15-112 Term Project Design Proposal

Road Detection

By Tianyi Li

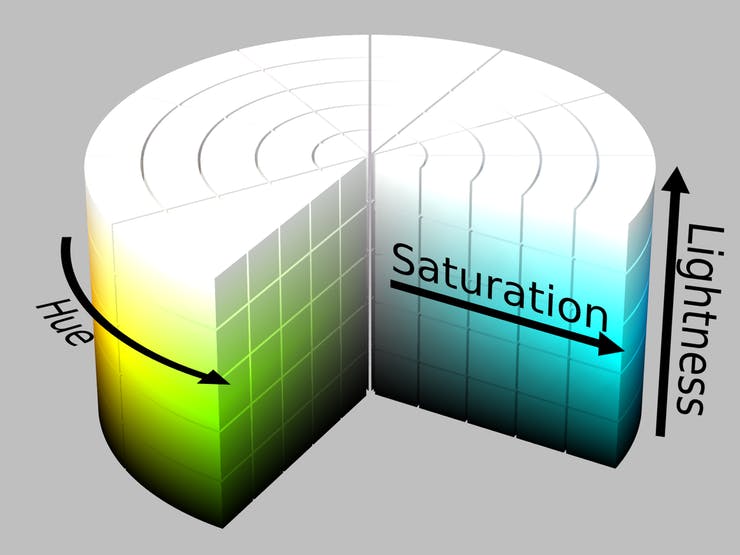
**Project Proposal**

* **Description**:

The name of the project is called “Road Detection”. This is a project that mainly focuses on using OpenCV to detect roads from photos/videos provided by a car’s dash camera. Once the program can detect lanes from photos and videos, more features will be implemented. Such as, car detection, pedestrian detection, cross roads detection, etc.

* **Competitive analysis:**

Lane detection projects is becoming more and more popular along with the advances of self-driving cars. There are many projects about lane detections with similar core concepts: detecting roads from the data provided. This project, similarly, focuses on the goal of detecting lanes by using camera feeds. However, there are varieties of approaches on the topic of detecting lanes. Some converts the feed given by camera into grayscale then uses “canny edge detection” algorithm to capture the change in gradient (sharp changes in color) in order to find the lanes. Some projects convert the color scale from BGR (blue, green, red) into HLS (hue, lightness, saturation) that can easily detect, for example, a yellow line’s unique characteristic: Hue between ~10-50.

 (Picture of HLS color space)

In addition to the lane detections, which is what most projects are about. This project will try to implement more detections on different objects occurring in the camera feed, like cars, pedestrians, road signs, etc. In Short, the main differences of lane detection, fundamentally, are the different methods one can implement to find the locations of lanes. Since this project wishes to implement more features other than only detecting lanes, efficiency is crucial. Throughout the project, I will be trying out different methods and, in the end, find a method that best suits my goal.

