SMART GPA

JENNIFER CASTILLO – ELECTRICAL ENGINEERING

JORGE DARDON – COMPUTER ENGINEERING

DANIELA OTERO – COMPUTER ENGINEERING

NICHOLAS ZOLLO – PHOTONICS ENGINEERING

Group 25
Spring 2020
Senior Design 2

SMART GARAGE PARKING AID

What is the SMART GPA?

- The SMART GPA is a wall mounted garage parking aid for personal home use
- The system measures the distance from the wall to a car in real time using laser triangulation system
- Colored light indicators will display based on the current distance measurement
- The system will include a touch display which can be used for configuration, personalization, and tuning

Motivation

- Idea came to mind when one of our members hit the wall of a parking spot at a UCF garage.
- Getting a close parking spot in an enclosed space can be difficult and risky to do, so we want to make a product that makes this easier by showing a distance indicator.
- While products for doing this already exist and are employed in parking garages, we want to add convenience features for a personal home environment.
- **We want to make coming home and parking after work relaxing and comfortable!**

Project Goals & Objectives

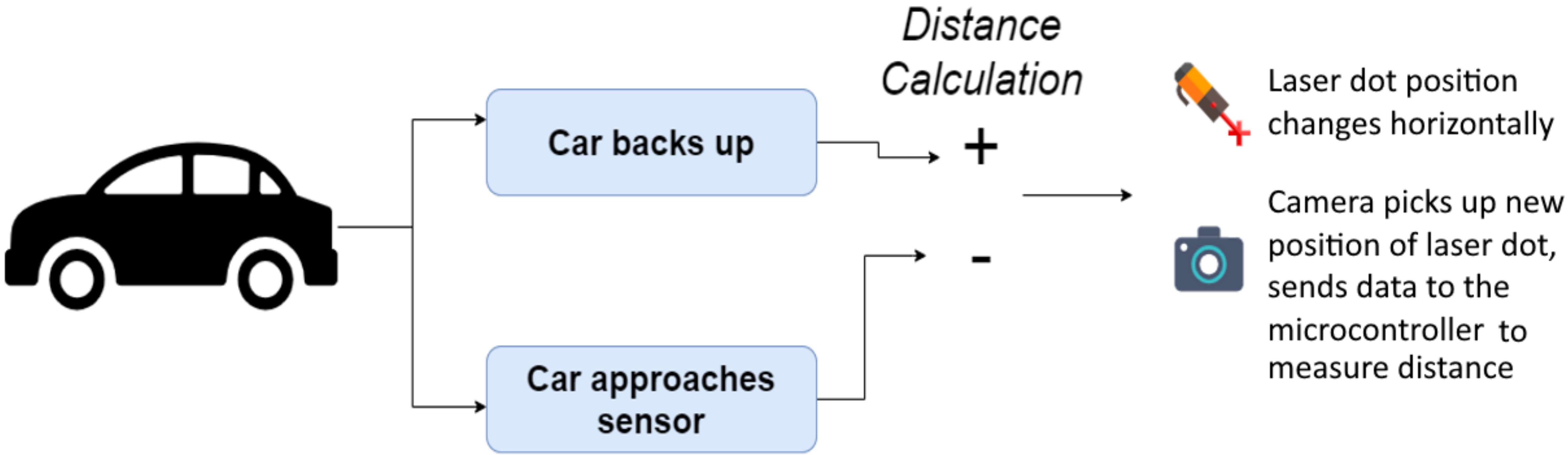
- Make the system simple and convenient to use
- Make distance measurement accurate and fast enough to precisely track a car coming towards the device
- Allow all maintenance and customization of the system to be done directly by the user with the touch display
- After main system is complete, add additional features to improve home user experience and make product more unique

Specifications

Parameter	Specification
Operating Range	1 to 10 feet
Measurement Rate	$\geq$ 20 Hz
Measurement Accuracy	Error $\leq$ 10 cm
Output Power of Laser	$\leq$ 5 mW
Output Power of LEDs	$\leq$ 15 W
Input Voltage	$\leq$ 12 V
Enclosure Size	1.5 m x 0.2 m x 0.5 m

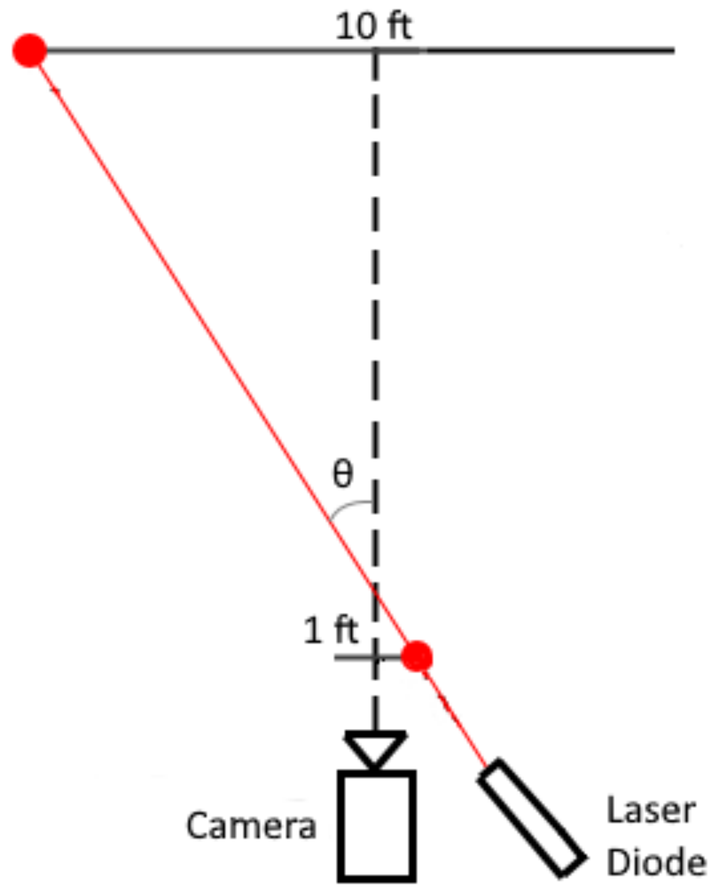
Subsystems & Mechanisms

Use Case Diagram



Laser Triangulation Distance Sensor

- For the SMART GPA's distance measurement, we are using a basic laser triangulation by setting a laser diode at a specific constant angle from a camera and observing the position of the laser dot on the camera to observe the distance.
- This method was chosen as it works well for a variety of ranges, including our specified range of 1 to 10 feet
- Significant image processing work is required to identify the position of the laser dot consistently and quickly



