# CIVIL-459 Deep Learning for Autonomous Vehicles – Project Report

Tanguy Lewko, Chengkun Li, Yifeng Chen, Aoyu Gong
*École Polytechnique Fédérale de Lausanne, Switzerland*

## I. Introduction

During this project, we developed a detector and a tracker that we later used to implement a person-following algorithm on a Loomo robot. The algorithm was then tested during a race. The main challenge is this project is that our algorithm should be effective even in hard conditions, for example when the target person is not in the frame, or obstructed by other objects. The project is divided into three milestones: detection, tracking, and implementation on the Loomo robot.

## II. Milestone 1 - Detection

The goal of the first milestone is to be able to detect, in real-time, a specific person, by using our own approach. We chose to detect a particular clothe: the hat of one group member.

### A. The YOLOv5 Model

Aiming to detect the hat, we chose YOLOv5 as the object detection algorithms due to its speed and accuracy. As shown in Figure 1, the architecture of YOLOv5 consists of three main parts: backbone, neck, and head. The backbone is mainly based on *Cross Stage Partial Network* (CSPNet) and *Spatial Pyramid Pooling* (SPP) which extracts feature maps of different sizes by multiple convolutional and pooling layers. The neck includes two parts: *Feature Pyramid Network* (FPN) and *Pixel Aggregation Network* (PAN). The FPN conveys semantic features from higher feature maps to lower feature maps. The PAN conveys localization features from lower feature maps to higher feature maps. The head is used to predict targets with different sizes on feature maps. The model consists of four architectures: YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. In this milestone, YOLOv5s was trained and tested due to its smallest and fastest model.

### B. Dataset

The dataset for the object detection contains 1166 images of one group member wearing the hat in many different places, such as campus, gardens, museums, subways, streets, shops, and so on. When taking them, we changed the distance, light, directions, and obstructions as much as possible. We also considered different angles and focal lengths of the camera. Then, we labeled the hat on all these images using Roboflow. To adapt to the architecture of YOLOv5, we resized all images into the size of $640 \times 640$ pixels. Then, they were divided randomly into the training, validation, and testing dataset, which consisted of 816, 225, and 125 images respectively. For data augmentation, we implemented the following steps to generate three new images from one training image, including randomly cropping, rotating it between $-15°$ and $+15°$, changing its saturation and brightness between $-25\%$ and $+25\%$, and applying mosaic. After that, the training dataset consisted of 2448 images.

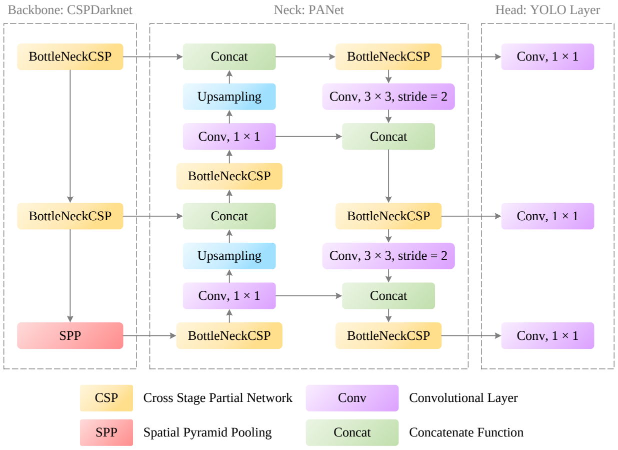


Figure 1: The architecture of YOLOv5.

### C. Result Analysis

Then, we trained YOLOv5s on our dataset. We used this architecture because its inference speed is really fast and its results are great. The training part was really easy to do, so what we really learned in this milestone is how to create a good dataset, and use it with already existing algorithms. Our results using this algorithm are good and robust. We often have a very good confidence in the detection: more than 0.9. We have extremely few false detections when using a confidence threshold of 0.5 (In the final race we set it to 0.6). Since we know there will be only one person of interest wearing the hat on a frame, we also decided to only keep the bounding box with the highest confidence. We added pictures with no hat labeled to avoid false positives when detecting. We show in Figure 2 the results on some of our images from the test set. We can see that even on the fourth image, where the hat is far away, obstructed behind an object, and with bad light conditions, the hat is detected with a high confidence.

