Driver 2.0

Team Members:

Vasu Bhog

ChaoYang Zhu

Chunjie Pan

Advisor: Wen-Ben Jone

Goals Statements

Background

Billions of people drive on the road with no assistance from smart devices to aid them, resulting in a continuous increase in crashes over the decades. Our project aims at creating a simple device that uses computer vision and machine learning to detect distracted drivers and conditions on the road. This will decrease the probability of crashes, as well as increase the awareness of the driver.

Goal

Create a device that can detect and notify drivers about dangerous objects and distractions on the road in order to reduce the likelihood of a crash along with increasing safety.

Intellectual Merits

Inadequacy of Current Solutions

There is no device that currently out there that detects objects, tired drivers, and road signs for drivers. There are currently only advanced cars that cost an additional hundreds of thousands of dollars that have sensors and cameras built in the vehicle. Not everyone can afford expensive vehicles, therefore a device that can detect and notify dangerous objects and distractions on the road will reduce the likelihood of a crash along with increasing the safety.

Our Device

Our project will use state of the art hardware and open-source software to detect many different objects using machine learning and computer vision. This will allow us to ensure that the driver is aware of his surroundings at all times through the use of our compact device. Our device is one of a kind on the market, as there is no device to directly help drivers of all ages be safer in any car they own.

Broader Impacts

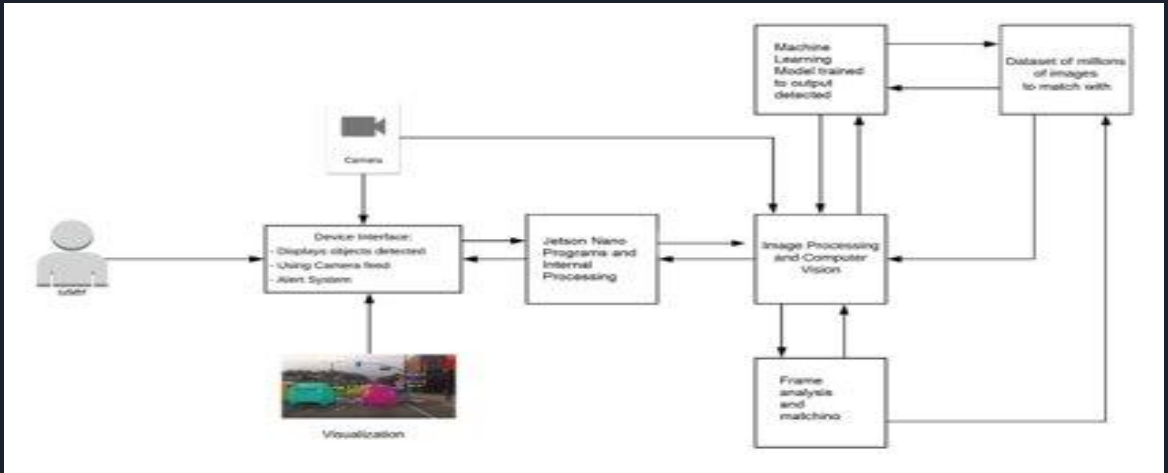
Our project focuses on the safety of the drivers around the world, no matter the vehicle. The global impact of our project is direct and can be exponential due to our project software and low costs. Our users age ranges from teenagers who are starting to drive to elderly individuals who may need additional help driving. As a result, our device will ensure that the driver is aware of their surroundings and of any dangerous conditions while driving. This will decrease the probability of crashes and injury, as well as increase the awareness of the driver.

Design Specifications

System Overview

A device that can detect and notify drivers about dangerous objects and distractions on the road in order to reduce the likelihood of a crash along with increasing safety.

Design Diagram



Technologies

- Software
  - YoloV3 - The latest variant of an object detection algorithm YOLO, A pre-trained model. We used preset weight and other configurations that YoloV3 has defined to train our prediction models.
  - OpenCV - A library of programming functions mainly aimed at real-time computer vision. We used for image processing, outline and send the detected object to the interface (monitor)
  - PyTorch - A open source machine learning library that can be used for computer vision and natural language processing. We used to train our prediction model with the preprocessed dataset.
- Hardware
  - Jetson Nano - A machine we run our application on. This is an NVIDIA Developer Kit that is designed to run multiple neural networks in parallel for applications like image classification, object detection, segmentation, and speech processing.
  - Cameras - We used a Road Camera and a Jetson Camera to record and live feed the video/images to our application.

Milestones

1. Milestone 1 *Dec 30 2019 - Feb 11 2020:*

   Create a program that detects multiple road objects in a video/live feed.

1. Milestone 2 *Feb 12 2020 - Feb 22 2020:*

   Connect the program to a camera to read live feed data.

1. Milestone 3 *Feb 24 2020 - Mar 02 2020:*

   Setup the program with Jestson Nano.

1. Milestone 4 *Mar 03 2020 - Mar 19 2020:*

   Create an alert interface/program that triggers when object detected and ensure the user is aware.

1. MileStone 5 *Mar 20 2020 - April 13 2020:*

   Test our product and implement front facing camera.

Results

We were able to create machine learning model using a pre-trained network that can detect objects and dangerous conditions at a high accuracy and will show to the driver in few milliseconds. This program is demonstrated in our Driver 2.0 Demo and Testing video (link in Appendix). The program was run on a Jetson Nano standalone and plugged into a car using an adapter that is able to supply enough battery. Then the device is placed on the dashboard of the vehicle for the driver to clearly see the dangerous conditions and objects on the road. Furthermore, there was another program that we were building using a secondary camera that would utilize our computer vision program for face directional detection to ensure the driver is always aware while driving. We were able to implement this only partially as there were issues with the camera connection to the Jetson Nano. We found that both programs could work simultaneously while giving the driver and other passengers feedback in real time.

Challenges

1. Using the Jetson Nano with our own configurations
    a. This was difficult as we need to ensure that what we use on our laptops to train and test models could transfer to the Jetson Nano.
    b. We found that Jetson Nano was only compatible with a few our softwares which we used and then built our models off Jetson Nano framework and capabilities.

2. COVID-19
    a. It was quite difficult to meet and work on our project due to the coronavirus, as we had only one device that we could use. However, we found that through splitting up certain tasks we were able to bring together our project in the end, even without physically meeting. We were unable to complete and entire casing as we were not able to use 1819 3D printers, and had to use a box instead.

3. Model Training
    a. In the beginning of our testing, the accuracy of our model was very low. However, after discovering a few pre-trained models and doing transfer learning we were able to achieve high accuracy and fast feedback

4. Full system Setup
    a. We had trouble finding all the items we needed in the beginning of our project, however, after continuously developing we found that we could buy the items we needed by researching what we need to code next. As a result, our item would arrive as we were completing the software side and then test the hardware.