Distance Calculation Equation

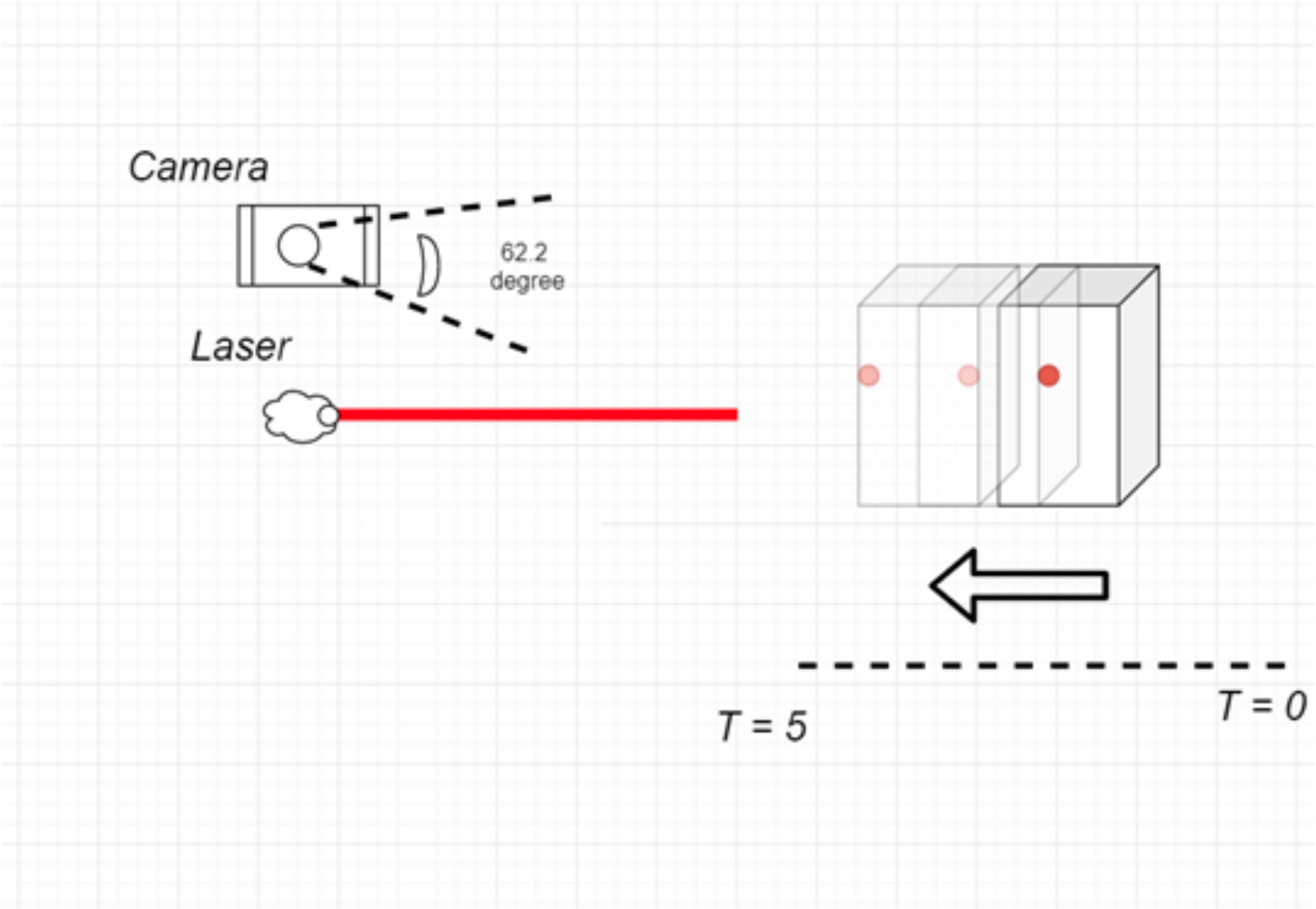
$$z = \frac{d}{\left(\frac{2x}{W} - 1\right) * \tan\left(\frac{HFOV}{2}\right) + \tan(\Theta)}$$

- z = Final measured distance
- x = Pixel position of laser dot on camera

- d = Distance between laser diode and camera
- W = Horizontal resolution of the camera
- HFOV = Horizontal angular FOV of the camera
- Θ = Angle between laser diode and camera

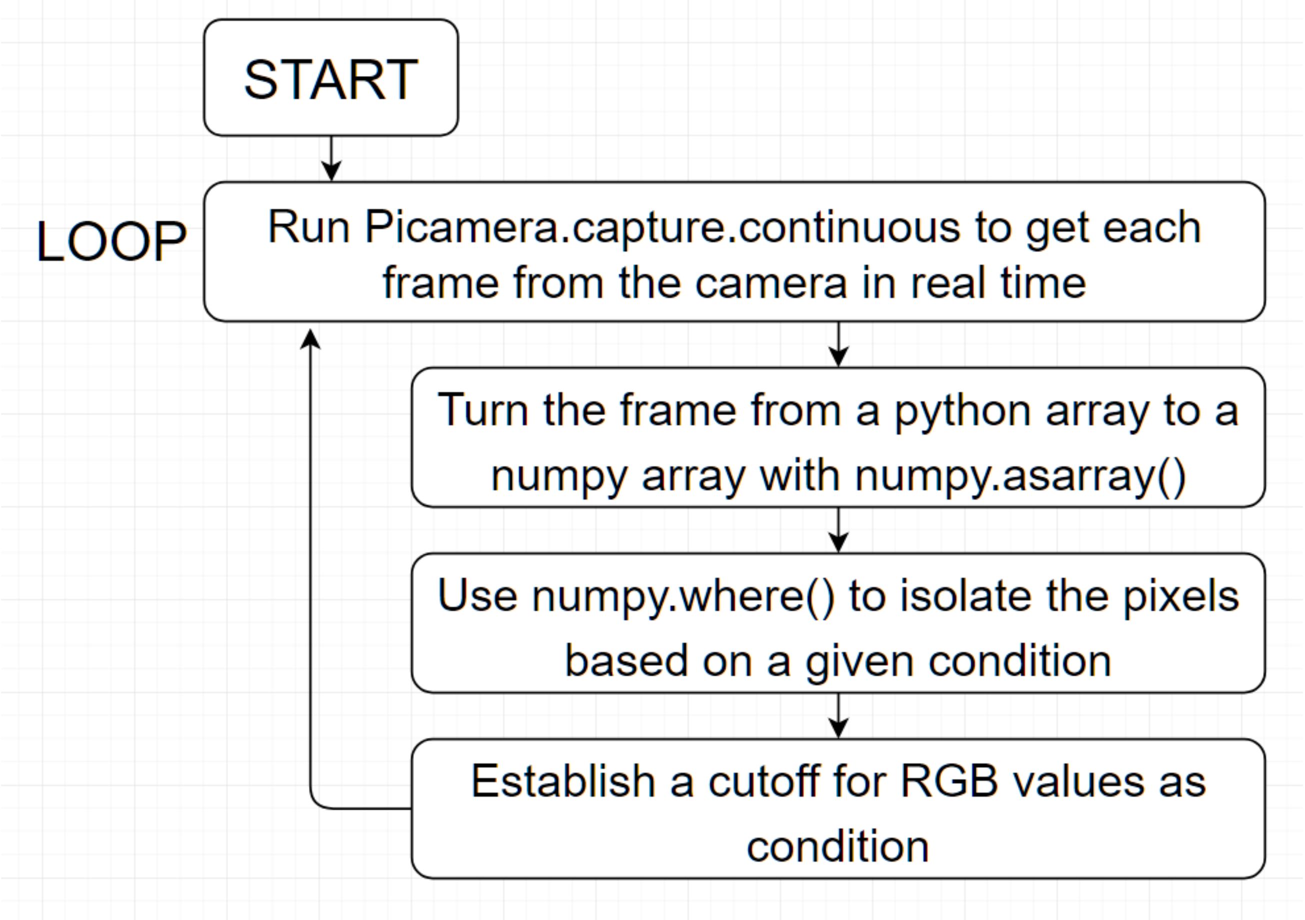
} All of these are constants!

