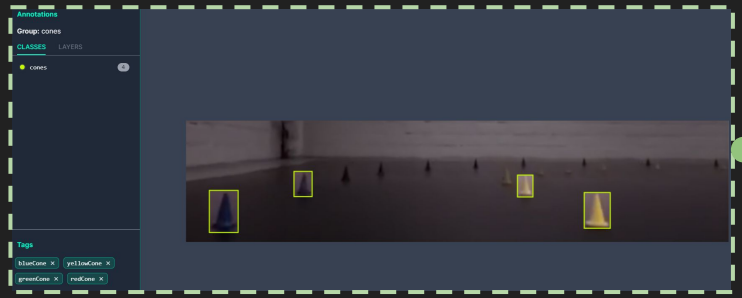
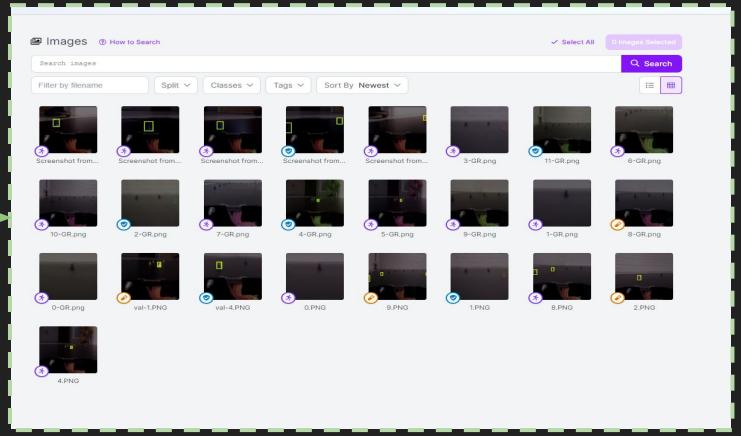
Group 14

Yolov8: Train Dataset with Roboflow

Manually annotate cones:



Dataset to train custom model:

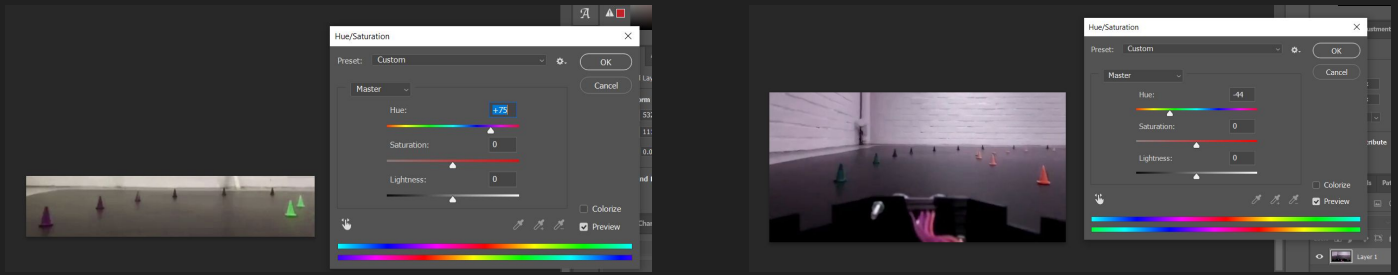


Yolov8: First video generated:

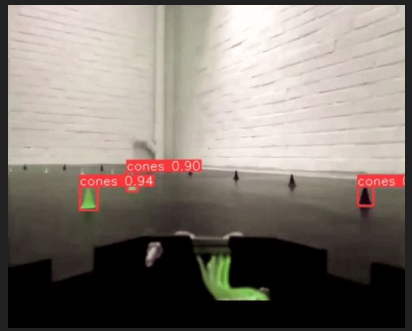
The video detected random stuff on the roof and car, therefore we concluded that the next step was to crop the image

Train yolov8 to detect cones of different colors:

Step 1: Change Hue of images to change cone-colors



Step 2: Run the prediction of the trained dataset



Yolov8: Final version

- Cropped
- Accurate detection of the closest cones

HSV - Cone detection

The trade-offs:

Advantages:

