07.06.2022

Deep Learning for Autonomous Vehicles

Final Project Report

Bastien Darbellay, Cengizhan Bektas, Zakarya Souid

This project has been made as part of the lecture “Deep Learning for Autonomous Vehicles”. The goal being to win a race in a human-robot tandem, we had to develop an algorithm which allows the robot to follow a person. To do so, we divided the problem into two different parts. First, the robot must be able to detect humans and differentiate them one from another. Second, the robot must be able to track a person over different frames. Those two parts are further detailed below.

Milestone 1

The goal of this milestone is to detect the different persons that are in the field of view of the camera. In a second step we then also want to identify a person of interest based on a hand sign.
To detect the humans we decided to use YOLOv5. YOLO belongs to the One-Stage Detectors family and is the state of the art algorithm for object detection due to its speed and accuracy. It divides images into a grid system where each cell in the grid is responsible for detecting objects within itself. To detect the person of interest we decided to use a hand sign. More precisely, showing the hand palm with spread out fingers. To detect the sign, we are using our own trained model. Once the sign has been detected, we are searching for the person in the list of detected humans with the biggest intersection of the rectangles between each person and the rectangle of the hand sign. Then we are selecting the person with the biggest overlap. To avoid unneeded processing, we are removing all persons in the list which has a smaller area of a specific threshold we set, to guarantee that we are only interested in the persons which are in the foreground. Finally we are calculating the coordinates of the bounding box to return the position of the rectangle of the person of interest.
One issue we had with this method was, when two persons were one behind the other, the overlapping area of the hand and the persons were equal and could lead to a wrong initialization of the person of interest. To solve this problem, we decided to identify the person of interest not only based on the overlapping area with the hand, but also based on the size of the bounding box of the person. We assumed that the person of interest would be close to the robot and therefore have a bigger bounding box than the persons in the back. With this modification, we were able to accurately detect the person of interest, even in ambiguous conditions.
Another issue we had in this milestone was the training of our custom model. Since we worked on Google Colab, we were quite limited by the hardware and the time this hardware was at our disposal. To solve this issue we had to move the training process to SCITAS where those limitations were no longer an issue.

Milestone 2

For milestone 2, we aim to address the problem of tracking objects. During the initial research we looked at various state of the art algorithms and simple methods to do tracking. Some of the methods we looked at includes deep sort tracking, which looks at not only the distance and velocity of the object detected but also the similarity of the images. Another method we looked into was SiamMask, which is another modern implementation of the Siam network. Ultimately, we decided to follow through with siamFC, which applied a simple tracking algorithm on the fully-convolutional Siamese network. SiamFC suits our needs as it is a single object tracker, SOT, and matches what we needed for this project. The Siam network is essentially a trained network to locate an exemplar image within a larger search image, where the similarity learning is done within the bilinear layers.

Milestone 3

For the last milestone we just combined the first milestones and insert this into the ***detector.py*** file we were provided for this project. We established a connection to our home directory on the V100 and copied all needed files in order to run our implementations. For this we just updated the ***class Detector()*** which was already implemented in the detector.py file. We inserted all the functions we implemented above and called the function for detection and tracking in the ***forward()*** method, which is also implemented in the class ***Detector()*** inside the detector.py notebook. The problem we had during milestone 3 was to test our implementations on the robot. Due to the limited number of robots and a huge amount of groups within this class, we were not able to test our system in a performant way. Even we appeared to each of the TA sessions, we just had the opportunity to test our implementations in a really short time interval. After performing very well on milestone 1 and milestone 2, we really wanted to win the race.