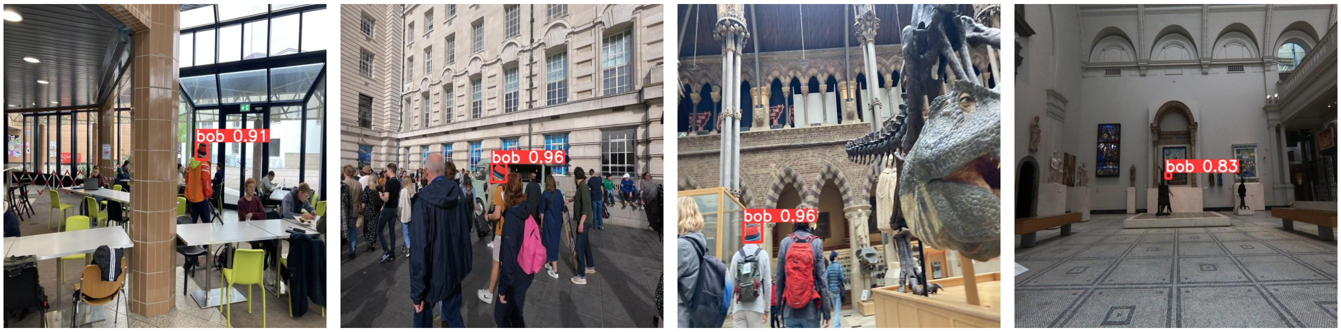


Figure 2: Detection with testing images.

## III. Milestone 2 - Detection + tracking

The second milestone was about developing a tracker to track the person of interest. In order to do so, we used the Deep SORT algorithm combined with YOLOv5s.

### A. *The Deep SORT Algorithm*

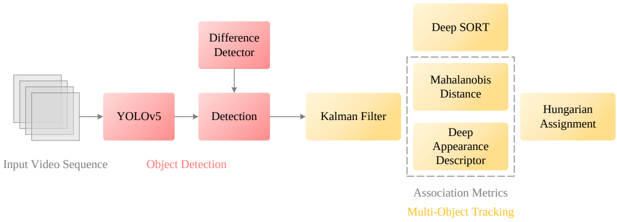


Figure 3: The architecture of deep SORT.

### B. *Initialization and Tracking*

The tracker is initialized when we detect the hat. Then, YOLOv5 is used to detect the bounding box of the person which has the biggest intersection area with the bounding box of the hat. The bounding box of this person is then fed to the tracker, and the person of interest can remove the hat and be tracked.

### C. *Result Analysis*

Our tracker handles well with scenarios where the person of interest leaves the frame, and when that person comes back the tracker could immediately rediscover and track the location of the person. The main weak point of our tracker is that if a person is partially obstructed, by another object for a long time, the tracker might identify another person as the person of interest (this happened a few times during tests with many people in front of the camera). To solve that, we thought about using the first bounding box to track instead of the last one. It solves the problem but we ended up with less good performance in general when the person is moving fast. So we decided to stick to the first method. Maybe using a combination of the first and and the last bounding boxes could give us better results but we did not do it before milestone 2 deadline.

## IV. Milestone 3 - Tandem race

### A. *The Choice of Algorithm*

The final milestone is deploying our algorithm on the Loomo robot. To do so, since our detection part was working perfectly well and was robust thanks to the fact that we spent some time creating good dataset, we decided to only use detection and not tracking. Indeed, even if the tracker was working well, we still take the risk that it might change the person of interest in some rare cases, which we hope to avoid for the race. Also, it saves up the time for re-initialization of the tracking algorithm if the tracker loses the person of interest.

### B. *The Novelty of the Final Detection Algorithm*

To add smoothness to our person-following algorithm, we decided that if the person of interest was not detected for an intermediate frame (usually happens when the light condition changes drastically), keep the last bounding box for a five frames before making the robot stop. This allows us to simulate a zero linear constant Kalman Filter for a short time in case there is no detection. And it worked pretty well in practice given the fact that manipulator of the Loomo rarely gets out of camera's FOV.

### C. *The Result of Final Race*

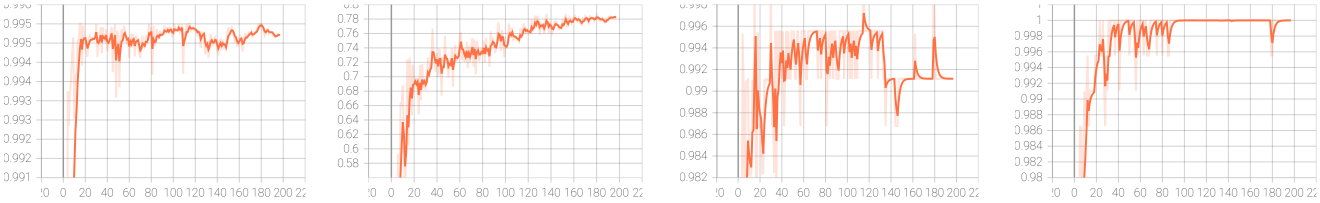
Since we chose the appropriate algorithm and guided the robot well during the race, we achieved a great result for the final race. For the race against the clock, we took 49.58 seconds to finish it, ranking second among all teams. For the final competition, we achieved the first place!