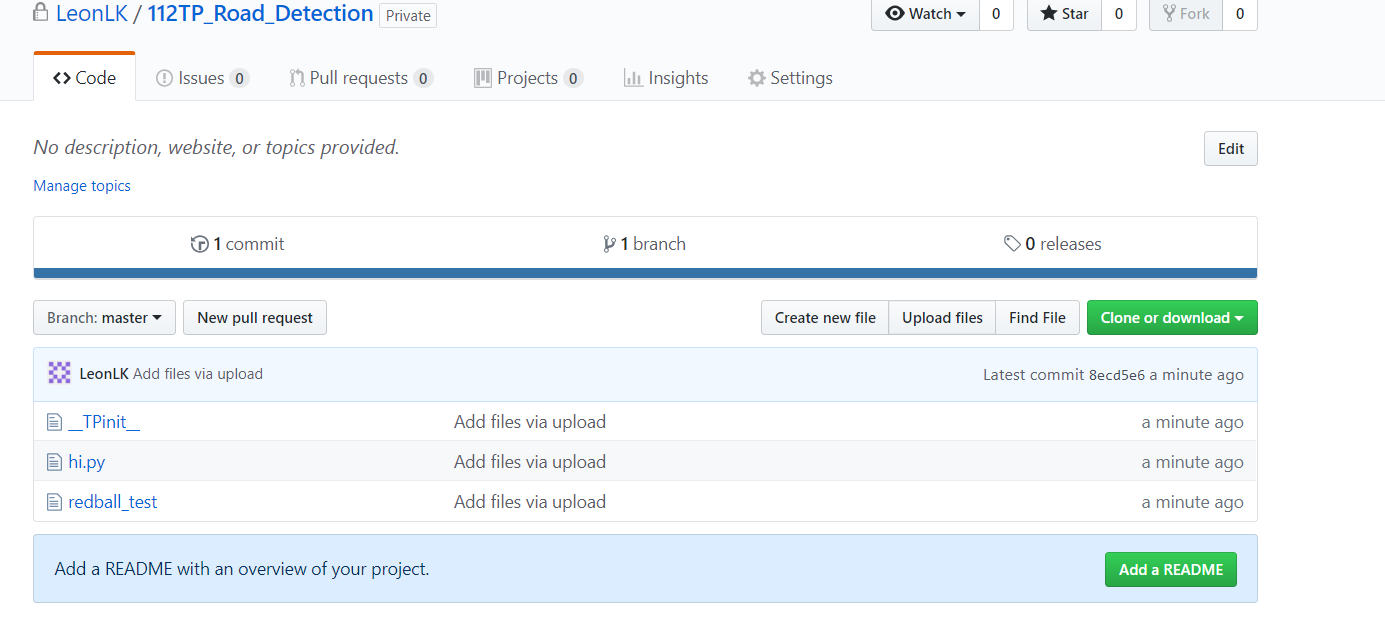
* **Structural Plan:**
  + The main file is lane(straight) detection
    - Code for detecting lanes from photos
    - Code for detecting lanes from video
  + Test file that I use to practice and try out different algorithmic approaches and implantations.
  + Implement additional code in the main file for curved lane detection
  + File for vehicle detection, after completing lane detection
    - Code for detecting specific shape and area of interest
  + File for road sign detection (mainly stop sign)
    - Code for detecting different shape
  + File for pedestrian recognition
    - Code for face detection
    - Code for body detection
    - Cascade used (Haar Cascade)
* **Algorithmic Plan:**

The core part of this project is lane recognition. The hard part is to consistently and quickly display the detected lane in real time while a video feed is provided. Initially, the project will be focused on simply detecting straight lanes from a given photograph rather than video. Either BGR to Gray or BGR to HSL color space method will be implemented for faster lane color detection. Color filtering will be applied, focusing on specific colors of the lanes (yellow, white, etc.). Region of interest (ROI) will also be written to reduce the focus and narrow down the region for better efficiency. Canny Edge Detection is used to detect sharp color transitions (large gradient change).

The difficult part within lane detection is the Hough Line Detection. By using Hough Line Detection, the lines outputted by Canny Edge Detection will be filtered by some sort of slope equation to check whether it is the actual lane’s line or some exterior factors. Afterwards, this concept will be applied to videos instead of images.

