**UI System Design Document**

**For**

**EcoCAR UI Team**

Team members:

* Teja Tiriveedhi
* Nathan Rose
* Sabrina Yepez
* David Fadini
* Raymond (Sky) Kwasneski

|  |  |
| --- | --- |
| Version/Author | Date |
| 1/EcoCAR UI Team | 10/1/20 |
| 2/EcoCAR UI Team | 10/26/2020 |
| 3/EcoCAR UI Team | 11/29/2020 |
|  |  |

TABLE OF cONTENTS

[1 INTRODUCTION 3](#_Toc530500649)

[1.1 Purpose and Scope 3](#_Toc530500650)

[1.2 Project Executive Summary 3](#_Toc530500651)

[1.2.1 UI System Overview 3](#_Toc530500652)

[1.2.2 Design Constraints 3](#_Toc530500653)

[1.2.3 Future Contingencies 3](#_Toc530500654)

[1.3 Document Organization 3](#_Toc530500655)

[1.4 Project References 4](#_Toc530500656)

[1.5 Glossary 4](#_Toc530500657)

[2 UI SYSTEM ARCHITECTURE 4](#_Toc530500658)

[2.1 UI System Hardware Architecture 4](#_Toc530500659)

[2.2 UI System Software Architecture 4](#_Toc530500660)

[2.3 Internal Communications Architecture 4](#_Toc530500661)

[3 HUMAN-MACHINE INTERFACE 4](#_Toc530500662)

[3.1 Inputs 5](#_Toc530500663)

[3.2 Outputs 5](#_Toc530500664)

[4 DETAILED DESIGN 5](#_Toc530500665)

[4.1 Hardware Detailed Design 6](#_Toc530500666)

[4.2 Software Detailed Design 6](#_Toc530500667)

[4.3 Internal Communications Detailed Design 7](#_Toc530500668)

[5 EXTERNAL INTERFACES 7](#_Toc530500669)

[5.1 Interface Architecture 7](#_Toc530500670)

[5.2 Interface Detailed Design 8](#_Toc530500671)

[6 UI SYSTEM INTEGRITY CONTROLS 8](#_Toc530500672)

UI System Design Document

*Overview*

*The UI System Design Document describes the UI system requirements, operating environment, UI system and UI subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.*

# INTRODUCTION

## Purpose and Scope

The purpose of this project is to provide a Human-Machine Interface (HMI) for EcoCAR that is reliable and easy to use for the driver.

## Project Executive Summary

The EcoCAR UI (user interface) Team’s goal is to provide an easy to use interface for drivers of the EcoCAR. The UI system will get drivers to understand and adapt to the current state of the Adaptive Cruise Control (ACC) system. The UI system will do this by providing a “birds-eye view’ of external obstacles that the EcoCAR detects. (These obstacles include cars, trucks, and pedestrians.)

### UI System Overview

The UI system will have a 6.2’’ X 3.06’’ display on the dashboard that will show ACC information. The display will also show external obstacle locations.

### Design Constraints

The UI system needs to be easy to use for drivers who are unfamiliar with it. The UI system is part of a ridesharing service that is frequently used by drivers who may not be comfortable with the UI system.

## Document Organization

This document is organized in a manner that highlights components in the following order: UI system architecture, design, and external interface.

## Project References

ROS data is taken from the EcoCAR GitHub directory.

## Glossary

* ROS: Robot Operating System
* UI: User Interface
* ACC: Adaptive Cruise Control

# UI SYSTEM ARCHITECTURE

## UI System Hardware Architecture

The hardware architecture for this project is to use a Raspberry Pi connected to the display for the UI. The Raspberry Pi needs to be connected to the EcoCAR’s computer by Ethernet or WIFI. A 15V to 5V buck converter will be used to drop the EcoCAR’s voltage to the Raspberry Pi’s voltage.

## UI System Software Architecture

The Software for the UI system will connect the UI system itself to the ACC of the EcoCAR. This will allow the data from the EcoCAR ACC system to be passed to the EcoCAR UI system. The information will be passed through the ROS. This information will then be processed and shown on the display current information including relative obstacle locations determined by the EcoCAR.

