Adam Veraszto, Zoltan Bereczki

CodeLikeABosch Software Development Challange

Project Documentation

09/30/2023

Budapest

1. Introduction

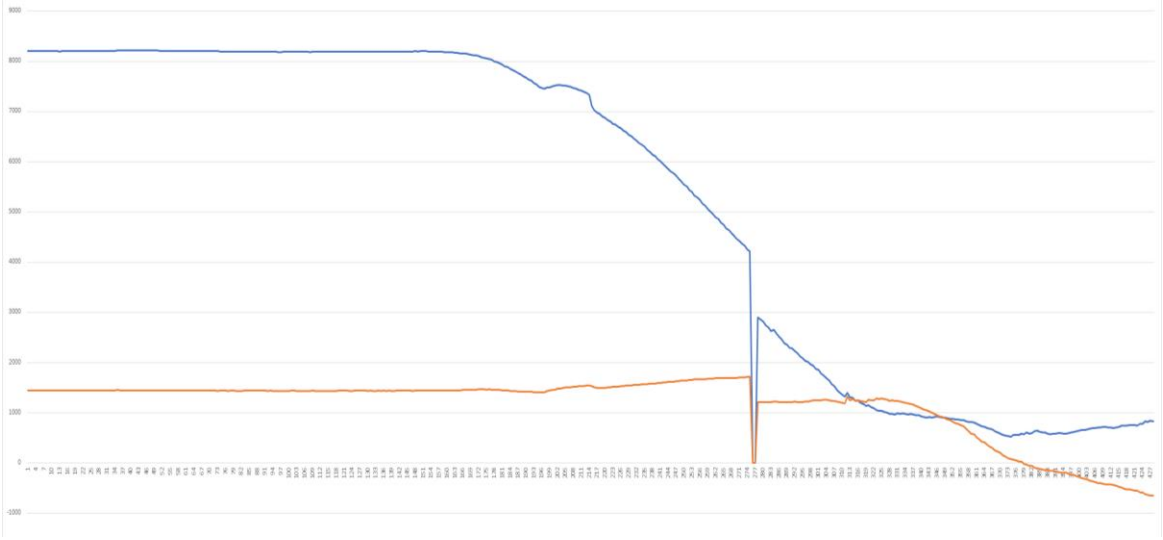
The SW Challenge is a three-part task that provides an opportunity to acquaint ourselves with the work performed by developers that steer the transportation industry towards autonomous vehicles.

In the first part of the Challenge, we get to understand what a Perception developer does. In this task, we need to create an environment based on available data. The data includes our vehicle's speed, coordinates of up to 4 other objects (X and Y coordinates), the objects' velocities (lateral and longitudinal velocities), our vehicle's angular velocity, and timestamps. It's essential to consider only one object as relevant, and we must filter out the others, as deleting datasets is not an option.

In the second part, we delve into the work of a Localization developer. Based on the created environment, we must decide what situation is occurring. We have to choose from three predefined scenarios.

2. Theory and Implementation

First, we plotted the data to examine its noise level and whether it contains many outliers, etc. This was necessary for initially understanding the data, visualizing what is happening, and deciding the order of the filter to be used. For example, this is what we received for the coordinates of the first object.

