Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4317: Senior Design I
Fall 2023

IGVC
Autonomous Ground Vehicle

Rodrigo Hernandez-Pleitez
Nabeel Ur Rehman Nayyar
Jerome Siljan
Vanshdeep Singh
Jeanne Uwineza
Tsebaot Meron

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

The goal for this is to produce a vehicle that will be able to be autonomously maneuver through a course at the Intelligent Ground Vehicle competition. The vehicle will be built according to our sponsors specifications and the guidelines set by the competitions rules. Elements such as the wheels (mecanum wheels),micro-controllers (cube orange),LiDAR and material for the shell of the vehicle where specifically recommended by our sponsor while the configuration for the vehicle such as the dimensions (length,width, and height), average speed, E-stop protocols and payload are all set by the 2023 Annual Intelligent Ground Vehicle Competition rules.

2 SYSTEM OVERVIEW

In this diagram each layer works together in order to make it possible for the vehicle to collect environmental data and make decisions from internal calculation to avoid obstacles and any-other object it faces during its time on the course. For the hardware layer, it will be responsible for the controlling the software mounted on it in conjunction with making sure the circuits,powering,motors and computing resources are available for any needed jobs. The sensor layer will like the eyes of the vehicle. The sensors will help the vehicle gather the needed information from its environment to send to the controller layer to process, the camera will be used for image detection (such as obstacles and objects), the LiDAR maps where the object is in relation to the vehicle and lastly the GPS will be used for path planning. Lastly the controller layer serves to create a navigation plan for the vehicle by using the implemented AI component it is given.