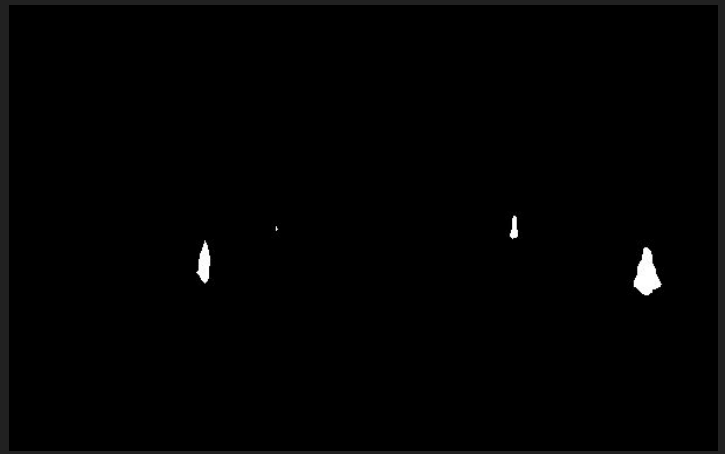
- Highly flexible with regards to supported hardware and environments
- A more performant approach, operations run quickly

Problems:

- Filtering and detection is far from perfect
- Noise:
  - Objects having same color as cones (.e.g ground, wires)

Efforts to improve accuracy:

- Mask unwanted regions of the frame



Efforts to improve accuracy:

Filter detections by shape:

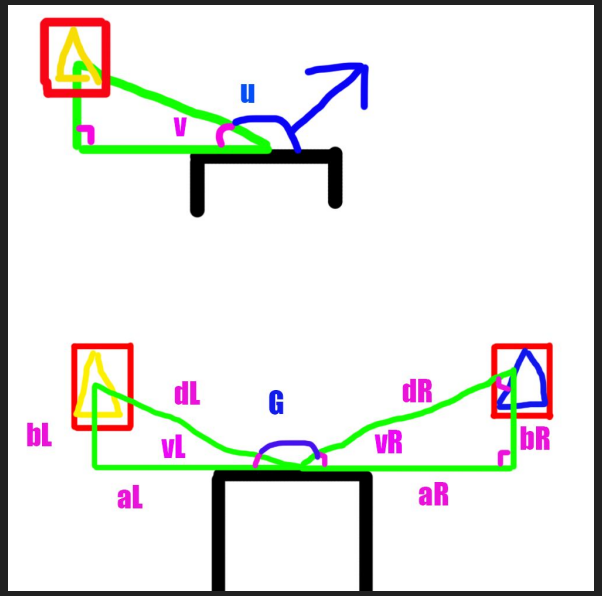
Cones are generally in the shape of a vertical rectangle, this allows us to get rid of more unwanted noise.

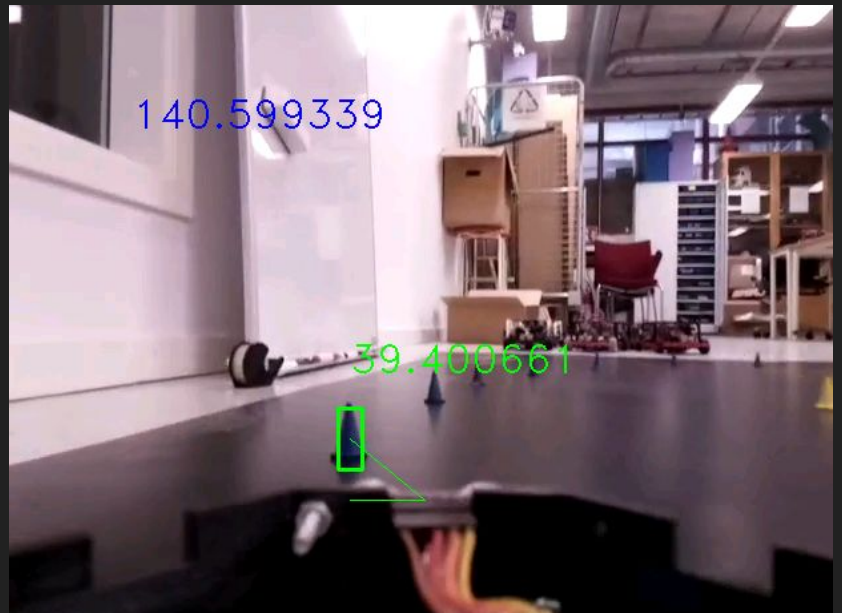
```
bool isCone(cv::Rect rect)
{
    return rect.height > (rect.width * 1.35) && rect.area() >= 700 && rect.area() <= 1600;
}
```