Image Processing - Goal

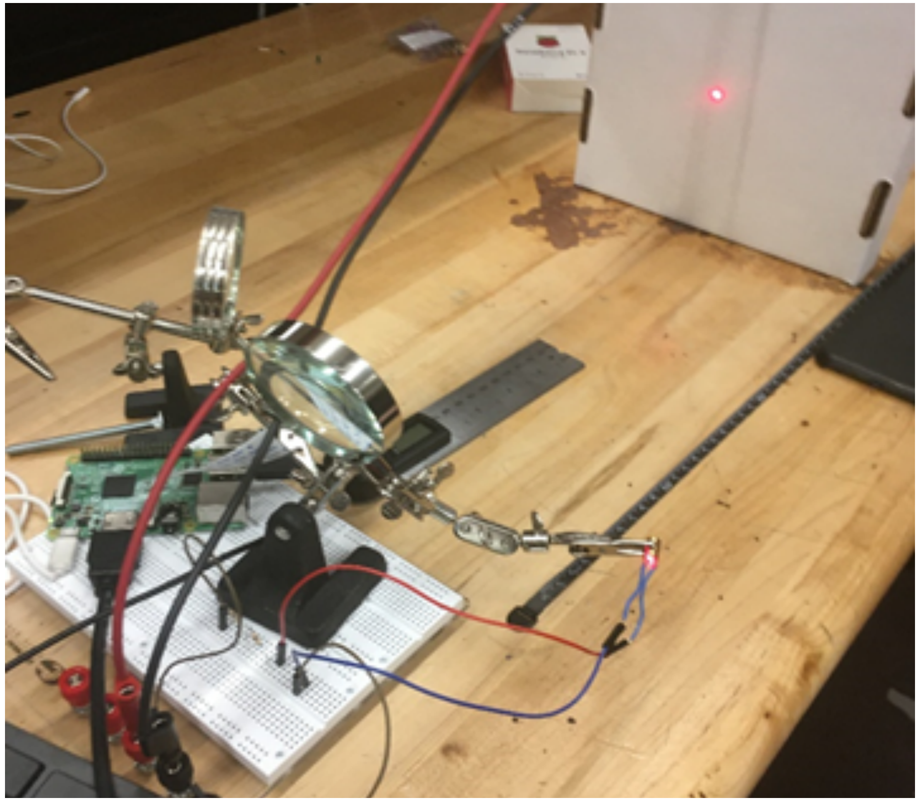


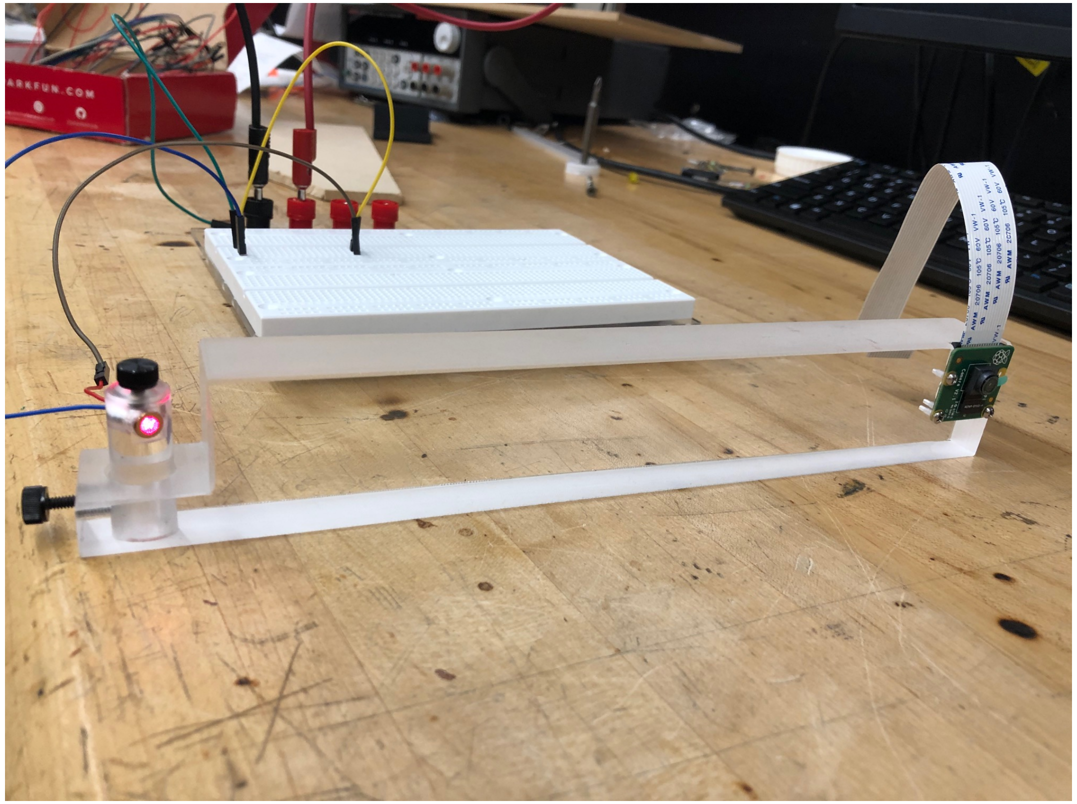
The goal is to be able to detect the laser dot (find the pixels where it is found on the image) on a moving object and run an algorithm to extract the distance

Image Processing – Basic Algorithm



Distance Sensor Testing



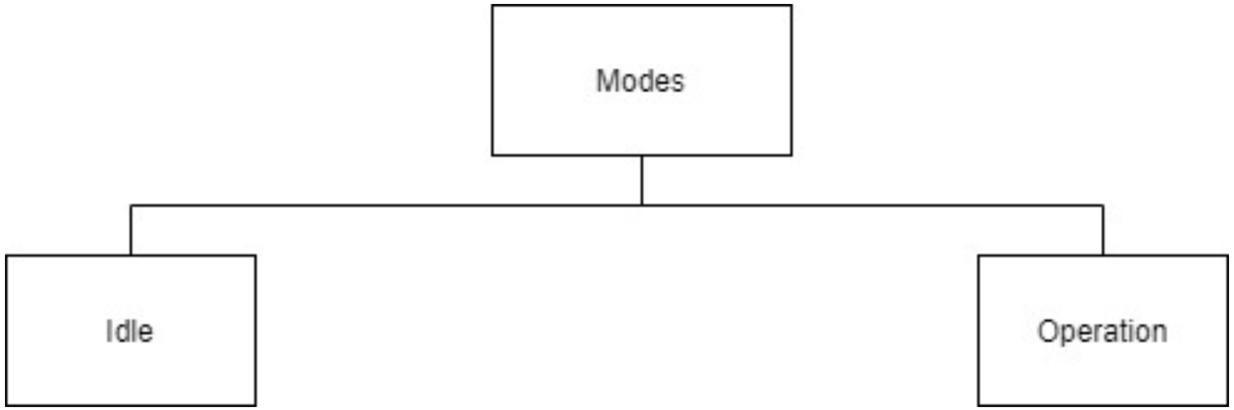


Graphical User Interface (GUI)

Framework Choice

- We have chosen the PyQt framework, with its GUI building tool
- The GUI is the main interface that the user can interact with to modify settings for the entire system