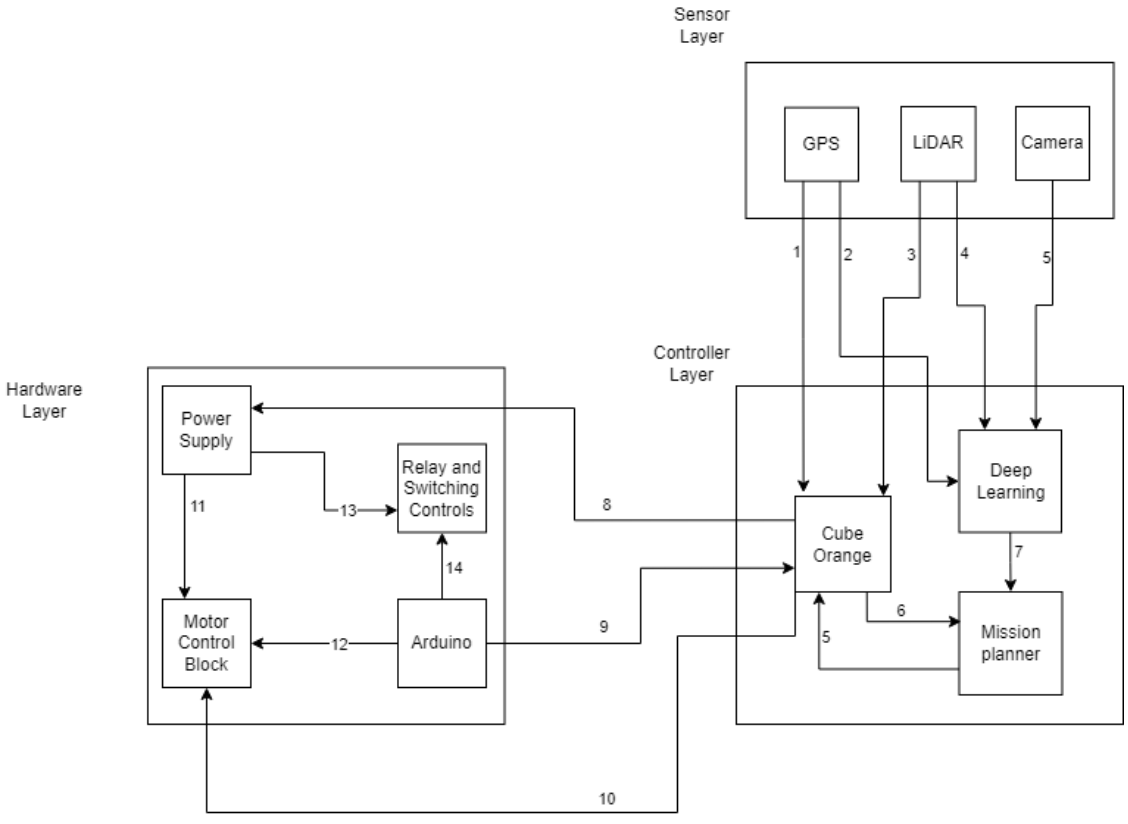

Figure 1: System architecture

3 Sensor Layer Subsystems

The sensor layer of the data flow diagram is mostly comprised of the three hardware components that are used for data collection and vehicle environmental detection. This data collected by these systems will be sent to the cube orange for use in calculations. The programming language that we will be using to communicate to the cube orange is going to be python. The software dependence for this layer will be the cube orange ARDU pilot 4.0 for software connection.The operating system that will be used for this project will be linux (ubuntu systems).

3.1 Layer Hardware

For this layer the GPS subsystem will come built into the cube orange while the camera and the LiDar will be separately connected and calibrated to deliver data to the cube orange and the ROS system we will be using.

3.2 Layer Operating System

For the software of this project we will be utilizing the Linux operating system

3.3 Layer Software Dependencies

We will be using open CV for the camera and python libraries like LiDAR for the LiDAR that we will be using. Since the GPS is in the cube orange it depend on the ARDU pilot system that is run on the cube orange.

3.4 Subsystem:LiDAR

The LiDAR is a hardware sensor that is capable of gathering depth information from its surroundings that will allow us to see what the vehicle sees in a form of a point cloud. The sensor itself will be configured to face forwards on the vehicle since in the use case, the robot will be always moving forward.

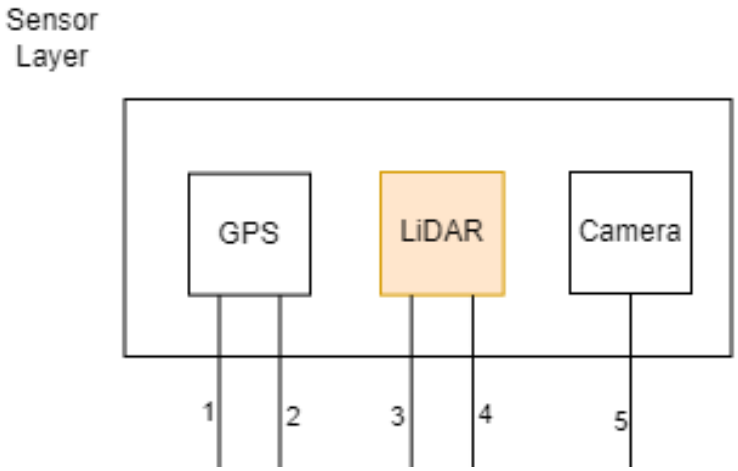

Figure 2: Lidar located within the sensor layer

3.4.1 Subsystem Hardware

The system will use a SLAMTEC RPLIDAR S1 Sensor which is capable of 360 degree sensing and can take 9200 samples per second with a 30 meter range for white objects and as low as a 10 meter range for black objects. In order to connect to the system it uses a proprietary communication and power interface, the cable itself is on one end uses a SH1.0-6P male socket and the other end XH2.54-5P male socket.

3.4.2 Subsystem Operating System

The sensor itself not have a operating system as it only purpose is to send the 2D point cloud data to the controller layer which will interpret the information that it has received.

3.4.3 Subsystem Software Dependencies

The library that will be used is the python library known as LiDAR. This will allow us to more conveniently use the data with the vision system on the companion laptop. However for testing purposes the LiDAR can be used with RoboStudio for testing a debugging with the Framegrabber plugin on a windows or linux system.

3.4.4 Subsystem Programming Languages

Python will be used as it will allow for rapid development as well as make the code more readable and maintainable.

3.4.5 Subsystem Data Structures

The data coming out of the LiDAR sensor will be in the form of point cloud with the data types of Distance, Heading, Start Flag, and Checksum being used. The Distance will be in millimeters with the distance being calculated from the rotating core of the LiDAR and the sampling point, the Heading will be in degrees, which is the current heading angle of the measurement, the Start Flag will let us know when a new measurement is coming in and a old one has ended, and finally the checksum which will let us confirm the data that we have received has not been corrupted. Also to note is that this data is a continuous flow which is why the start flag is needed in case the sensor gets disconnected or the companion laptop is not able to process the information that fast it can just continue on without needing the previous data. This also allows us to decide on how detailed we want our sensor data.

3.4.6 Subsystem Data Processing

The data will be continually process by the LiDAR library in python that will be continuously parsing new information sent by the sensor.

