Final Project Proposal
COMP4102A

Automotive Safety Suite

Authors:

Conner Bradley – 101073585
Christian Belair – 101078744
Adam Payzant – 101082175
Carleton University

February 5, 2020

1 Summary

The goal of this project is to create an automotive safety suite using computer vision. Our goals are to create a suite of services to track pedestrians, road signs and stop lights, as well as tracking the driver's eye movements to detect distracted driving. A secondary goal is to make the system highly modular, so new components can be easily added in or create interactions between modules.

2 Background

2.1 Gaze Detection and Tracking

Gaze detection and tracking has many positive implications for driver safety, namely to detect if a driver is distracted or drowsy. The US department of transportation found that in years 2011-2015 an overall 2.5% of fatalities were caused by drowsy driving [**?**], and for distracted driving in 2018 it was found that 8% of fatalities were distraction-affected [**?**]. Currently there are a wide variety of well-established and novel techniques for gaze tracking as found by a survey by Chennamma and Xiaohui [**?**].

2.2 Pedestrian Detection and Tracking

Pedestrian detection and tracking is an important aspect in driver safety. Drivers must be aware when one or more pedestrians are moving around their vehicle as to ensure their safety. Stimpson, Wilson, and Muelleman research into pedestrian fatalities shows that from 2005 to 2010 the fatality rate of 116.1 per 10 billion vehicle miles driven had increased to 168.6 in 2010 [**?**]. OpenCV includes an implementation of the Histograms of Oriented Gradients (HOG) which can identify a person within an image or video. This creates a basis for identifying pedestrians. The implementation details in terms of human detection are found in this paper[**?**]

2.3 Road Sign Tracking

Road signs are vital to road safety. Stop sign violations accounted for approximately 70% of all crashes.[**?**] In addition, in some areas, speed limits are lowered in unexpected areas pose great risk for a driver, even when they're striving to follow the speed limit. While much research into this area is focused on using machine learning, some computer vision exclusive implementation have been studied.[**?**]

3 The Challenge

This space has a number of challenges. While any one component of this project has many implementations even within the computer vision space, making them all work together in real time on mid to low end hardware. In addition, gaze tracking with a single camera is a more difficult area with no simple, of-the-shelf implementation in openCV.

4 Goals and Deliverables

4.1 Primary Goals

The primary goals for this project is to create a suite of car safety features which are solved using computer vision. The three main safety features involves gaze tracking, road sign and traffic light detection, and pedestrian detection. Each play a role in distracted driving incidents where a driver's gaze might not be focused on a pedestrian, road signs, or traffic lights when a driver should be.

4.2 Stretch Goals

The primary stretch goal for us to add an interaction between the modules. As a proof of concept, we would implement a system to ensure a driver has actually looked in the direction of a pedestrian or sign. An additional goal would be to validate it running on a Raspberry Pi Zero as a proof it can be embedded into an actually.

4.3 Evaluation

As this system is made up of many components, each element must be evaluated individually for success.

- Gaze Tracking
  - Accuracy: Must be able to differentiate between different eye states and edge cases (eye direction, eyelids open/closed) with 80% success rate
  - Must be able to give near real-time feedback on eye states, as these will be used by the safety system.
- Pedestrian Tracking
  - Accuracy: Minimum of 80% correctly classified pedestrian
  - Must be able to detect a pedestrian in <500ms after appearing in frame
- Road Sign Tracking
  - Accuracy: 80% of signs correctly detected
  - Be able to detect a sign in <500ms after appearing in frame
- Raspberry Pi Port
  - All targets must still be met when running the code on a Raspberry Pi Zero
- Module Interaction
  - Using the pedestrian or sign tracking, the system can detect if the driver has looked in the direction of the detected object after 5 seconds.

Each component will be evaluated using a test framework of mock video footage in a variety of scenarios and edge cases. The video footage will have timestamps and other metadata regarding certain events, thus we can correlate a time delta between when an event happened and when a given component registered that event.

5 Schedule for Completion

Generally speaking, this project is split into 3 sub-projects for gaze tracking, pedestrian tracking, and road sign tracking. These 3 components will be developed seperately; however, near the end they will all be integrated together into a comprehensive suite of computer vision technologies to aid with automotive safety.

Week	Task Summary	
Week 1	**Adam**	Early research regarding sign, number, and word detection
	Christian	Finding papers regarding person detection
	Conner	Preliminary research, scope out papers in eye tracking
Week 2	**Adam**	Finalize theory for implementation
	Christian	Design structure of pedestrian tracking component, identify necessary components
	Conner	Scaffold project structure, determine entities related to task, build a common object model
Week 3	**Adam**	Begin implementing sign detection
	Christian	Begin implementing pedestrian tracking, gather test data
	Conner	Begin implementing eye tracking, gather test data
Week 4	**Adam**	Continue implementing sign tracking, begin testing
	Christian	Continue implementing pedestrian tracking, use test data to establish a base line
	Conner	Continue implementing eye tracking, integrate with test data to establish a test bed
Week 5	**Adam**	Finalize initial sign tracking, begin building API to work with tracking data
	Christian	Finish implementing pedestrian tracking, start creating API to work with pedestrian tracking data
	Conner	Finish implementing eye tracking, provide a high-level API for working with eye tracking functionality

Week 6	**Adam**	Begin testing and optimizing sign tracking
	Christian	Begin testing pedestrian tracking, gather accuracy data, improve implementation
	Conner	Perform testing, gather result data for tracking accuracy, determine edge cases, make improvements where applicable
Week 7	**Adam**	Integrate API with main application
	Christian	Integrate pedestrian tracking API with main application, create UI in main application to interface with API
	Conner	Integrate high-level API with main car safety application, begin making UI for working with feature
Week 8	**Adam**	Continue UI for sign tracking
	Christian	Complete UI for pedestrian tracking API
	Conner	Finsih UI for interacting with feature, build notifications
Week 9	**Adam**	Finalize sign tracking integration and begin real world testing
	Christian	Continue week 8 work
	Conner	Continue week 8 work
Week 10	**Adam**	Work on stretch goals, prep for final demo
	Christian	Help work on strech goals / rasberry pi port, help prep for final demo
	Conner	Work on strech goals / rasberry pi port, prep for final demo
Week 11	**Adam**	Finalize and package project
	Christian	Help package final deliverables and continue prepping for final demo
	Conner	Package final deliverables and perform demo