## Acknowledgements

The authors thank Prof. Alexandre Alahi, Parth Kothari, George Adaimi, Mohammadhossein Bahari, Saeed Saadatnejad and other TAs for providing the robot, computing resources and careful guidance, as well as for the well-organized competition.

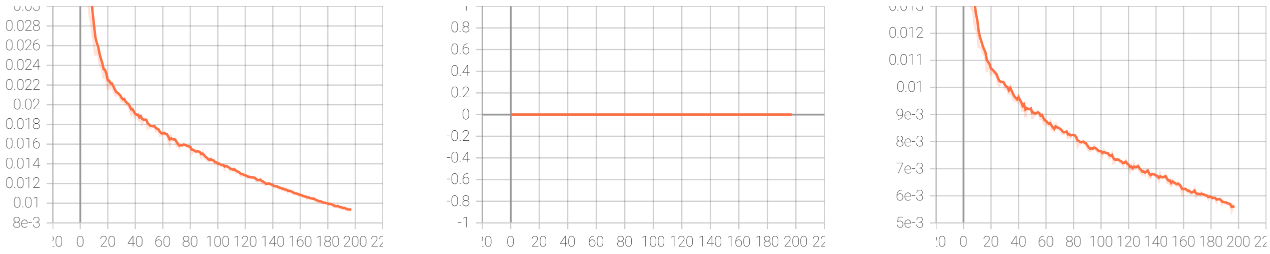
## Appendix A.
## Training Results

The training results are provided as follows.



(a) mAP@.5 (b) mAP@[.5:.95] (c) Precision (d) Recall

Figure 4: The four metrics as a function of the number of iterations.



(a) The bounding coordinates (b) The object confidence (c) The classification loss

Figure 5: The three losses as a function of the number of iterations.

## Appendix B.
## Photos in Tandem Race

The photos in Tandem Race are provided as follows.

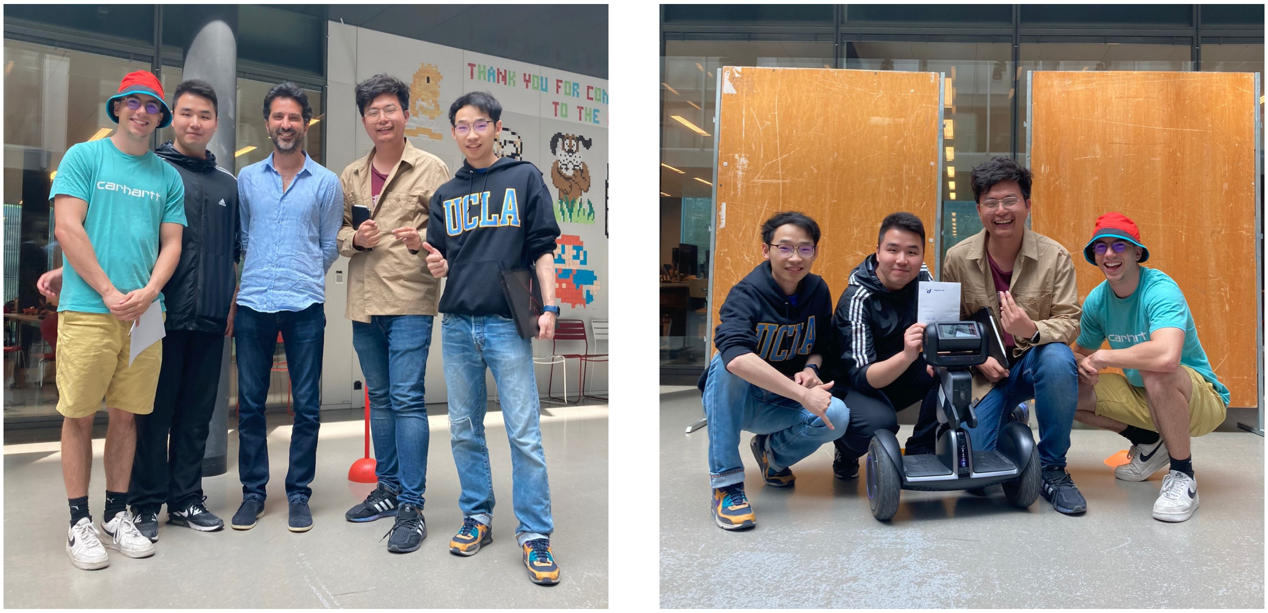


Figure 6: The photos in Tandem Race.