GUI Functionality



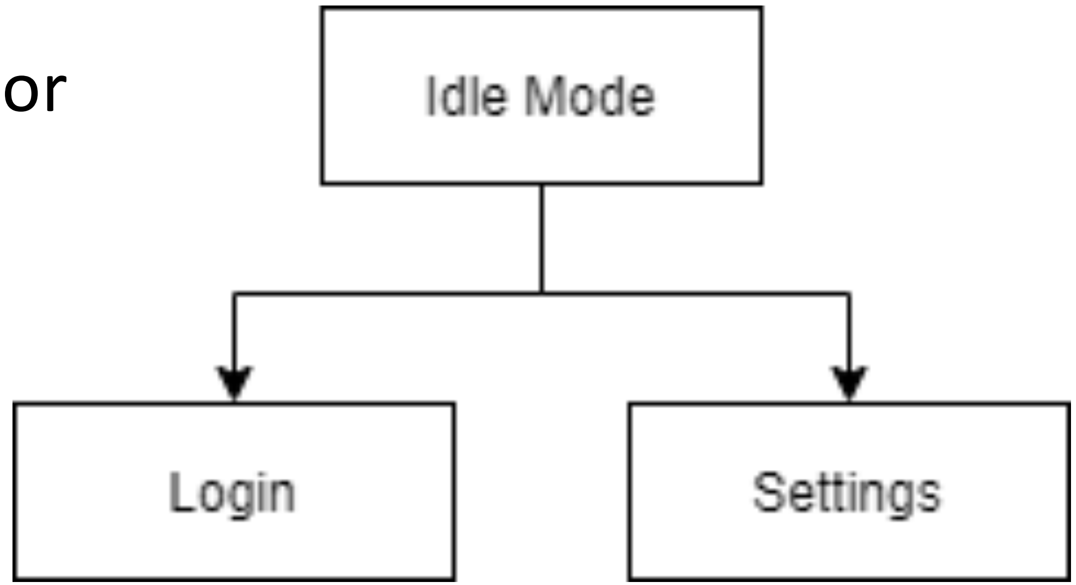
- The GUI will have two basic modes of operation:
- An Operation mode that shows the distance between the vehicle and the LED lights as they are attempting to park
- An Idle mode when the user is trying to access the settings

GUI Continued - Mode Switching

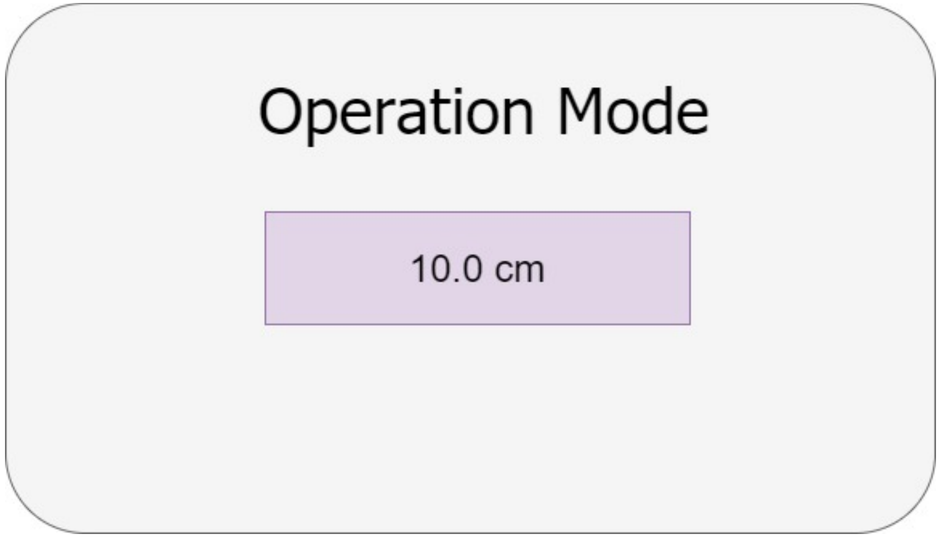
- The Raspberry Pi will run two processes simultaneously in idle mode
- The first one will run the camera continuously, polling to check for a distance under 10 feet, while the other will perform idle mode operations
- When a distance of 10 ft is found, the GUI switches from Idle Mode to Operation Mode
- When distance measurement is either not found or stays about the same for 5 seconds straight, GUI switches from Operation Mode to Idle Mode

GUI Continued - Idle Mode

- Will be using the PyQT framework
- The GUI will display two options to select from in Idle Mode
- It will have the option to go to settings or to login
- Will have account and general settings
- Login will save user's settings
- GUI will also display the date and time



GUI Continued - Operation Mode



- Operation Mode is the GUI state when the user is attempting to park
- The GUI will display a numerical value to the user in large text. This value will be dynamically changing as the user adjusts the position of the vehicle and will represent the parallel distance between the user and the LED lights.
- Operation Mode will work alongside the lights to give user the feedback visually through the lights or numerically if they prefer. The text will be large enough to see and will occupy at least 50% of the total GUI screen space. It will be read-only.

Light Display System

- Light display will be used to alert users when they are close to proximity to the garage wall
- Green light will be displayed when distance sensor senses the car approximately 10 feet away
- Yellow light will be displayed afterwards when distance sensor senses car around 3-5 feet
- Red light will signal the driver to stop when their car is 2-3 feet away
- Light system will have default setting, but can be changed based on user configuration in settings from the GUI system
- The closer the car gets to the garage wall, the faster the light display will blink

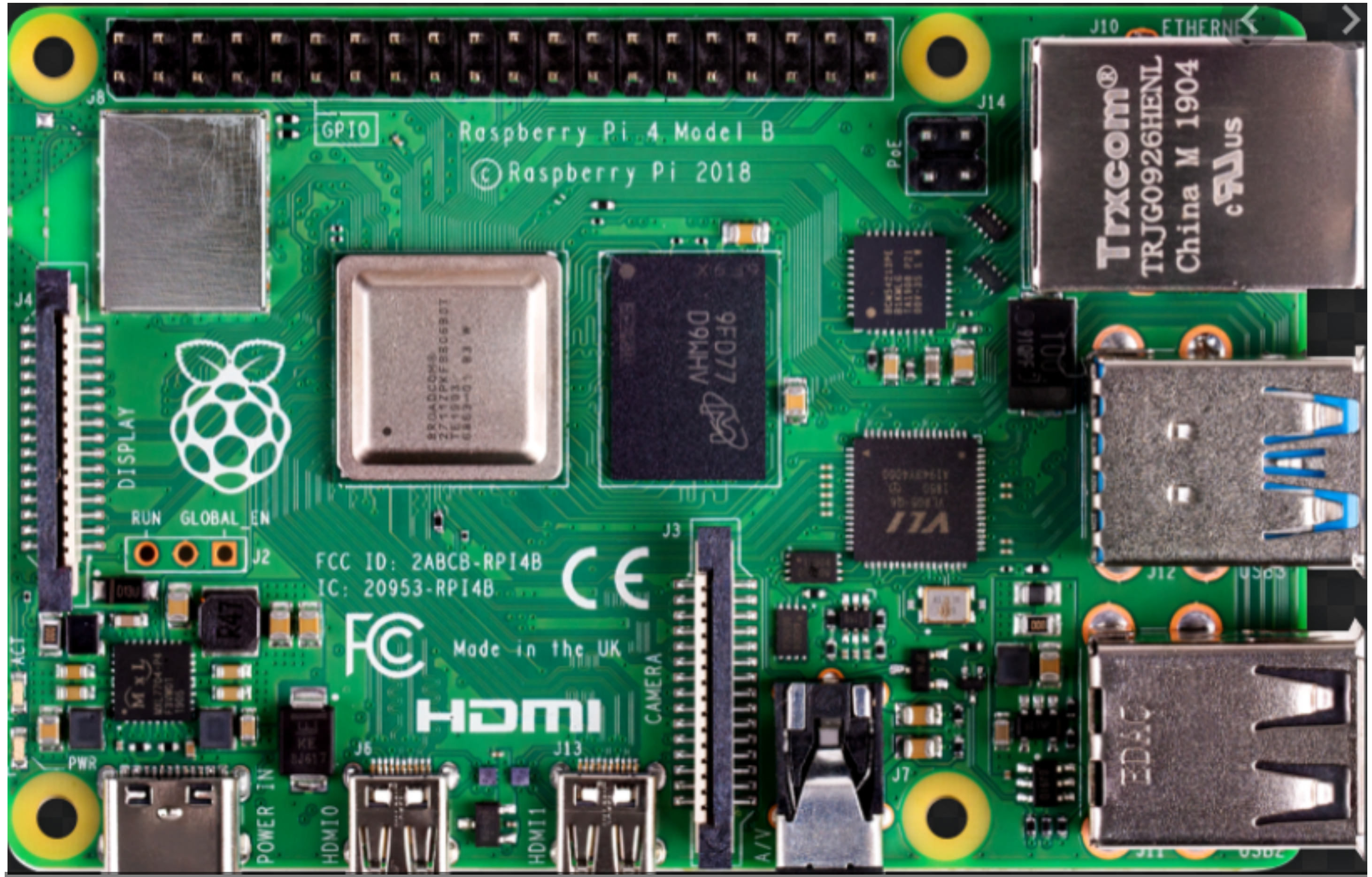
Current Progress - Subsystems

- The distance sensor is able to perform measurements from 1 to 10 feet within an error of +/- 5 cm ✓
- The distance sensor fails to detect the laser on certain surfaces, such as black/dark colors and shiny surfaces ✗
- LED lights have been tested and are known to accomplish the requirements for light display ✓
- Still in enclosure prototyping phase so still trying to figure out physical layout of light system ✗
- GUI display system is in early designing phase

