Real-time activity detection and social distancing estimation with a webcam

I. Intro

Following the Covid pandemic the Monoloco library was extended with a new social distancing feature. Utilizing the absolute distance to estimate the relative distance between individuals, it also estimates the orientation of people to determine if a risk of contamination is present (based on a threshold). To fully take advantage of this feature in a real world scenario it was essential to be able to take video as input. It also allows users to test the algorithm quickly using their webcam and facilitates the debugging of certain new features.

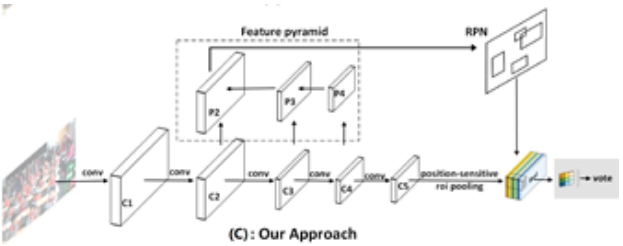
Creating the pull request for the webcam was also an opportunity to simplify the creation of different activity detections. This allowed us to create an extension for raising arm detection and also for the recognition of cyclist intention. Detecting a raised hand could be very useful in the context of autonomous driving by allowing pedestrians to signal an autonomous taxi for instance. Or also in a bus, allowing passengers to signal the driver just by raising their hand. Cyclists intention recognition is essential for an autonomous car allowing the prediction of movement of vulnerable road users. Monoloco allows us to take OpenPifPaf key points as inputs for these problems.

II. Related work

A. Raising hand detection

Previous work on this subject focuses on the classroom scenario, i.e. detecting which students in a classroom are raising their hand. The state of the art for counting the number of raised hands is this paper : Jiaxin Si, Jiaojiao Lin, Fei Jiang, Ruimin Shen, *Hand-raising gesture detection in real classrooms using improved R-FCN* [1], which achieves 90% in mean Average Precision. It uses direct video output as input and a dataset that is not publicly available for training. The architecture of the neural network is based on region-based, fully convolutional networks (R-FCN). Another paper uses a previous iteration of this architecture (Lin et al. 2018) but in addition to direct video output they use pose estimation to match each raised hand to the correct student : Huayi Zhou, Fei Jiang, Ruimin Shen, *Who Are Raising Their Hands? Hand-Raiser Seeking Based on Object Detection and Pose Estimation* [2]. They achieve 83% accuracy (compared to 85% with direct video output as input from Lin et al. 2018). They also use their own dataset that is not publicly available, created with videos of classrooms of children. They first run the improved R-FCN on the input image to create boxes around raised hands and then

use Pytorch Openpose to match each raised hand with the correct student using heuristics.



The improved R-FCN architecture

B. Cyclists intention recognition

The state of the art for this problem is : Zhijie Fang, & Antonio M. López. (2019). *Intention Recognition of Pedestrians and Cyclists by 2D Pose Estimation* [3]. They created their own dataset consisting of 40,218 frames from videos they took and 1,626 additional frames from YouTube that they annotated. They perform arm signal classification using a skeleton of 13 key points. They detect 4 different gestures : standard left (extending the left arm to the left), standard right (extending the right arm to the right), alternative right (raising the left hand to go right) and stop (pointing down with the elbow making a 90 degree angle). The annotations are vehicle centric, examples can be seen below:

Examples of standard gestures, these pictures are respectively annotated as right, right and left.

Examples of non-standard gestures, these pictures are respectively annotated as right and stop.

They achieve very good results on their own dataset with 93% accuracy and a F1 score of 92% and on the YouTube videos they achieve 82% accuracy and a F1 score of 76%. The skeleton key points are predicted using an R-CNN.

III. Method

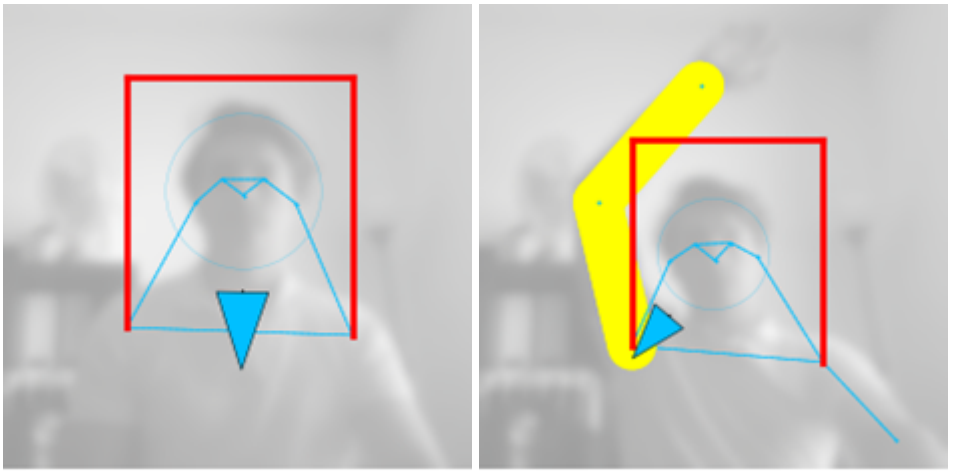
A. Simple heuristics

1. Raising hand detection

Having no dataset available to train a model for this task, I first decided to try with simple heuristics to create a baseline and see if the results were good. Therefore, using the OpenPifPaf key points I wrote an algorithm that would define a bounding box around the head of the person to exclude any head scratches or phone calls. Then I just had to check if the hand was above the shoulders with a sufficient angle.