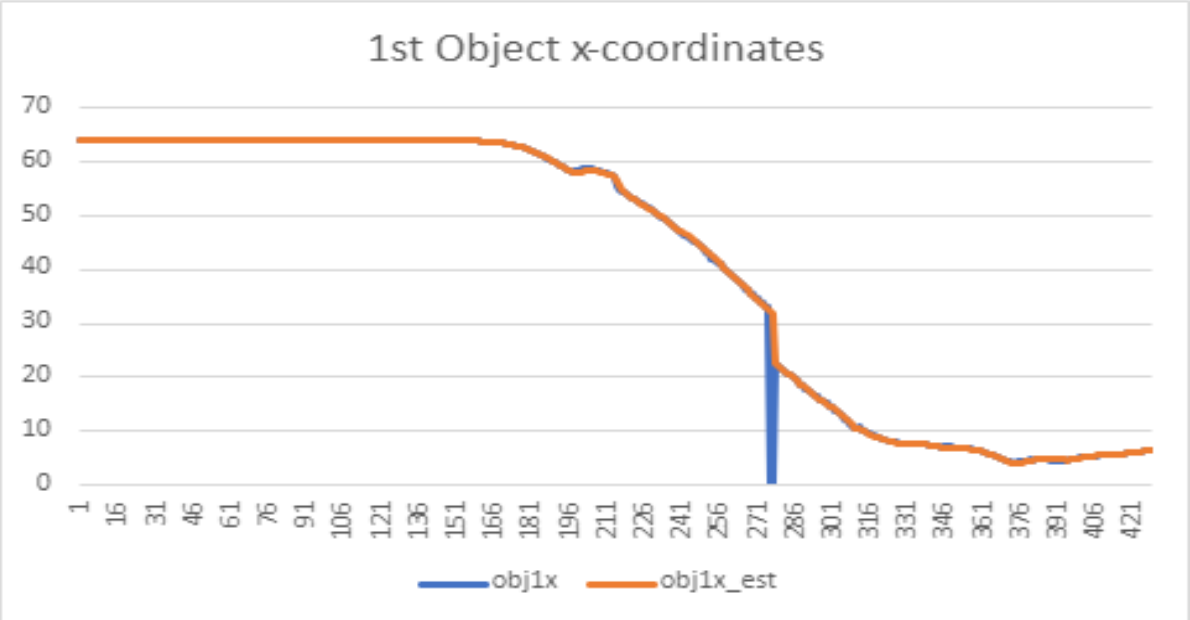


Later, we noticed that there are 0 values, but these often only last for 1-2 frames, and the object continues its path afterward. However, there are cases where we no longer see the object, or it may be obscured by another object. Afterward, we concluded that it might be worth processing the data as pairs of columns and creating conditions to determine whether an object "returns" or another object enters through the sensor.

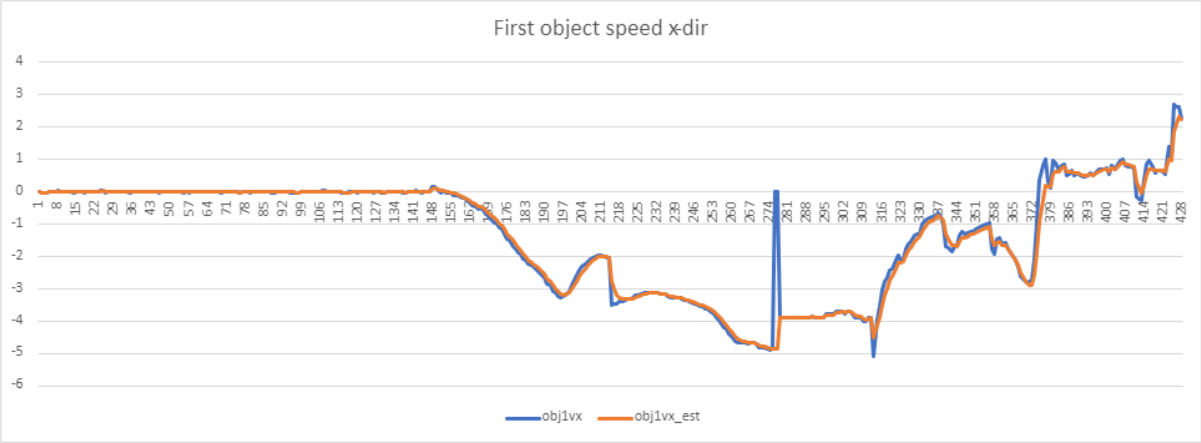
We handled data visualization through a frontend using React and Three.js backed up by a backend using PostgreSQL, Celery and Django. The project was dockerized. The coordinates and velocities of individual objects were displayed in textboxes that are always facing towards the camera. We have chosen the relevant object via making it’s displaybox red and adding a text to it which says the exact scanerio that happened.