Algorithm - Trigonometry

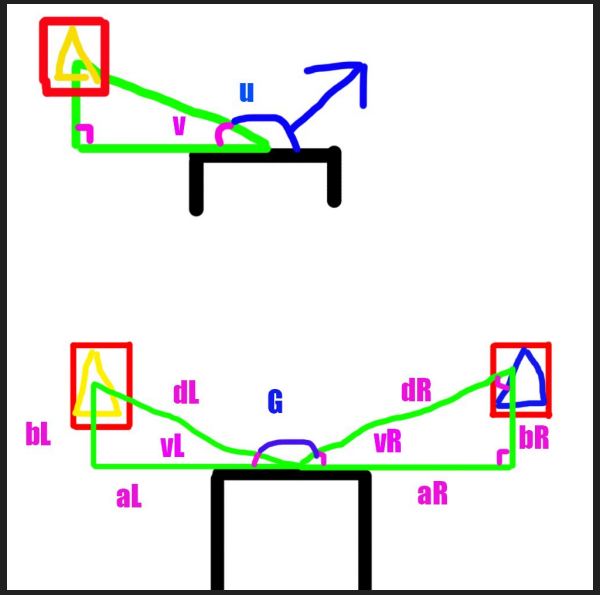
G = 180 - (vL + vR)