Stretch Goal: Car Light Detector

- Many people sometimes accidentally leave their car on after parking, or otherwise leave their lights on, wasting battery and posing a security risk
- To detect if the car's lights are on, a system of at least two photodiodes will be used. One will be placed at the level of a car's lights, while the other will be placed away from a car's lights
- The signal on each will be compared to tell whether the lights are on
- If the lights are left on for some time after parking, an SMS will be sent to the user alerting them that their lights were left on

Components



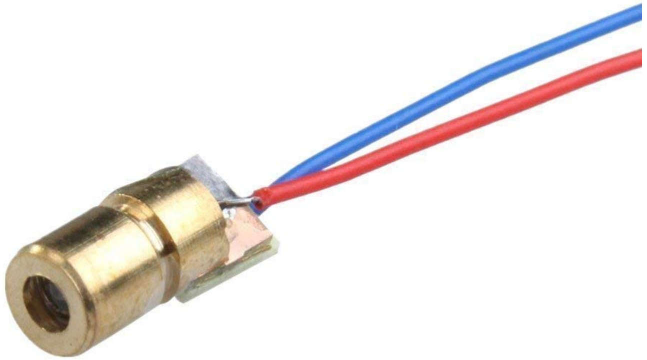
Raspberry Pi 3 Model B

Microcontroller

- 4-core 1.2 GHz CPU
- 1 GB RAM
- USB communication between devices
- 40 GPIO pins
- Native compatibility with external cameras and LCD displays
- Power Requirements: 5VDC, 3A

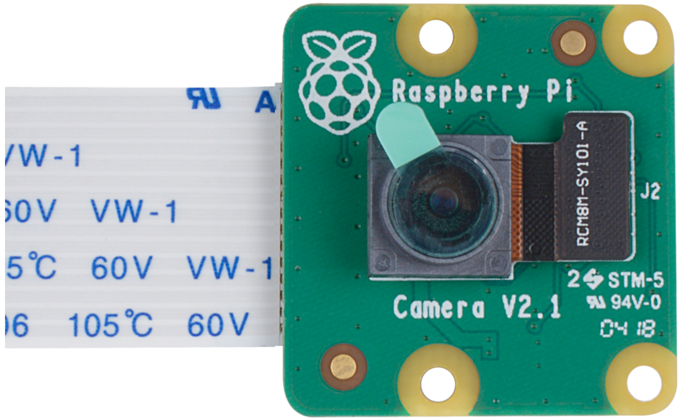
Laser Diode Choice

- **HiLetGo Red (650 nm) Dot Laser Diode**
- Working Voltage: 5 V
- Operating Current: < 20 mA
- Output Power: 5 mW
- Class IIIa laser, safe for skin exposure and momentary/accidental eye exposure
- Chosen because it was cheap and contained a package of 10 laser diodes which could be used for testing



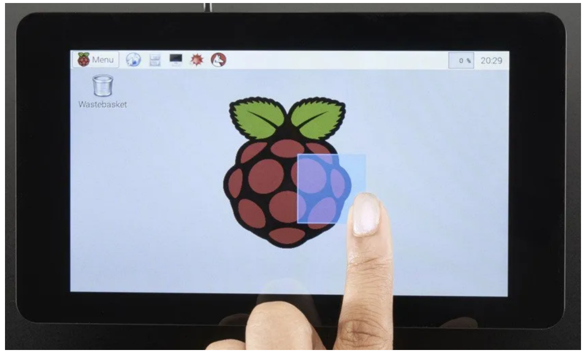
Camera Choice

- Final camera selection was **Raspberry Pi Camera Module V2** due to its easy integration with our Raspberry Pi microcontroller
- Supports many resolutions and video capture as well as individual frame capture
- Python package allows for extracting stream of frames for image processing
- Power fed from Raspberry Pi



