés volt, ellenben rengeteg ígéretes eredmény született.

1 Introduction

In this semester our task was to create a self-driving AI and test it in Duckietown simulational environment. Reinforcement learning is widely used to problems in robotics, and control engineering, involving self-driving cars, decision processes, and control systems.

The Duckietown Project was initiated in 2016 as an MIT graduate project. It contains vehicles called Duckiebots which circulate around the town. In this project we can participate in a bunch of challenges such as lane following, passing through intersections or controling multiple Duckiebots at once. In this paper we will describe our methods on solving the problems mentioned above and talk about the results.

Submitted to the course Deep Learning a gyakorlatban Python és LUA alapon (BMEVITMAV45). Do not distribute.

2 The solution

2.1 Motivation

Mentioned before: Q-learning was rarely implemented for this specific problem, so we chose to experiment using this method. Our motivation came from an AI beating Google Chrome's built-in dinosaur game. The algorithm converged rapidly with appropriate image pre-processing, hence it seemed good to implement in the Duckietown environment.

2.2 First try - pathfinding

Our first idea was to find the shortest path between two points on a given map and after so a lane following agent could go through that. Hence an A* algorithm was implemented to obtain a valid route, but this solution basically does not involve any neural networks, thus it is scratched.

2.3 Image preprocessing

In the simulation in every step a screenshot is taken, preprocessed and after that it goes into the convolutional neural network. First of all, we cut and resized the pictures to 80x60 pixels, so the unnecessary data won't affect the agent's behaviour.

Figure 1: The raw resized image.

Our first approach to process these images was to filter out the lines on the road, giving us a black and white picture as seen below. To make the learning more efficient we have implemented a better segmentation algorithm. In this case the outer white lines are filtered to the blue channel while the inner dashed line is filtered to the red channel. This is managed in a HSV color model since OpenCV supported this way better: lower and upper thresholds are given for the white and yellow colors. If a pixel is within the pre-specified range then it is filtered to either the blue or red channel. A mask is created such way which is merged with the raw picture, and after that is normalized. The results can be seen below:

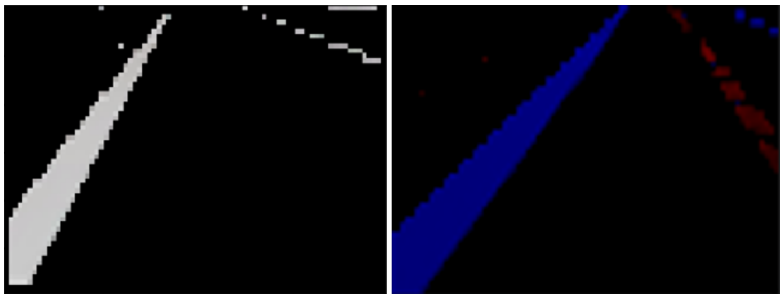

Figure 2: The results from segmentation.

2.4 The Convolutional Neural Network

Since our task is based on visual elements, we have created a 2D Convolutional Network:

Layer		
Name	Shape	Param
Conv2D	(58,78,32)	896
MaxPooling2D	(29,39,32)	0
Conv2D	(27,37,32)	9248
MaxPooling2D	(13,18,32)	0
Conv2D	(11,16,64)	18496
MaxPooling2D	(5,8,64)	0
Flatten	(2560)	0
Dense	(128)	327808
Dense	(3)	387

Table 1: The CNN model

2.5 Q-learning

In every step a screenshot is taken and based on the (1) equation, a value is calculated for every possible action - *going straight, turning left, turning right*:

$$Q' = Q + \alpha(r + \gamma \cdot \max Q_{est} - Q) \tag{1}$$

where Q' is the new value, Q is the old value, α is the learning rate (which was chosen as 0.05) r is the reward, γ is the discount factor (which is 0.95 in our case), Q_{est} is the estimated value to a given action. Using this function the best action is chosen based on the reward function.

3 Results

Our results didn't happen to be the best solution, however it seems like it is mostly just a matter of time to achieve better results. The following graph shows the loss function:

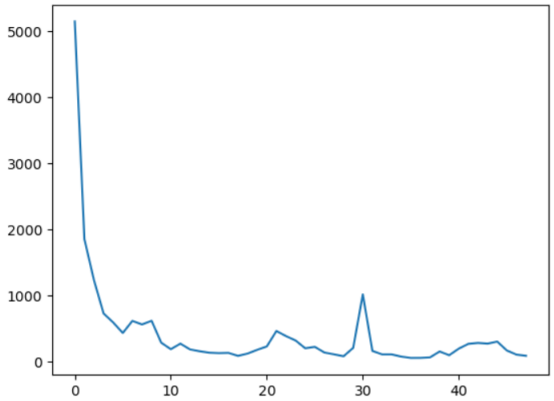

Figure 3: The loss function.

We can clearly see that the method did converge, however it is by far not enough for a stable solution. Several things can be done to get a better agent, such as using checkpointer, using a scheduler which reduces the learning rate α over time. Also it should be emphasized that our method was experimentary, and other reinforcement learning algorithms might do better on this task.

4 Conclusion

In this paper we describer our methods to solve an interesting lane following problem in the Duckietown simulational environment. We found that Q learning might not be the best solution to this problem nonetheless it seems promising. We showed that this can be a viable approach even if a bunch of things could be implemented to achieve better results.

Acknowledgement

We would like to thank all the professors and PhD students for all the assistance and guide given to us. Altough our solution is not perfect, it was a very interesting task to work on, and we feel grateful for the learning experience.

References

[1] Péter Almási, Róbert Moni, and Bálint Gyires-Tóth. Robust reinforcement learning-based autonomous driving agent for simulation and real world. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.

[2] C.K. Chow and T. Kaneko. Boundary detection of radiographic images by a threshold method. In Satosi Watanabe, editor, *Frontiers of Pattern Recognition*, pages 61–82. Academic Press, 1972.

[3] Eyal Even-Dar, Yishay Mansour, and Peter Bartlett. Learning rates for q-learning. *Journal of machine learning Research*, 5(1), 2003.

[4] Robert M. Haralick and Linda G. Shapiro. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29(1):100–132, 1985.

[5] András Kalapos, Csaba Gór, Róbert Moni, and István Harmati. Vision-based reinforcement learning for lane-tracking control. *ACTA IMEKO*, 10(3):7–14, 2021.

[6] András Kalapos, Csaba Gór, Róbert Moni, and István Harmati. Sim-to-real reinforcement learning applied to end-to-end vehicle control. In *2020 23rd International Symposium on Measurement and Control in Robotics (ISMCR)*, pages 1–6, 2020.

[7] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.