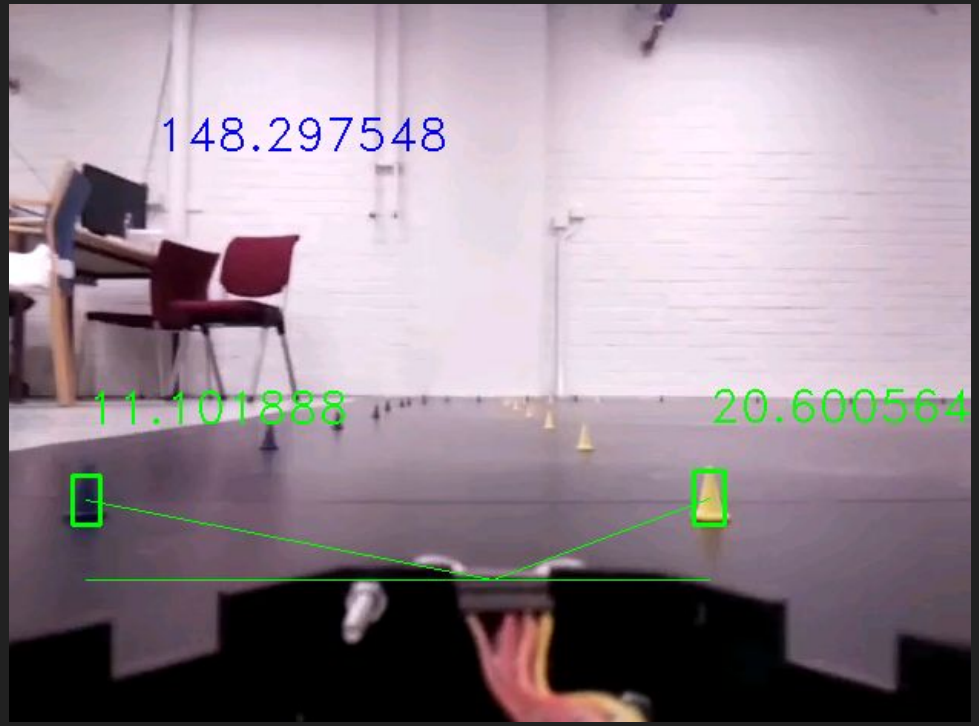
G = 140
vL = 39.4
vR = 0



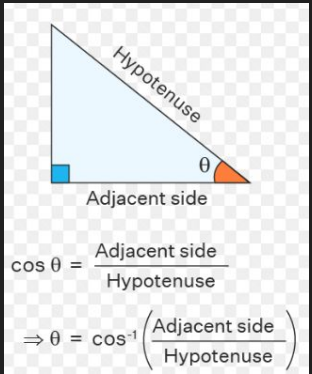


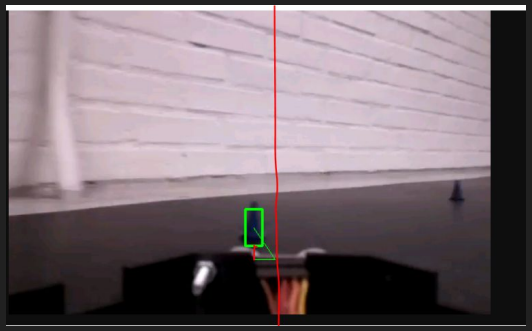
Algorithm UI with OpenCV



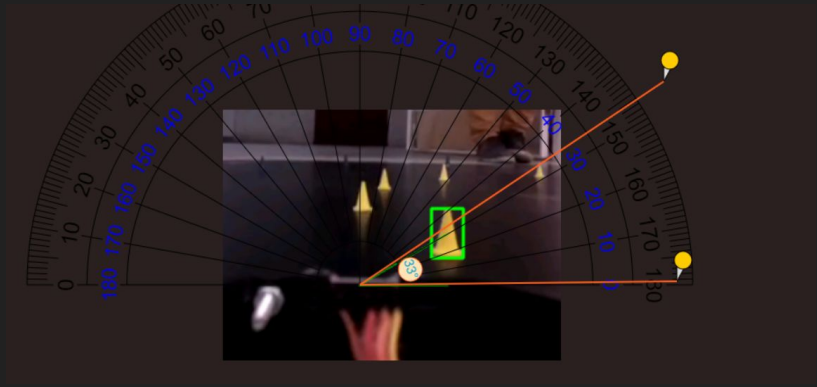


Cos inverse: Calculate vL and vR





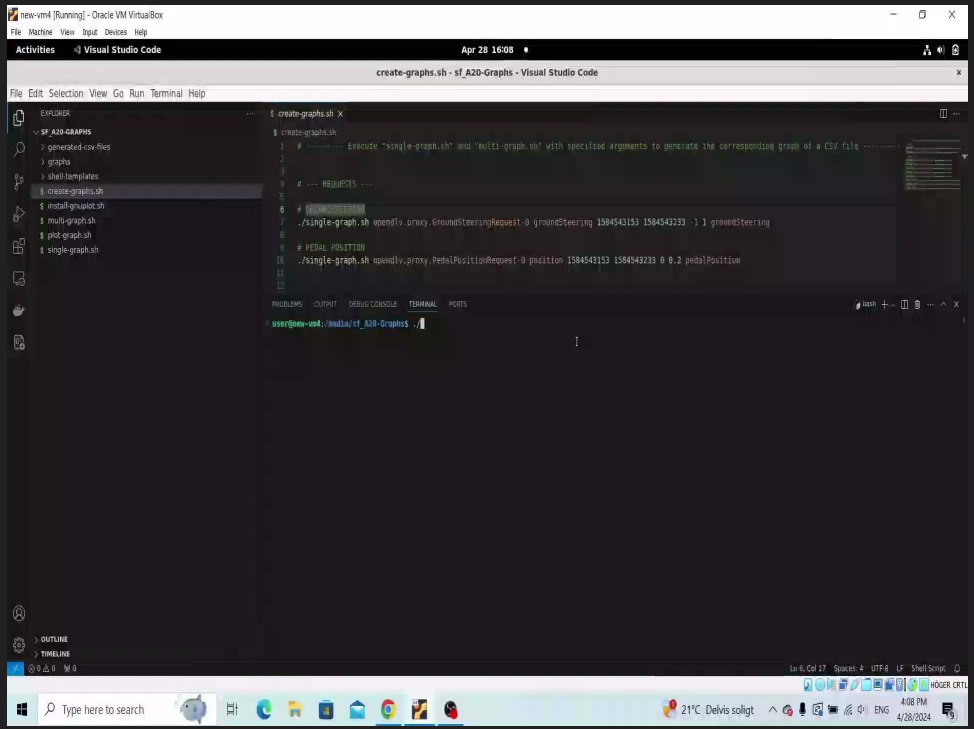
33 degrees → Confirmed



Automation: Plot all csv files as graphs using shell script

Enables us to:

- Visualize output
- Analyze mathematical correlations between variables



Linear Regression

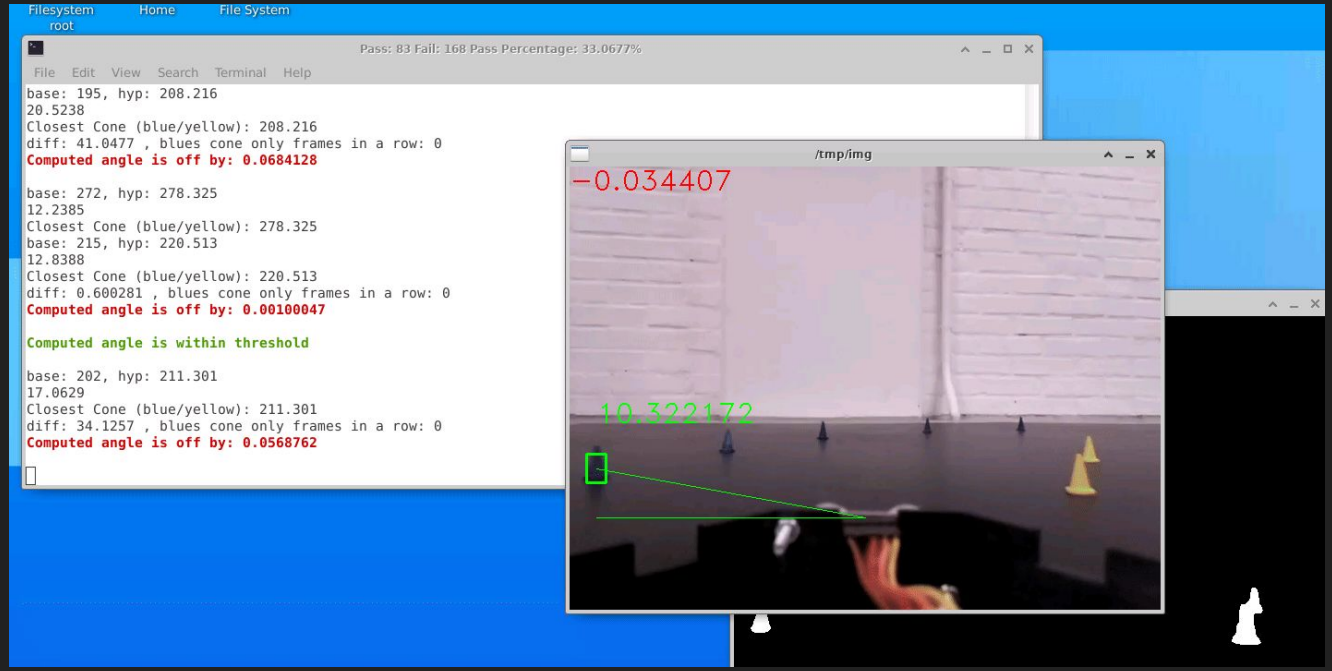
In our project we tried to use linear regression on the angles of the cones as inputs, and a steering angle as output.

We thought linear regression would be a good choice for our project because linear regression is good for finding direct correlations between sets of data, like housing prices and square footage, or cone angle difference, and steering angle of the car.

However when tested practically, we found that our formula wasn’t that accurate, often giving angles that weren’t close to the expected ones.

Our own test cases

NOTE: *In the video below, we count the angle as correct when the groundSteering is 0 and get 30.5% accuracy*