3.5 Subsystem:Camera

This subsystem is put into place for object recognition that will later be implements in our deep-learing and AI. This will help the vehicle collect the data needed from the course and feed it to the computer to be used for later computation.

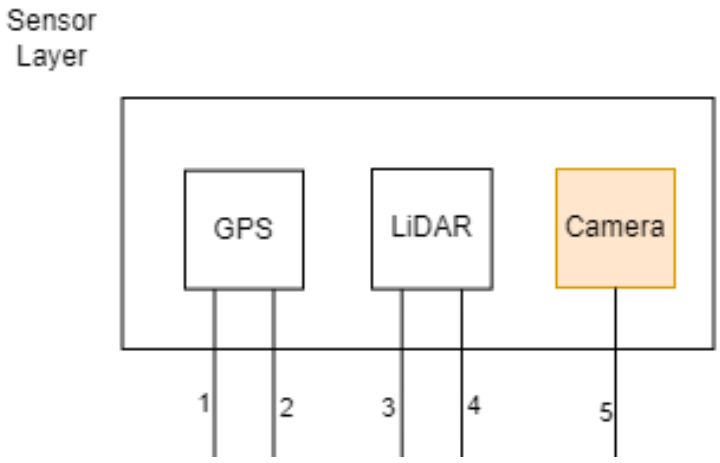

Figure 3: Example subsystem description diagram

3.5.1 Subsystem Hardware

A description of any involved hardware components for the subsystem. No hardware components will be used for this subsystem not including the camera its self.

3.5.2 Subsystem Operating System

Linux (Ubuntu) will be used as the operating system.

3.5.3 Subsystem Software Dependencies

Open CV needed for the camera to communicate to the cube orange using python.

3.5.4 Subsystem Programming Languages

Python will be the main programming language for maintainability and readability of the code.

3.5.5 Subsystem Data Structures

The data being transmitted from the camera to a computer via USB is going to be the image of 720 X 1280 x 3.

3.5.6 Subsystem Data Processing

N/A

3.6 Subsystem:GPS

This subsystem will be used for the vehicle to locate its self in space. This system is integrated in the cube will keep the vehicle updated on its position throughout its journey through the competition course.

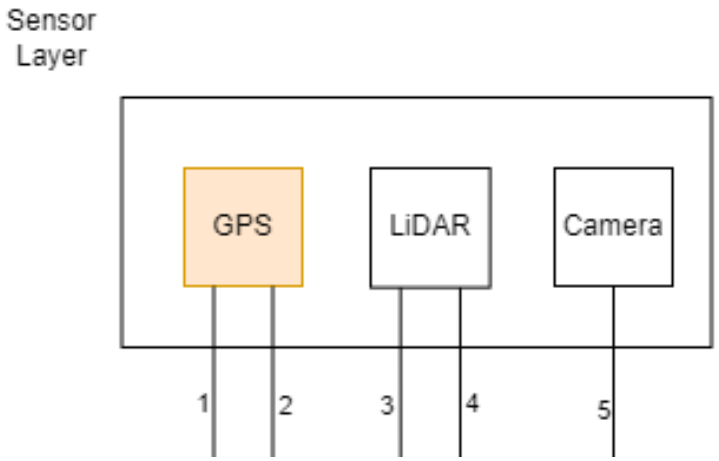

Figure 4: Example subsystem description diagram

3.6.1 Subsystem Hardware

The GPS is embedded into the cube orange so it does not require anyother hardware.

3.6.2 Subsystem Operating System

The operating system that is utilized by this subsystem is what the cube orange will be interacting with the Linux system.

3.6.3 Subsystem Software Dependencies

The software dependencies is the cube orange software ARDU pilot 4.0.

3.6.4 Subsystem Programming Languages

Python will be used for the coding of this subsystem

3.6.5 Subsystem Data Structures

The data coming in from this subsystem will be in the form of two components: latitude (giving the north-south position) and a longitude (giving the east-west position). This data will be used by the cube orange for further processing.

3.6.6 Subsystem Data Processing

N/A

4 Controller Layer Subsystems

The controller layer will be the backbone that will, once fully implemented, will let the vehicle be fully autonomous. The cube orange will control the motion of the vehicle while the mission planner and the deep learning portions of this layer make it possible for the vehicle to map and calculate the best way to get to its destination. The deep learning will feed into the mission planner data for mapping with the mission planner will exchange data to and from the cube orange to find its best way through the course.