## Internal Communications Architecture

The EcoCAR will be connected to the EcoCAR with a network connection (either Ethernet or WIFI). This will allow messages from ROS, specified in the various modules, to be passed from the EcoCAR to the UI. Refer to the UI System Hardware Architecture diagram in Section 2.1.

# HUMAN-MACHINE INTERFACE

This project will be an interface between the EcoCAR and the driver of the EcoCAR. This project will utilize the systems in the EcoCAR to communicate with the UI system. The UI will allow the driver to see external obstacles while inside the EcoCAR.

## Inputs

Inputs include:

* The ROS data from the EcoCAR’s computer.
* The driver.

## Outputs

The UI system will output the location of external obstacles with icons that will be shown in an overhead view. In the center of the display, a static (non-moving) car represents the EcoCAR. All external obstacles that come within a specific proximity of the EcoCAR will be shown on the display. (Proximity distances vary depending on the sides of the EcoCAR.) The UI system will also fade red at the top and bottom of the display to alert the driver when external obstacles are getting too close in front of the EcoCAR.

# DETAILED DESIGN

## Hardware Detailed Design

The Raspberry Pi will need a power source of 5V 3A. This can be achieved from the EcoCAR’s battery and a step-down converter.

There is the possibility of either Ethernet or WIFI for communication, the Raspberry Pi supports both natively.

The Raspberry Pi will have an SD card for storage. A 16GB or greater SD card will be a viable option because it is unlikely to be replaced soon due to storage constraints.

The Raspberry Pi has a 1.5GHz quad-core processor based on ARM.

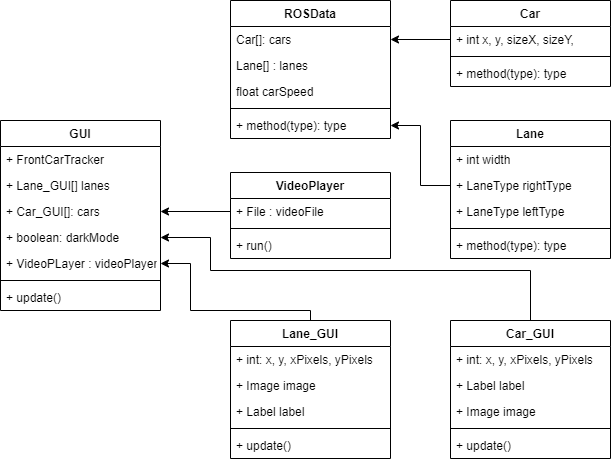
The Raspberry Pi will utilize the Raspberry Pi touchscreen.

The display can be kept at the top of the dash with the use of 8’’ of 18AWG cables while keeping the Raspberry Pi, step down converter, and all communications to the EcoCAR under the dash.

The step-down converter will be spliced into the **Accessory ON** power going to the EcoCAR’s center controls.

The Raspberry Pi touchscreen has a standard resolution of 800x480.

## Software Detailed Design

* 

The current software design has an EcoCAR UI master class which will control what the display shows by running different modules at different times.

The Video Player Module runs when the EcoCAR has a new driver and the EcoCAR is turned on. This allows a basic training for drivers on how the UI system works.

The Car Displayer module will show external cars on the display. This module needs to subscribe to appropriate ROS messages to determine the location of the external cars.

The Car Displayer module uses the Car class as a container for the location of all external cars.

The Proximity Alarm module will look at EcoCAR messages to alert the driver if an external obstacle is too close. This is done by adding a red tint to the top and bottom of the display.

## Internal Communications Detailed Design

The communication between the UI system and the EcoCAR will be through ROS messages over Ethernet or WIFI. These messages are specified in ROS messages in the various modules.

# EXTERNAL INTERFACE

## Interface Architecture

The overall interface architecture will work on the idea of utilizing ROS (Robot Operating System) for communication. The idea is that the training video will be shown when a driver starts the EcoCAR for the first time. This will help the driver learn how to use the EcoCAR’s ACC. A step-down converter will be used to make the 5V Raspberry Pi run off the EcoCAR’s 12V battery.

## Interface Detailed Design

The format of the data shown should be in JPEG. The relative position of obstacles should be in meters.

# UI SYSTEM INTEGRITY CONTROLS

The UI will be active only when signaled to do so. All communications such as the relative positions of obstacles shall be communicated with at least 98% accuracy, updated at an interval of no more than 100 milliseconds.