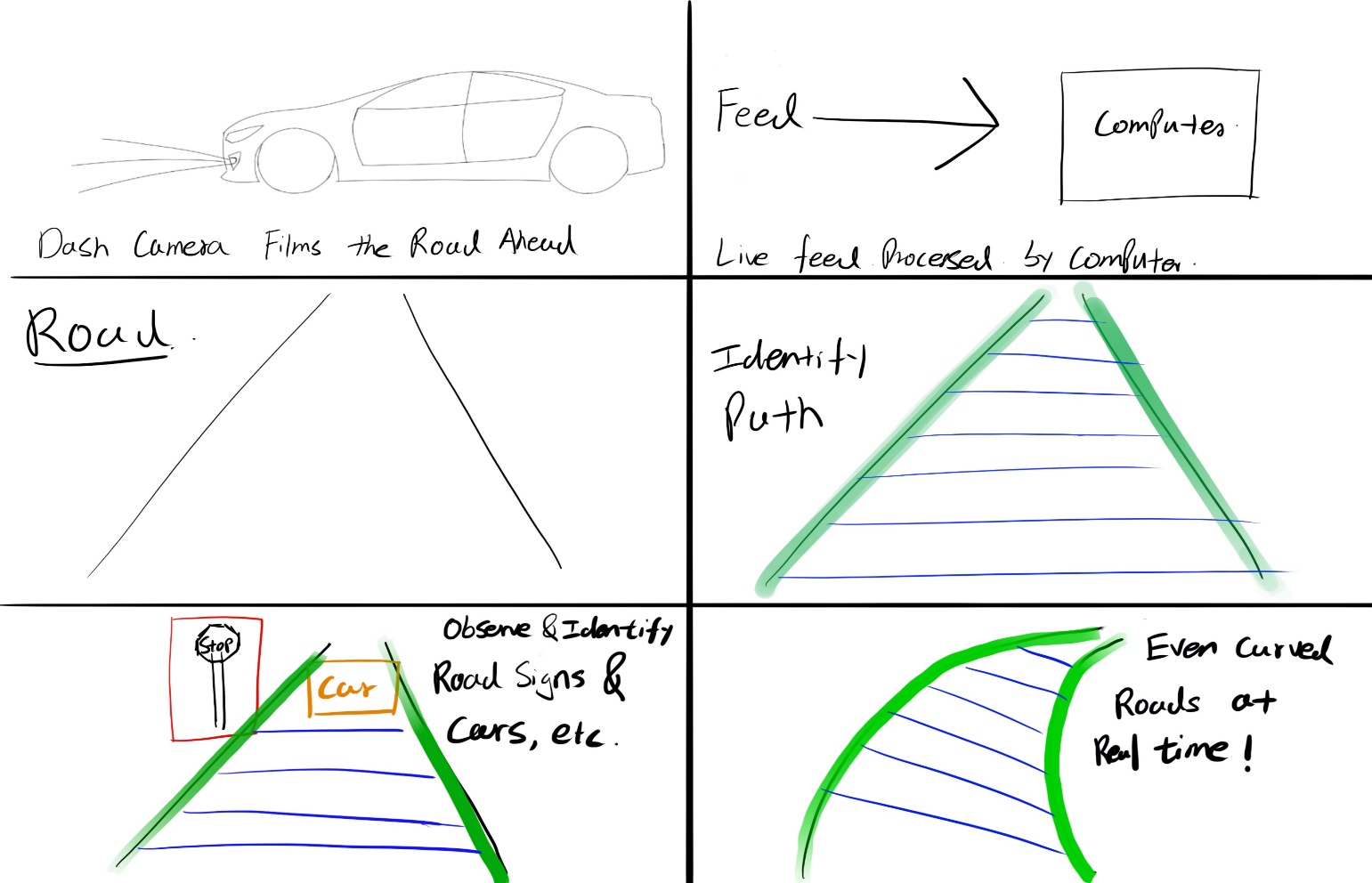
When the base code is completed, additional code will be added to make lane detection for curved roads. Also use object detection method to detect cars and road signs. Implements cascade (Haar Cascade) and body detection methods to detect human pedestrians.

* **Timeline Plan:**
  + 4/16 – 4/21: Finish the code for simple lane detection by image.
  + 4/21 – 4/24: Convert it from reading image to video. If time permits, polish the base code and make the display look nice.
  + 4/24 – 4/27: Implement curved lanes detection + car detection
  + 4/27 – 4/30: Adds road sign detection and pedestrian detection
  + 4/30 – 5/2: Final wrap up. Organize all the files and prepare demo video.
* **Version Control Plan**
  + I have been constantly uploading all my project files onto my GitHub.



* **Module List**
  + None

**Storyboard**

****

**TP2:**

I’ve chosen to use BGR to Gray instead of BGR. No significant plan changed.

**TP3:**

For tp3, I have modified my straight lane detection to detect curved lanes without making the whole program crash. I’ve sampled and averaged the previous 20 frames of data in order to better visualize the curvature, I also tweaked the region of interest, so it doesn’t read too far ahead and getting unwanted data. I’ve also implemented a not fully developed pre-trained car detection feature. It works nicely when there’re cars close to the view and not that accurate for objects that are further away from the camera. I’ve also improved by UI by letting user to select between 3 videos that I chose to be best fit for my code. Since different cars have different camera locations and height, which would affect the performance of the code. I’ve also allowed users to choose between only displaying road detection feature, the core of the project, also car detection only or both at the same time. The background color and font color are also improved for better visual experience.