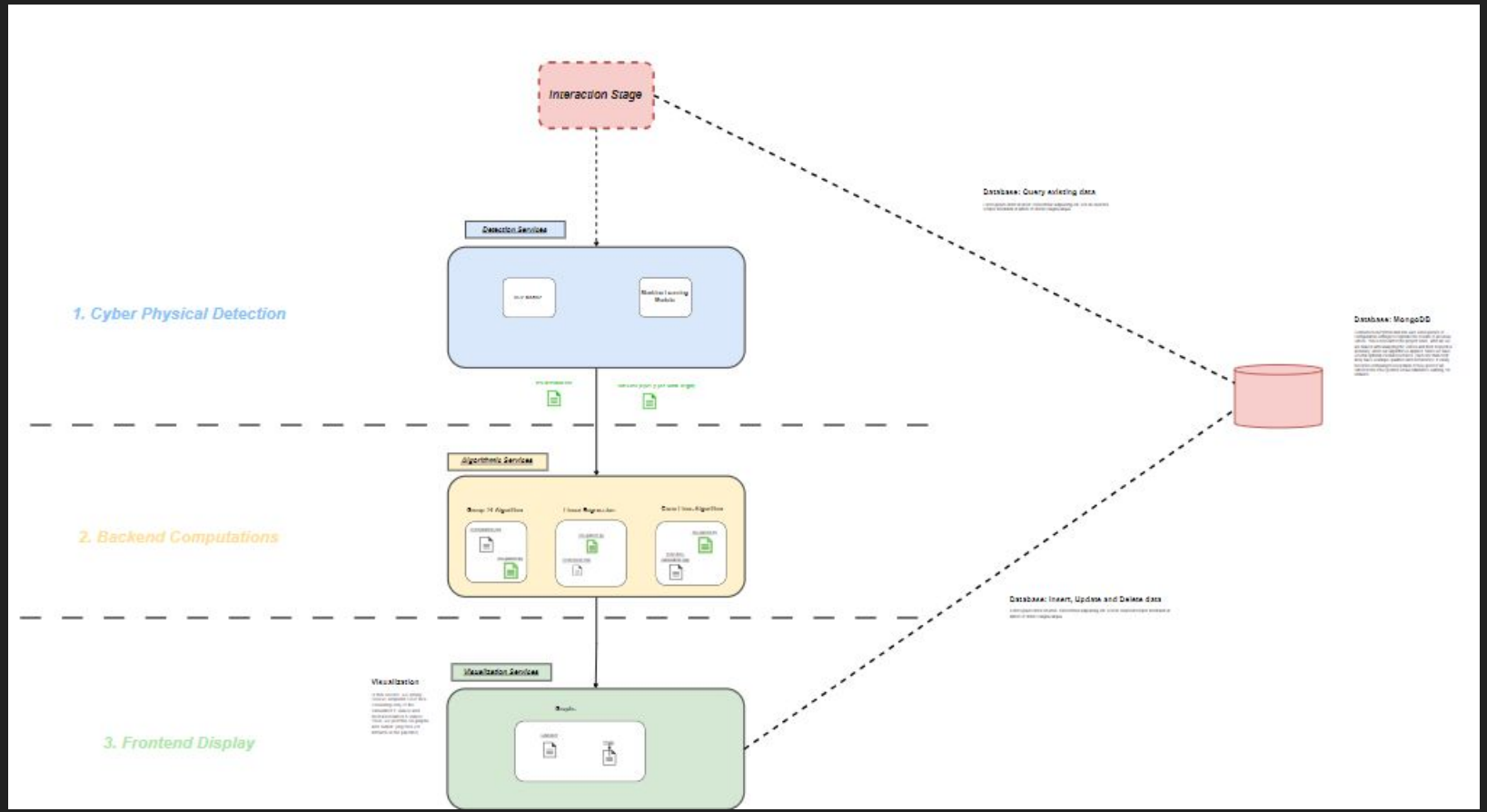


Questions Section

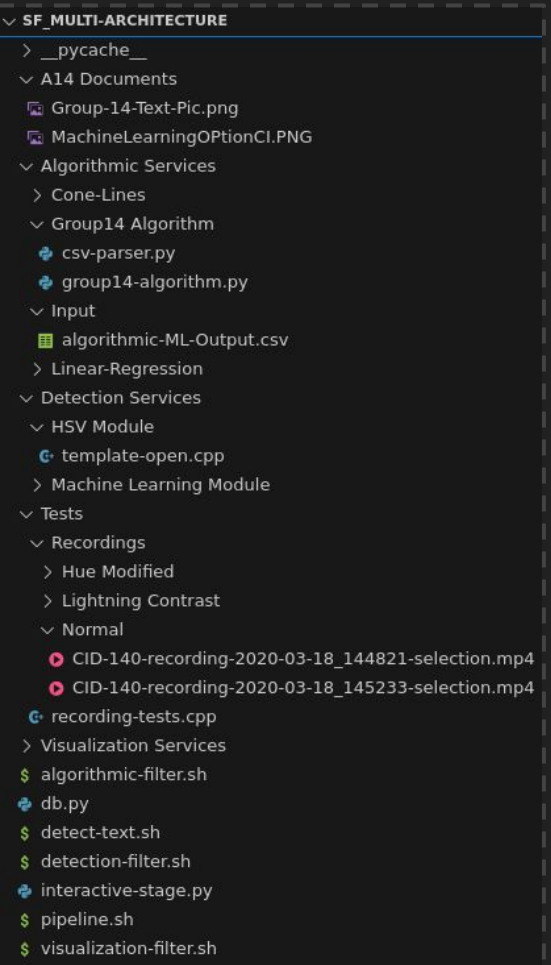
NOTE: We are aware that the specified requirements for this course doesn’t concern the following features, for which we have questions pertaining to their relevance of this project. This is simply us having some ideas that we want to implement because out of interest.

Question: Would do you consider this architecture to be relevant to this course and project?

Architecture: Pipe-and-filter and Service-oriented



Question: Would do you consider this architecture to be relevant to this course and project?