Display

Raspberry Pi 7” Touchscreen Display

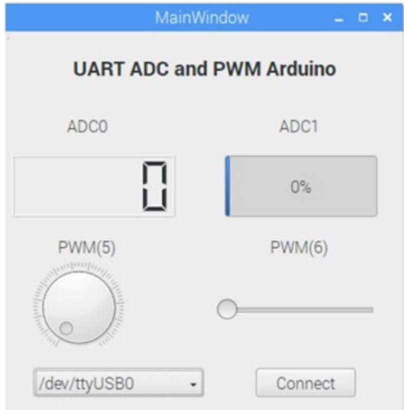


touch screen capability

native resolution of 800x48
pixels

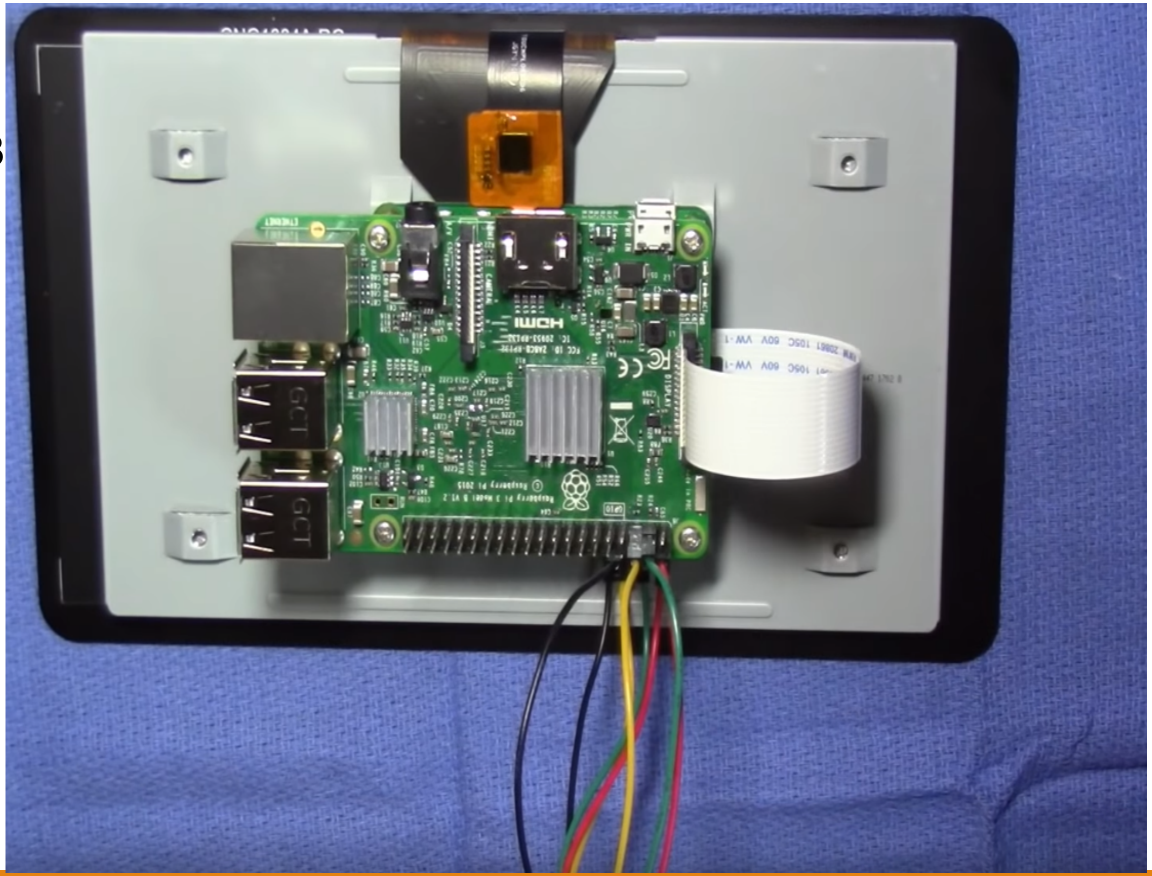
viewable screen size is
1.55mm x 86mm

Example GUI



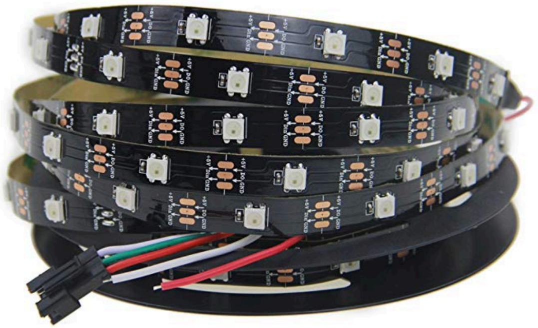
The purpose of the touch display screen is to give the user an interface to control the system via a Graphical User Interface

Mounted Raspberry Pi to Display



RGB LEDs WS2812B

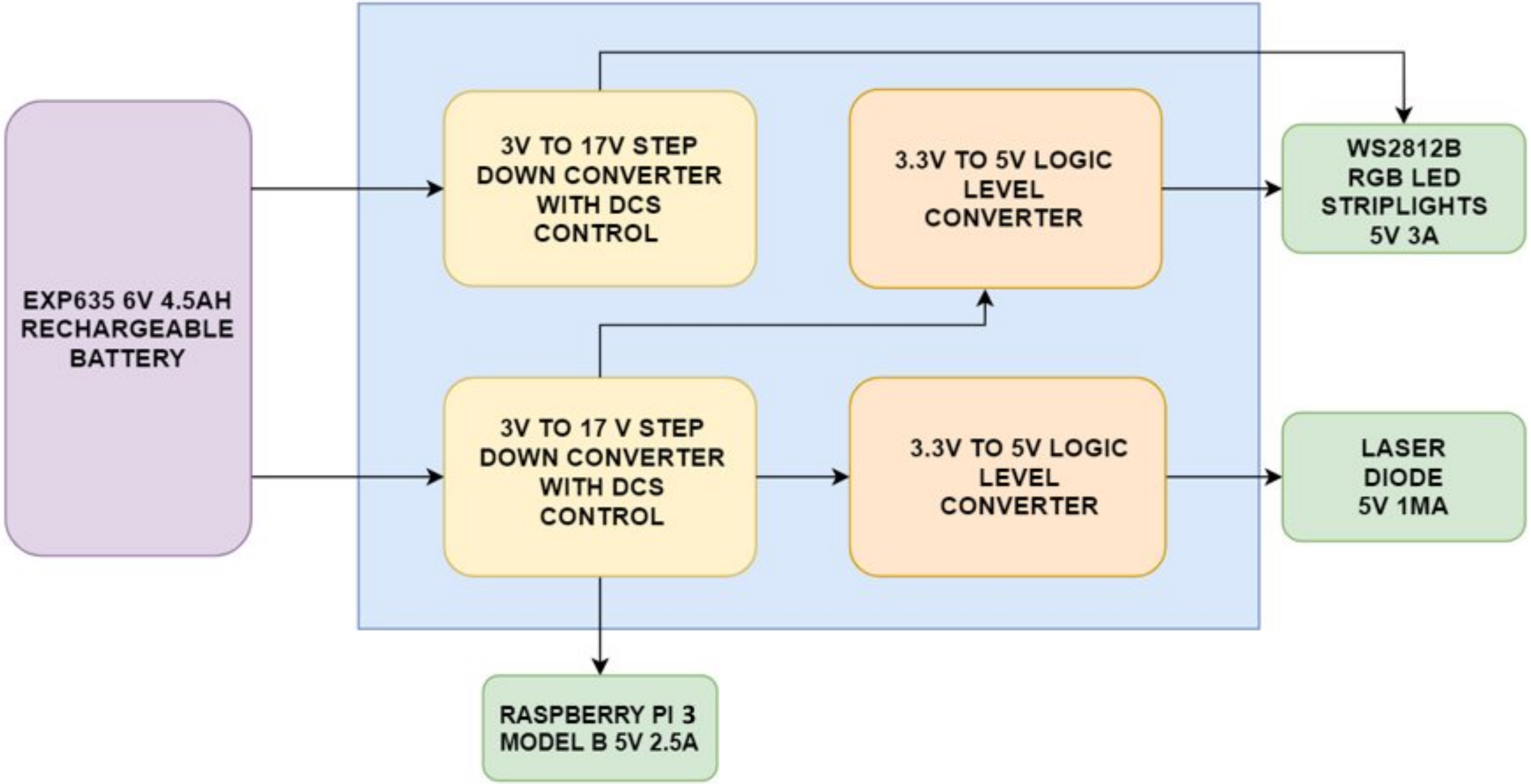
- We are using WS2812B individually addressable LED lights (Neopixel)
- LEDs will display specific colors/turn lights on or off depending on the distance measurements
- Each LED will be able to be individually configured
- Power requirements: 5.5V and 1.67A/m
- Will need a Logic Level Converter
  - Converts the 3.3V to 5.5V



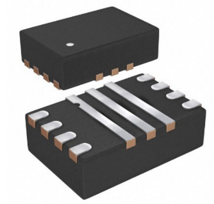
Enclosure

- Enclosure design is still in early stages
- Will contain all of our components, including holder for camera and laser diode, PCB, and Raspberry Pi
- Will be made at CREOL machine shop
- LED light display will be positioned to be viewable to the driver
- Camera and laser will have small openings
- Device should be able to be wall mounted