4.1 Layer Hardware

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

4.2 Layer Operating System

This layer will be running on the linux operating system.

4.3 Layer Software Dependencies

Mission Planner, LiDAR libray from python, and ROS.

4.4 Subsystem Deep Learning

The deep learning subsystem will be the software implemented to make it possible for this vehicle to learn and coordinate the next plan of action from the information it receives from its sensors. The plan is to implement a way for the vehicle to perceive the world around it, to calculate its position and orientation, to predicate its next move/ reaction and finally make its final decision on all of its calculations.

4.4.1 Subsystem Hardware

There will be no hardware involved in this subsystem since it is mostly the software computation for the data it receives.

4.4.2 Subsystem Operating System

This portion will be on a Linux operating system

4.4.3 Subsystem Software Dependencies

This system will be dependent on what data it receives from the sensor layer.The frameworks that we will be using for this subsection will be thing like PyTorch and Tenser-Flow which are good for numerical and image data sets.

4.4.4 Subsystem Programming Languages

The deep learning algorithm will be written in python

4.4.5 Subsystem Data Structures

A description of any classes or other data structures that are worth discussing for the subsystem. For example, data being transmitted from a microcontroller to a PC via USB should be first be assembled into packets. What is the structure of the packets?

4.4.6 Subsystem Data Processing

This subsection will first be tested with A* search. Since this subsection will be collecting data from the sensory layer, the data can be used to model a deep learning model using deep learning algorithms to make decisions relevant to the environment. Using Convolutional neural networks, we will be able to

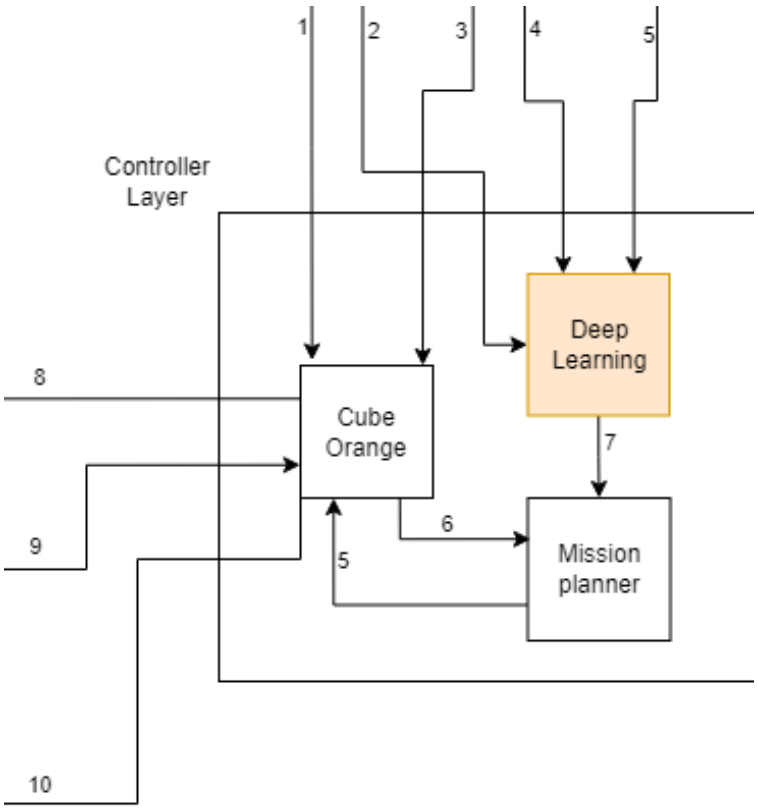

Figure 5: Example subsystem description diagram

extract features from images and process spatial information and for decision making, the vehicle will utilize path planning .

4.5 Subsystem Cube Orange

The Cube Orange at a high level is responsbile for managing the hardware variables of the vehicle. Such as the motor, battery, LiDAR, gps, speed, and such. It will be responsbile for our case to collect all that data and send it over to the mission planner subsystem.

4.5.1 Subsystem Hardware

The Cube Orange is a flight controller that will be used to get data such as gps but to also control and manage the hardware layer as it will be responsible for motor controls, battery, and telemetry.

4.5.2 Subsystem Operating System

To use the cubeorange we must use the software missionplanner.

4.5.3 Subsystem Software Dependencies

The Cube orange operates using mission planner which will be installed on companion laptop.

4.5.4 Subsystem Programming Languages

Mission planner uses python scripts to work.