Then, using the previous data, we made predictions for the current values of the car and individual objects. This model uses a relative system with the car's origin, which is then transformed into a stationary system for easier calculations, and finally converted back.

Upon closer examination of the data, we decided to use a second-order low-pass filter, which not only utilizes the previous data but also data from one step further back in the calculations. This way we were able to smoothen out the data and avoid the natural noise of the dataset. The first object looks like this on the x-coordinates after filtering the dataset:

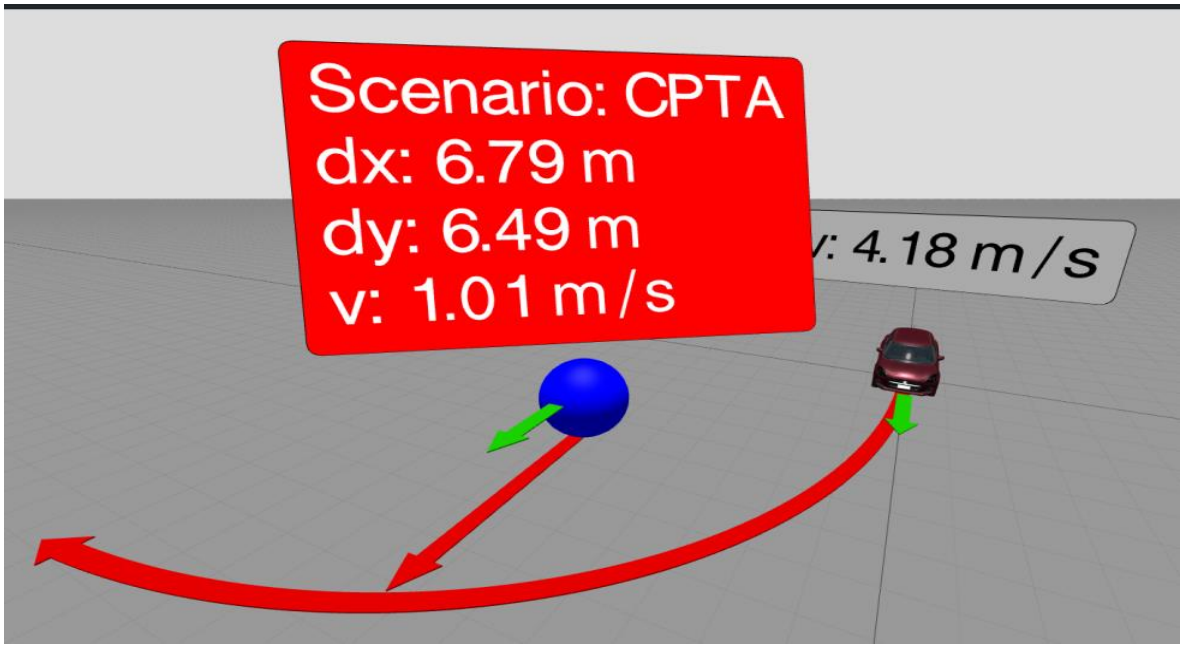


It was even better on the speeds:

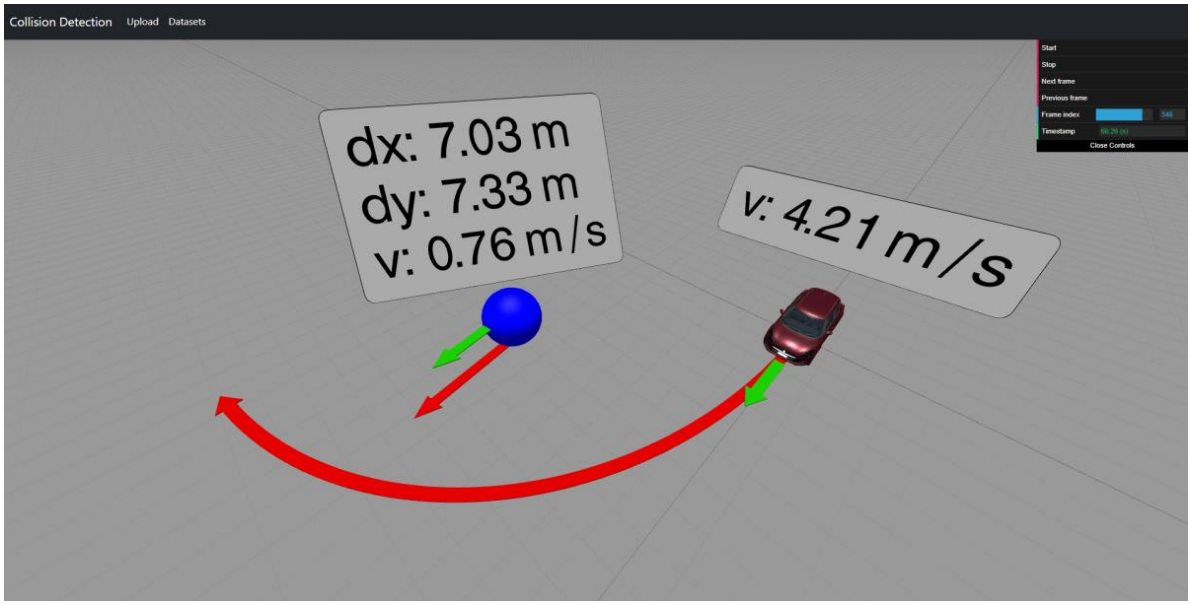


You can see it detects if an object gets out of vision but shortly comes back as we added a little time delay before deleting it.

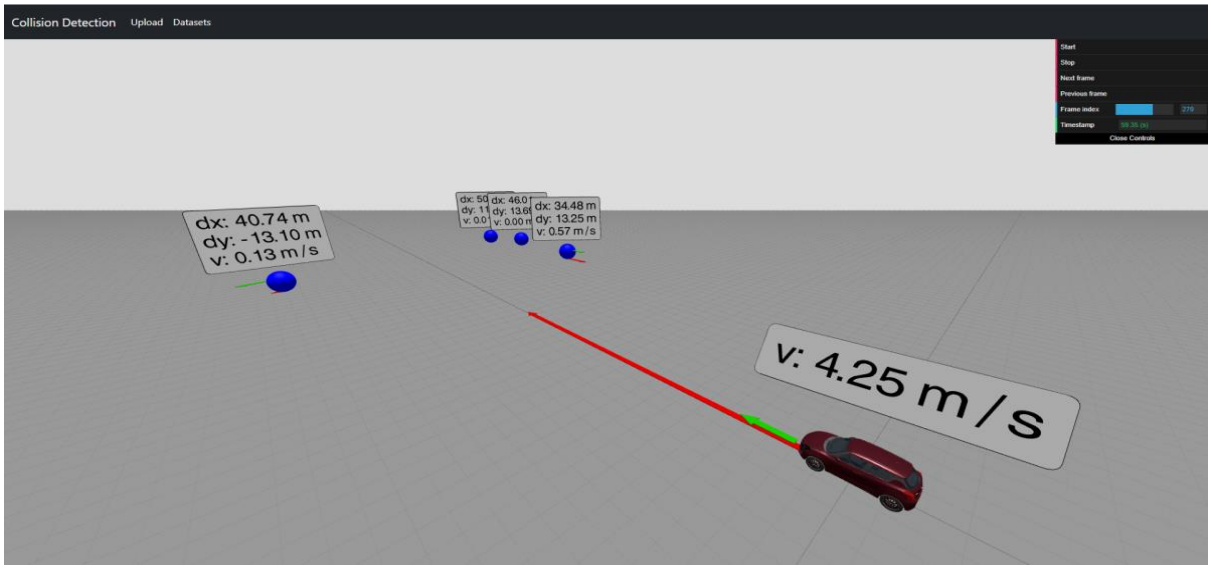
We marked the relevant object by using the the trajectories of the car and the objects and see if they intersect.
If they did then we checked if they get to the intersection point in a given time so there is a chance for collison and decided the type of it.