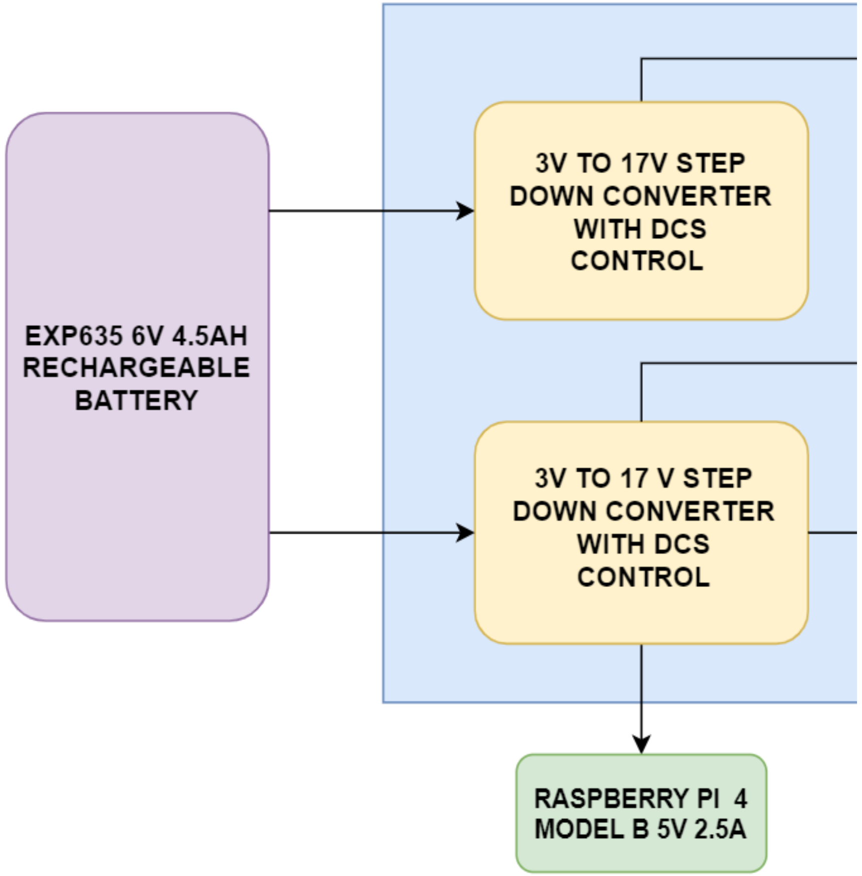
PCB Block Diagram



PCB cont.