$$[y(hand) < y(shoulder)] \wedge [\theta(forearm, arm) \geq 30] \wedge (\neg bounding\_box)$$

where $bounding\_box = [x(hand) \leq x(shoulder)] \wedge [y(hand) \geq [y(nose) - [x(left\_ear) - x(right\_ear)]]]$



Visualisation of the bounding box

2. Cyclists intention recognition

For standard gesture detection, this problem is quite simple, we just need to check if the angle between the arm and the forearm is bigger than 90 degrees and that the hand is past the axis of the shoulder. For more complex gestures we tried an approach similar to raising hand detection, by trying to detect a raised hand for the alternative right and the inverse for the stop signal (same angle but with the hand below the elbow).

B. Trained model

As we will show later, the results for raising hand detection with simple heuristics ended up being very good and the lack of a dataset led us to not train a model for this task.

For cyclists intention recognition however the results with simple heuristics were not convincing enough for the non-standard gestures. We therefore decided to train a model using the CASR dataset from [3]. For this task we adapted the existing

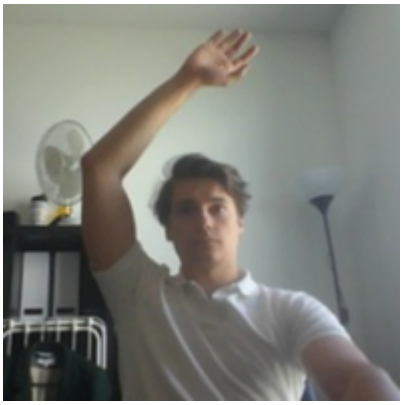
architecture of Monoloco to be able to train for classification tasks. This was done at the end of the project and thus not much time was left for training, testing and tuning the model. It is important to understand that this was done to test the performance of the current Monoloco architecture on these tasks.

IV. Experiments

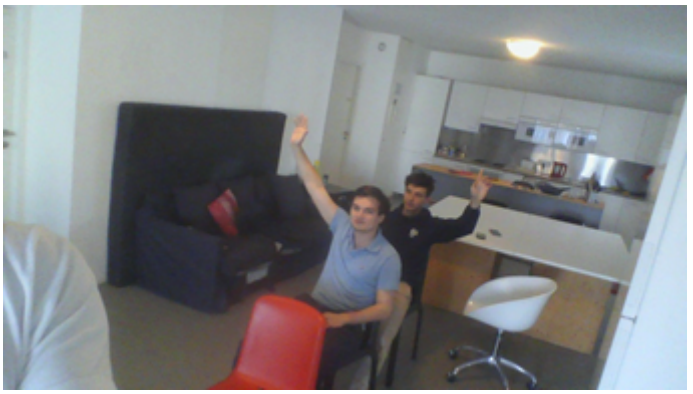
A. Raising hand detection

1. Dataset

No dataset was publicly available for this task I therefore chose to create my own for testing purposes. I created a first dataset composed of 495 frames using my webcam and the TeachableMachines tool from Google. This dataset was very basic and represented ideal conditions for OpenPifPaf detection, only one person, not too far away from the camera and with no obstructions, it was mainly to verify the behavior of my heuristics. I then created another dataset with the help of some friends, trying to replicate a bus environment. This dataset was composed of 459 frames, with 2 to 3 people sitting on chairs, one behind the other.



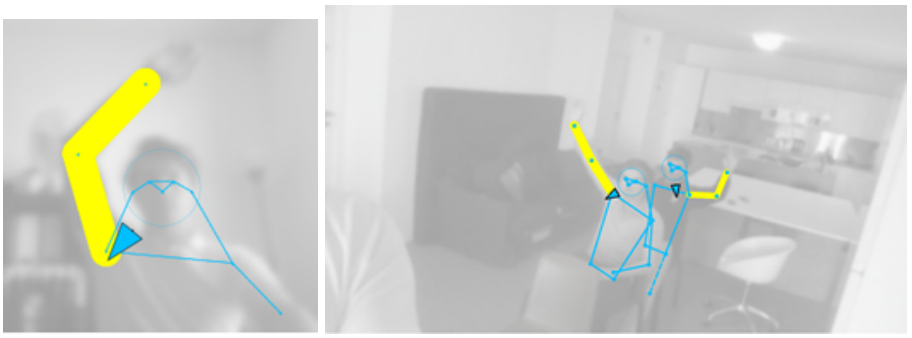
Example frame from the “ideal” dataset



Example frame from the “bus-like” dataset

2. Evaluation protocol

For each frame in the different datasets we checked if each person that is raising one or two arms is detected. This is done by using the activity extension of Monoloco and running the `predict` command with the flag `--activities raising_hand`. The output of this command is a json file that contains an entry named “raising_hand” consisting of a flag that is either ‘left’, ‘right’, ‘both’ or None for each person detected by OpenPifPaf. We then compiled the results, measured the accuracy, the proportion of false positives and false negatives.