The green arrows indicate the directions of the objects and the red arrows indicates the trajectories.



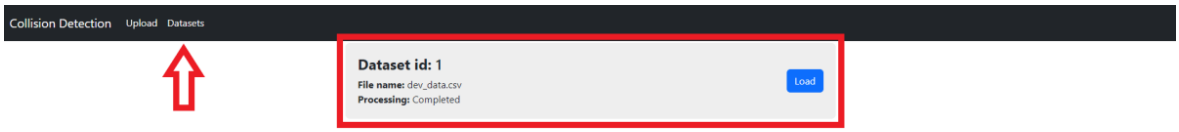
Values connected to the obejcts can be observed in their floating displays.



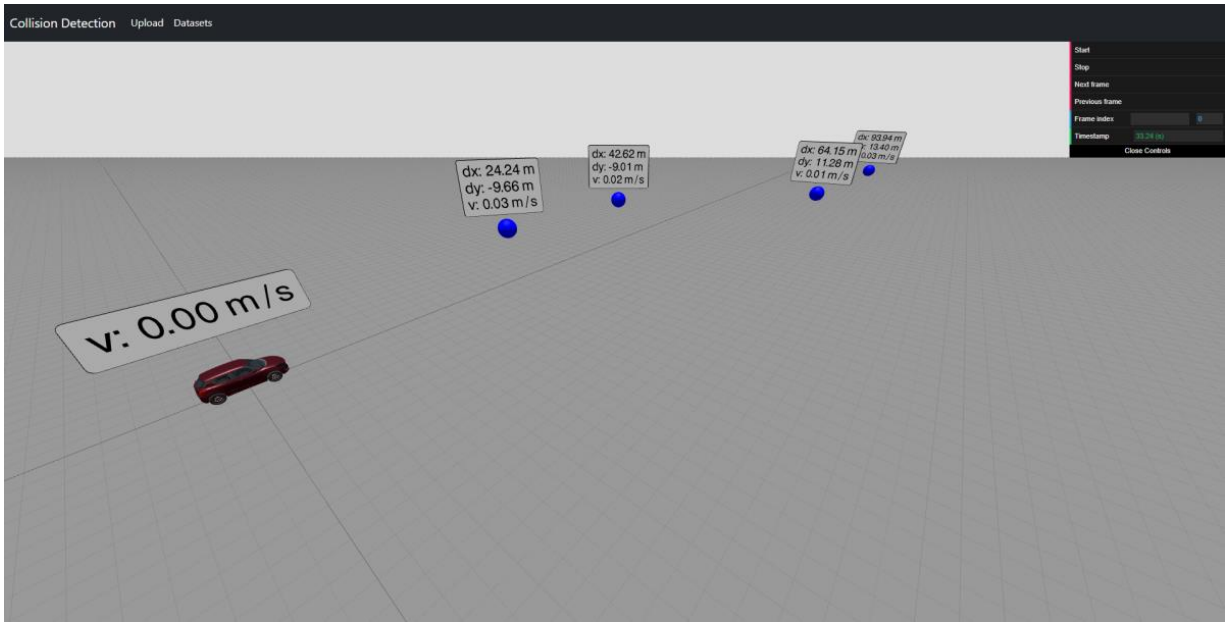
3. User manual

One can use the gui via updating a file, loading it, then playing it.

You can see that in the upload menu you have to choose your file then press the upload button, then on the dataset menupoint you have to load the dataset (shown in the pictures)



After that, you can enter the player which looks like this:



You can move around freely with right click and holding it down. You can choose the desired angle with the left click. The player menu is self-explanatory.