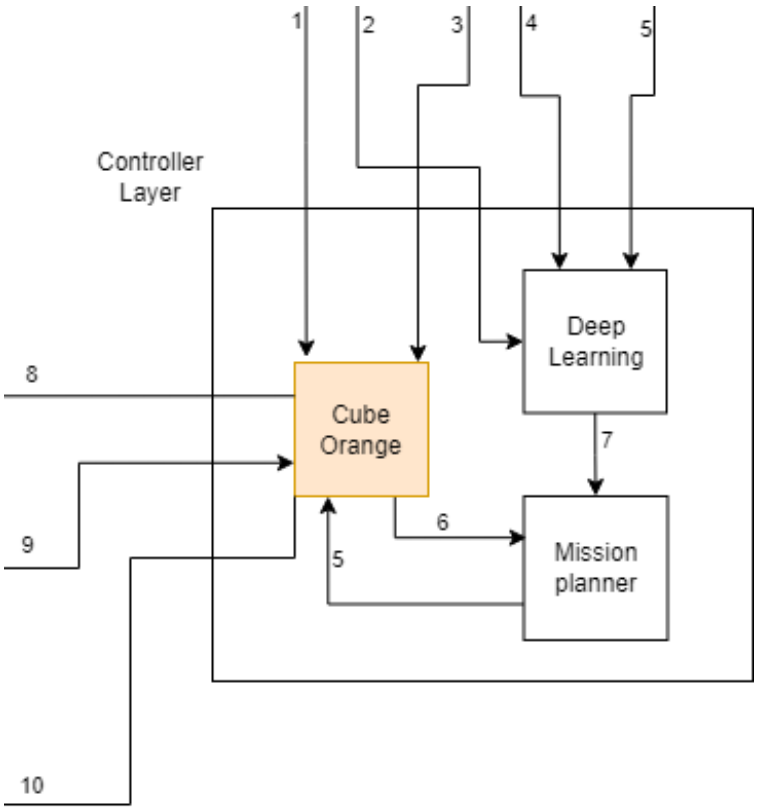

Figure 6: Cube Orange in the controller layer diagram

4.5.5 Subsystem Data Structures

A micro usb cable will connect from the mission planner to the Cube Orange. Other devices that will connect to it will be the sabre tooth motor driver to the main out connectors, the gps to the GPS1 port, the LiDAR to the TELEM1 port, and as well as the battery to the POWER1 port

4.5.6 Subsystem Data Processing

There are no algorithms that will be used since that is handled by the mission planner subsystem.

4.6 Subsystem Mission Planner

Mission planner at a high level sends commands to Cube Orange and recieves data from the Cube Orange such as gps,LiDAR,Battery, and speed. It use for the vehicle is that it will be used by the Deep Learning system to send commands on where it needs to go and mission planner will handle lower level controls.

4.6.1 Subsystem Hardware

Mission Planner will be ran on the same device as the deep learning will be done which is the companion laptop that will be attached to the vehicle.

4.6.2 Subsystem Operating System

The system will reqiure a linux operating system to run on

4.6.3 Subsystem Software Dependencies

The software has no dependancies besides the drivers for the Cube Orange which can be found on the website. Since that is required inorder for mission planner to detect the cube orange.