Examples of true positives

Example of false negative (the person behind is raising his hand but is not detected)

3. Results

Quantitative results are shown in the table below. The results are excellent with the ideal dataset, as expected, and with the bus-like dataset we still get an accuracy of above 85%, with mostly false negatives. Analysing the different frames where the algorithm fails, we notice that it is always due to OpenPifPaf not detecting the person or the joints of the arm. This is mostly caused by obstructions. When testing the algorithm on some classroom pictures (example pictures taken from [1] and [2]), we notice that the same problem arises, the pupils that are far away from the camera and obstructed by all those that are in front are often not detected by OpenPifPaf. There was also a case (pictured below) where the angle of the arm and the forearm was below the chosen limit of 30 degrees.

Dataset	**Accuracy**	**False positives**	**False negatives**
Ideal	99.59%	0.41%	0.0%
Bus-like	85.56%	3.01%	11.42%

No arms detected by OpenPifPaf

To far away to be detected by OpenPifPaf

Angle is too small for detection

B. Cyclists intention recognition

1. Dataset

For this task we used the CASR dataset from [3], composed of 40,218 frames from videos they took and an additional 1,626 annotated frames from YouTube. From this dataset we also selected frames containing only standard gestures (left arm extended to the left to go left and right arm extended to the right to go right) and created a smaller dataset with those containing 31 413 from CASR and all the frames from YouTube, or a total of 31,413 frames.

2. Evaluation protocol

a) Heuristics

For evaluating our simple heuristics algorithm we ran it on the full CASR dataset and on the standard gesture dataset (with the youtube videos each time) and compared our results to the ground truth given in the annotations.

b) Trained model

For the trained model, we used the same split as [3], by taking two of the cyclists for training and the other two for validation. The same was done with the smaller dataset containing only standard gestures.

3. Results

a) Standard gestures

(1) Heuristics

When considering only standard gestures the results with simple heuristics are very high on both the CASR dataset and the YouTube videos (see table below). The cases where it fails are usually when the cyclist is too far from the camera and is not detected by OpenPifPaf (like the example below).

The cyclist is too far to be detected by OpenPifPaf

(2) Trained model

With only the standard gestures, the model performs really well across the board, achieving more than 90% accuracy and F1 score on the YouTube videos, and near perfect scores on the CASR dataset.

Method	**Accuracy**	**F1**	**Acc-YT**	**F1-YT**
Heuristics	0.96	0.92	0.93	0.91
Model	0.99	0.98	0.92	0.90

Results for standard gestures

b) All gestures

(1) Heuristics

With all the gestures however, the heuristics algorithm's performance is quite poor. It is still high on the YouTube videos because those only contain standard gestures. Compared to the method used in [3], the accuracy is 16% less and the F1 score is 25% less on the CASR videos. On the YouTube videos this method is performing better than the original paper. This is where the limitations of simple heuristics really show, fine tuning the algorithm to be able to detect more complex gestures reliably would have been very inefficient and it would make more sense to use a machine learning approach.

(2) Trained model

With all the gestures, the model performs really poorly, it is very good on the dataset it was trained on, but on the YouTube videos, the F1 score is below 40%. This can be explained by the fact that the model confuses left signaling with the stop signal when the arm of the person is low, as shown in the example pictures below. The precision on the YouTube is quite high at 92% but the recall is very low at 22%, which explains the scores in the table.

The picture on the left is misidentified as stop, while the one on the right is correctly identified as left.

Method	**Accuracy**	**F1**	**Acc-YT**	**F1-YT**
Fang ([3])	0.93	0.92	0.82	0.76
Heuristics	0.77	0.67	0.89	0.88
Model	0.96	0.95	0.61	0.36

Results for all gestures

Bibliography

[1] Si, Jiaxin, Jiaojiao Lin, Fei Jiang, et Ruimin Shen. « Hand-Raising Gesture Detection in Real Classrooms Using Improved R-FCN ». *Neurocomputing* 359 (24 septembre 2019): 69-76. https://doi.org/10.1016/j.neucom.2019.05.031.

[2] Zhou, Huayi, Fei Jiang, et Ruimin Shen. « Who Are Raising Their Hands? Hand-Raiser Seeking Based on Object Detection and Pose Estimation ». In *Asian Conference on Machine Learning*, 470-85. PMLR, 2018. http://proceedings.mlr.press/v95/zhou18a.html.

[3] Fang, Zhijie, et Antonio M. López. « Intention Recognition of Pedestrians and Cyclists by 2D Pose Estimation ». *ArXiv:1910.03858 [Cs]*, 9 octobre 2019. http://arxiv.org/abs/1910.03858.