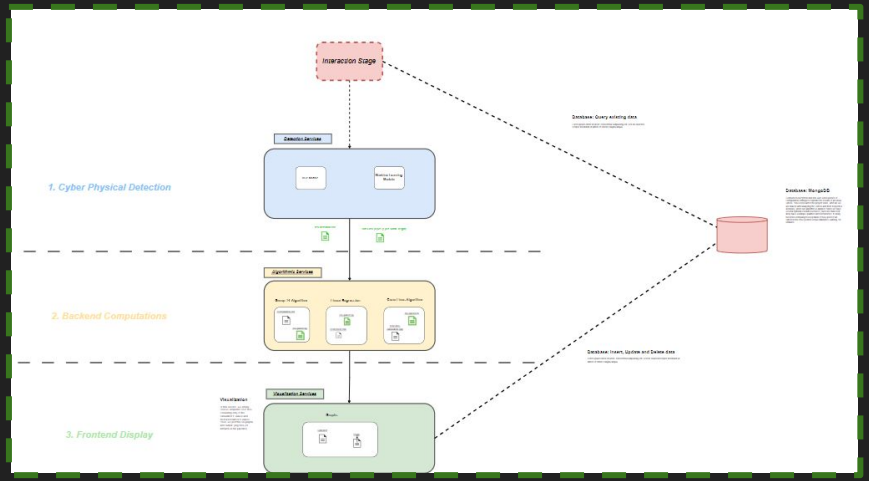
Start system: ./pipeline.sh

```
$ pipeline.sh
1   # pipeline.sh
2
3   # This script starts the entire system's pipe-and-filter architecture, by running
4   # the 3 services or filters in sequential order
5
6
7   videoToPlay=CID-140-recording-2020-03-18_144821-selection.mp4.mp4 # The name of the .mp4 file in the "/Tests" directory
8   detectionModule="Machine-Learning" # The name of the "Detection Service" module to use. Possible Values: ("Machine-Learning", "HSV")
9
10  # FILTER 1
11  ./detection-filter.sh ${videoToPlay}
12
13  # FILTER 2
14  ./algorithmic-filter.sh ${detectionModule}
15
16  # FILTER 3
17  ./visualization-filter.sh
```

Communication: CSV files
Code Flow Management: Shell scripts

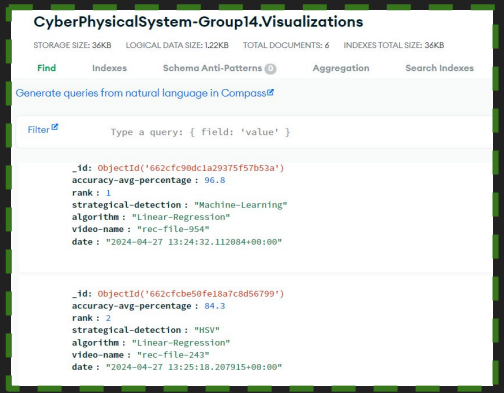
Question: Do you consider our database to be relevant to the project?

Enter “Interaction Stage” by passing an optional parameter “interactive=enabled” when running the docker container. In this way, your automated tests won’t interfere with this implementation



- algorithmic-filter.sh
- db.py
- detect-text.sh
- detection-filter.sh
- interactive-stage.py
- pipeline.sh
- visualization-filter.sh

DB Object
* Configurations
  - Navigate back to older versions
  - Compare current run against previous ones



- Potential future update: Have the latest commit tag as an attribute of each object instance: Allows developers to easily go back in time

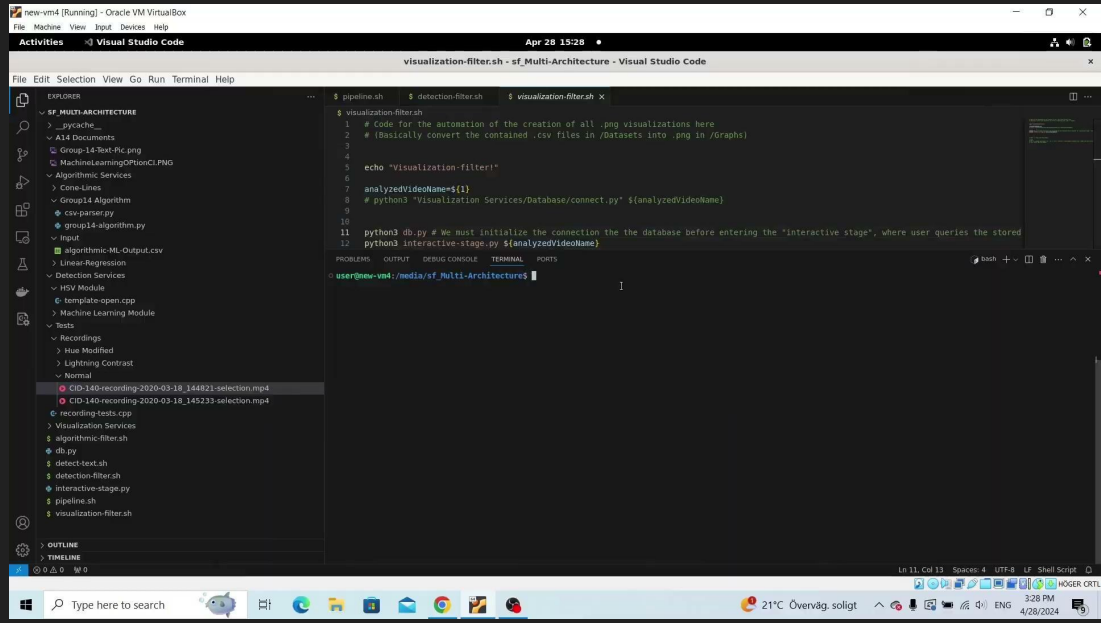
Explicitly add “interactive=enabled” here to enter “Interaction Stage” to allow interaction with DB:

```
docker run --rm -ti --net=host --ipc=host -e DISPLAY=$DISPLAY -v /tmp:/tmp my-opencv-example:latest --cid=253 --name=img --width=640 --height=480 --verbose
```