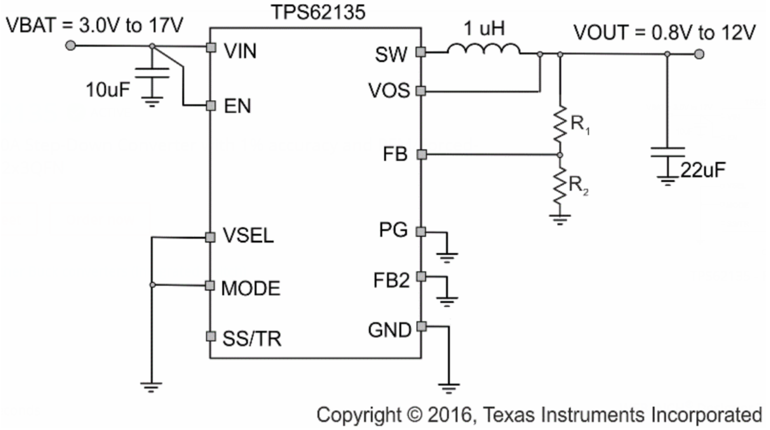


Texas Instruments, 2019



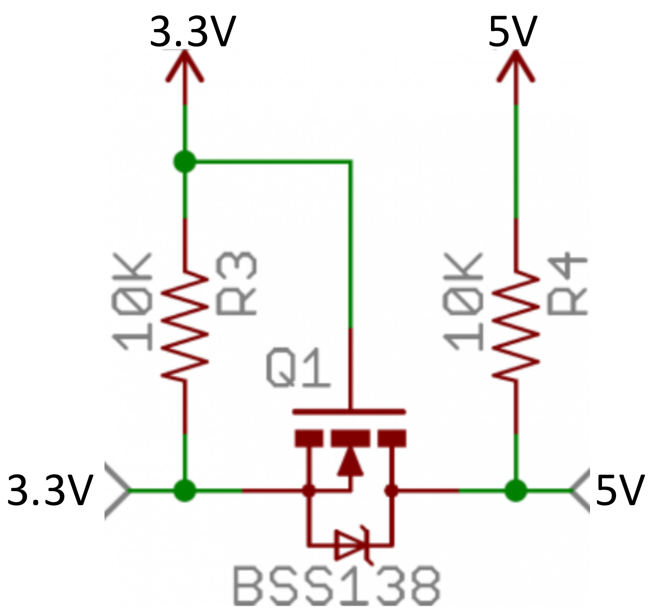
Expert Power, 2019

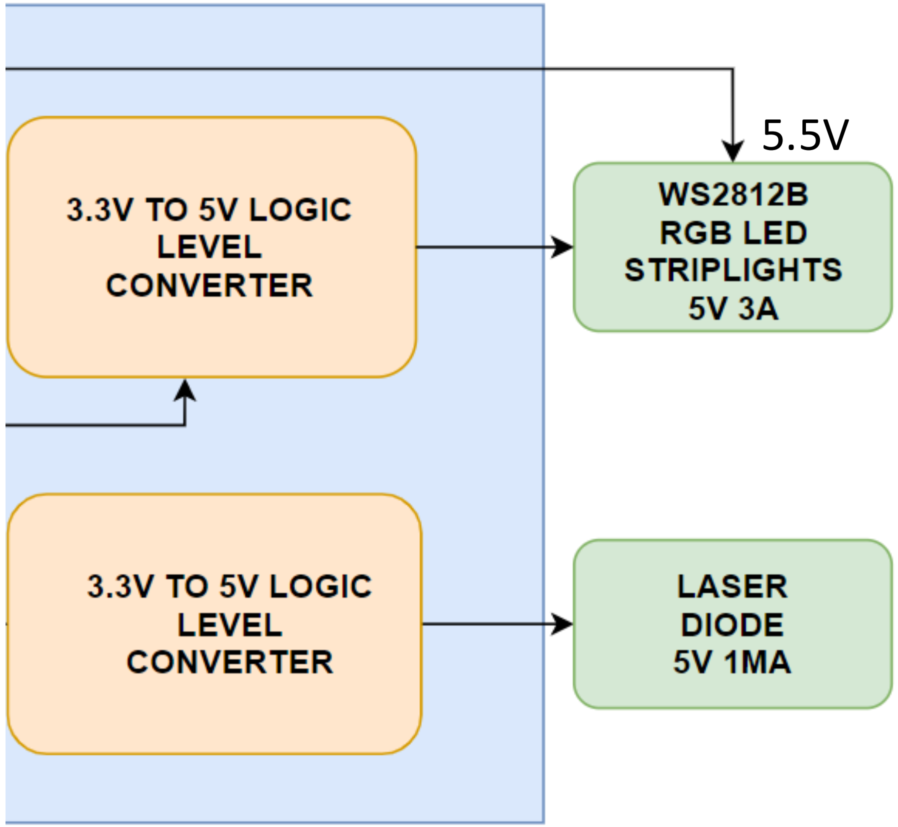
TEXAS INSTRUMENTS TPS621351



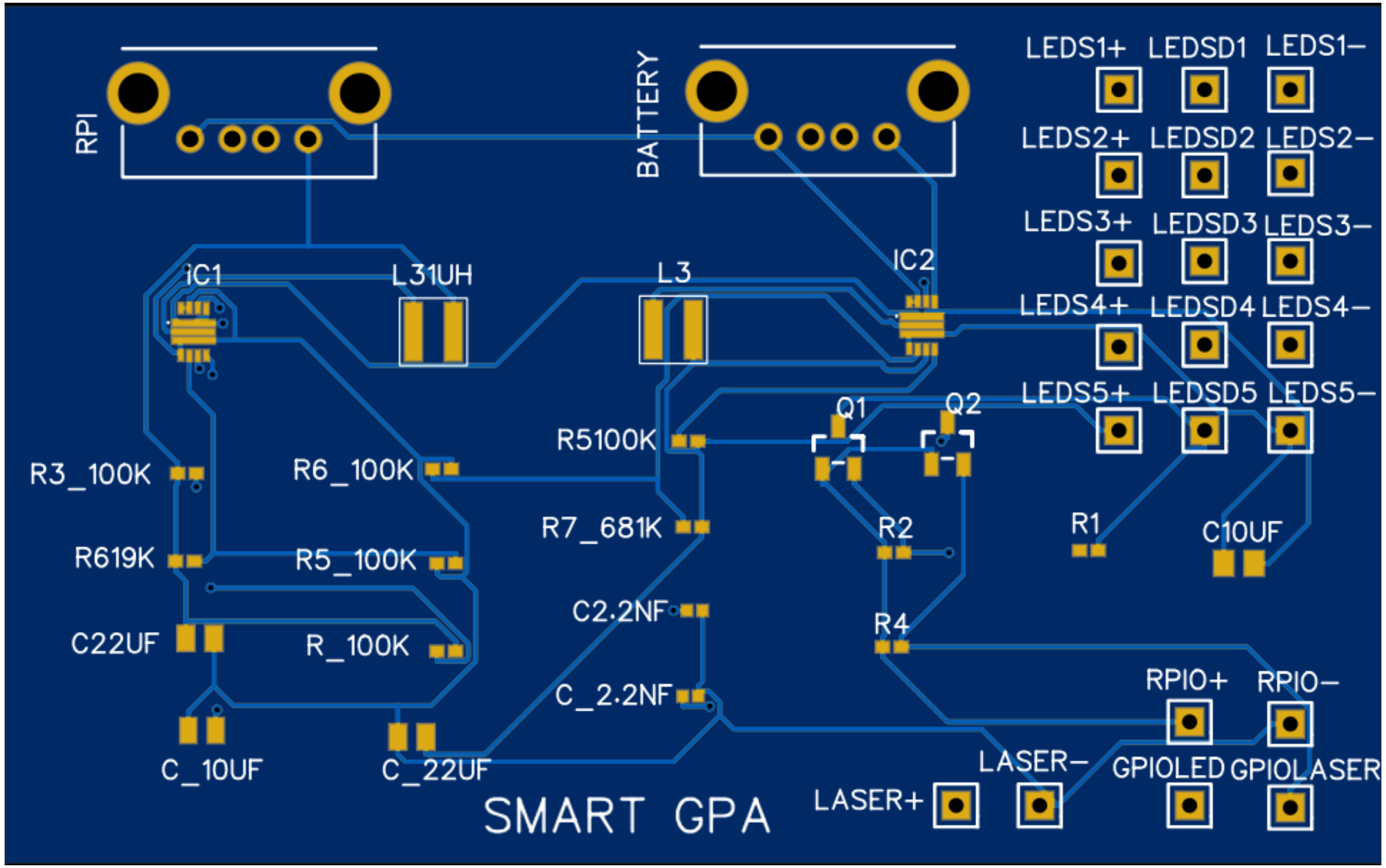
Buck Switching Regulator IC Positive Adjustable

PCB cont.


3.3V
5V
10K
R3
10K
R4
Q1
3.3V
5V
BSS138


5.5V
3.3V TO 5V LOGIC LEVEL CONVERTER
WS2812B RGB LED STRIPLIGHTS 5V 3A
3.3V TO 5V LOGIC LEVEL CONVERTER
LASER DIODE 5V 1MA

PCB Final Design



Administrative Content

Cost Estimation

Item	Quantity	Cost Estimate
Enclosure	1	$50
LED lights	1	$20
PCBs & Components	1	$40
Raspberry Pi	1	$50
Raspberry Pi Camera	1	$25
Red Laser Diodes	10	$10
Battery	1	$20
Touch Screen Display	1	$65
Total:		$280

Budget : $500

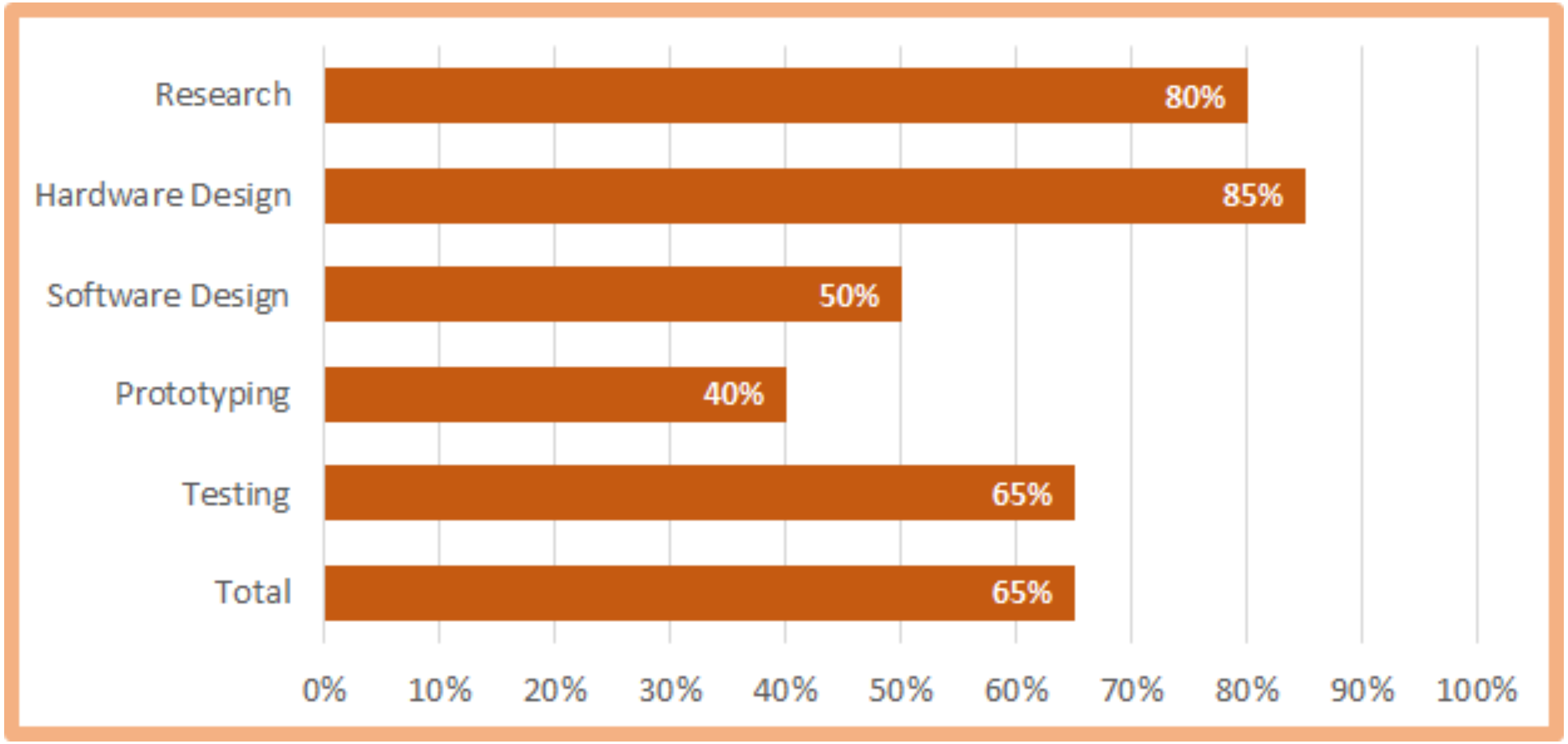
Project Difficulties

- Lack of PCB and soldering skills
- Controlling 5V components with 3.3V logic
  - Testing alternate components of different voltages and voltage converters
- Learning how to use the Raspberry Pi
- Python array manipulation is slow, so measurement rate is hard to improve without using code in a different language
- Laser diode dot is difficult to pick up on black, highly absorptive surfaces

Work Distribution

	Primary	Secondary
PCB/Power	Jennifer Castillo	Nicholas Zollo
Light Display	Daniela Otero	Jennifer Castillo
Laser Distance Sensor	Nicholas Zollo	Jorge Dardon
GUI Display	Jorge Dardon	Daniela Otero

Completion Chart



Questions?