Question: Would do you consider this database to be relevant to this project?

Currently supported database features:

- *Insert document*
- *Query by ranking based on accuracy (top 1, 3, 5, 10)*
- *Query by numerical accuracy value (at least 50%, 75%, 90%)*
- *Query by strategical-detection (HSV, Machine-Learning)*
- *Query all by video name (ex: myVid.mp4)*

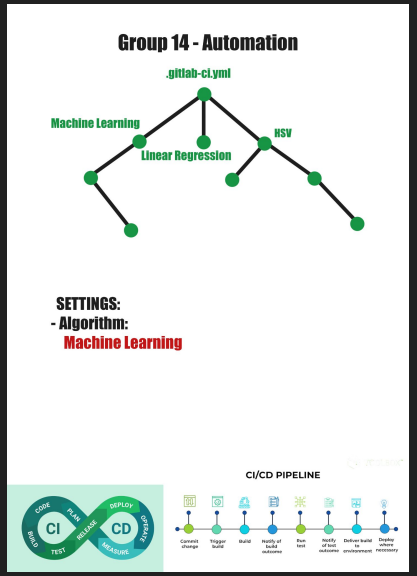


Potential future update: *Add as an attribute of the DB object, a link to the latest commit at the point when the system was executed*

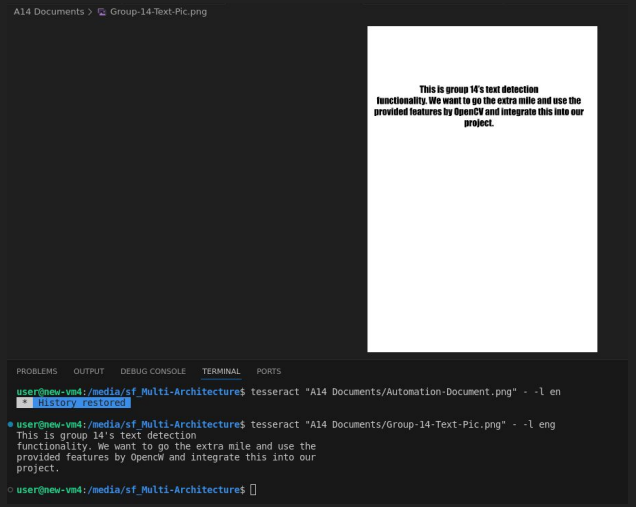
Question: Would do you consider our integration of text detection to be relevant to the project?

Integration:

Idea:



Current progress - Done:



Integration:

detectedText = tesseract …

if detectedText contains “Machine Learning”:
./detection-filter.sh “Machine Learning”
else:
./detection-filter.sh “HSV”

Other questions:

Yolov8 & Shared Memory

Is it fine if we inclure the yolov8 model implementation as a “execution pipeline path” that is only reached if optional arguments say so? I.e if someone runs it without providing our custom commands, the system will automatically execute the HSV and disregard the yolov8. We are wondering because we aren’t yet sure if we can connect yolov8 to the shared memory via Python. If we would fail, this would be our solution to still have it in the project but not officially using it as a detector for your tests

Face Recognition with OpenCV

- *No progress on this feature: We want to know your thoughts on our ideas of integrating this feature to our system*
- *Idea: Give it a .png, and if it detects one of group 14’s developers, it returns true, otherwise false*
- *Integration:*