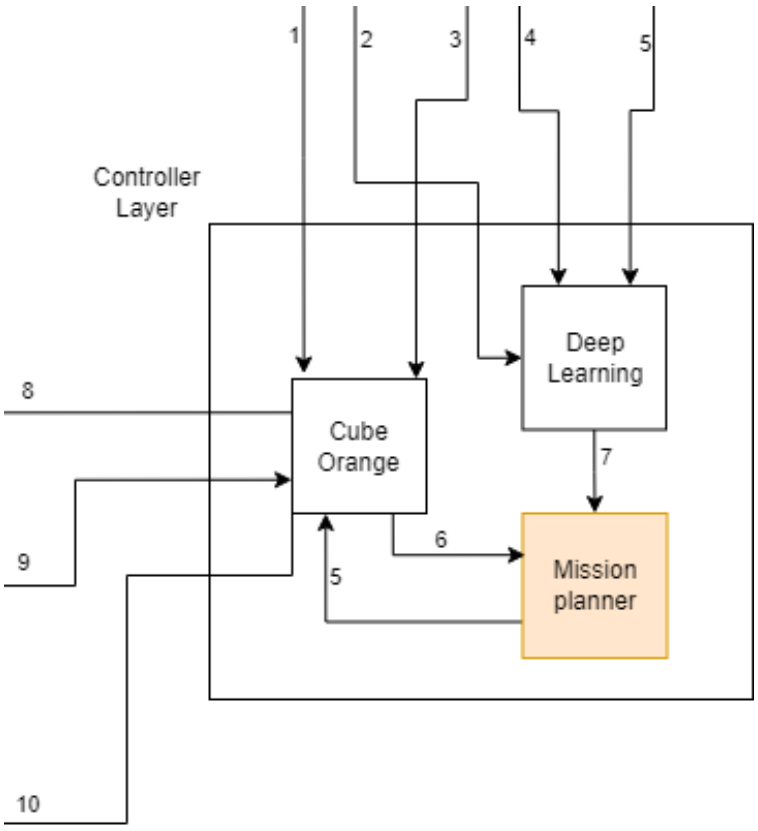

Figure 7: Mission Planner in the controller layer diagram

4.6.4 Subsystem Programming Languages

Mission Planner will use python scripts to run.

4.6.5 Subsystem Data Structures

There is no data structure as it will use the application to run.

4.6.6 Subsystem Data Processing

The data will be posted onto mission planner by the deep learning subsystem as a point to go to in which mission planner will execute. There is no path algorithm as that is handled by the deep learning subsystem.

5 HARDWARE LAYER SUBSYSTEMS

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project.

5.1 LAYER HARDWARE

A description of any involved hardware components for the layer. For example, if each subsystem is a software process running on an embedded computer, discuss the specifics of that device here. Do not list a hardware component that only exists at the subsystem level (include it in the following sections).

5.2 LAYER OPERATING SYSTEM

A description of any operating systems required by the layer.

5.3 LAYER SOFTWARE DEPENDENCIES

A description of any software dependencies (libraries, frameworks, etc) required by the layer.

5.4 SUBSYSTEM 1:ARDUINO

Subsystem 1 serves as the core control unit of the vehicle. It is a hardware component, the Arduino microcontroller, which acts as the vehicle's central processing unit. This subsystem facilitates communication with sensors and motors, enabling data collection and motor control essential for autonomous operation.

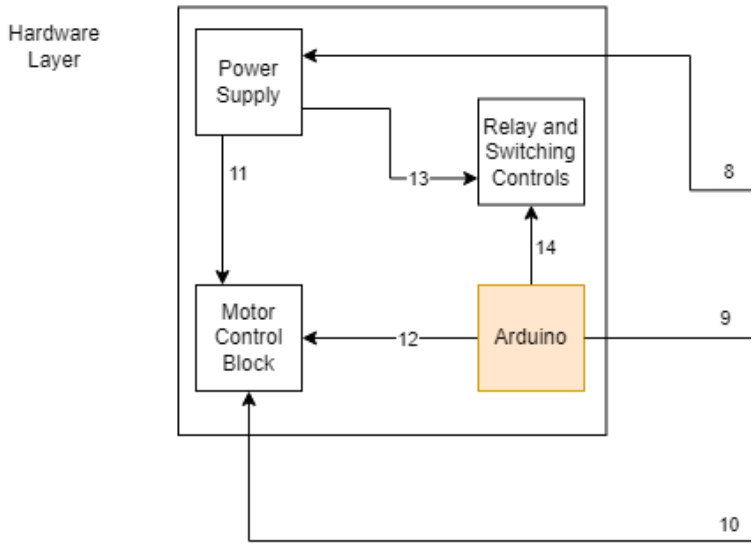

Figure 8: Example subsystem description diagram

5.4.1 SUBSYSTEM HARDWARE

The Arduino microcontroller is the primary hardware component in Subsystem 1. It's like the central processing unit (CPU) of a computer but designed for embedded systems. It has input and output pins that connect to sensors, motors, and other devices. These connections allow it to receive data from

sensors like cameras and LiDAR and send commands to motors for driving the vehicle. The Arduino microcontroller is like the vehicle's control center.

5.4.2 Subsystem Operating System

The Arduino microcontroller doesn't use a traditional operating system like a laptop or smartphone. Instead, it runs a simple program that we write to control its behavior. This program is uploaded to the microcontroller, and it follows the instructions we give it.

5.4.3 Subsystem Software Dependencies

To program the Arduino microcontroller, we use a software development environment called the Arduino IDE (Integrated Development Environment). This IDE provides a user-friendly interface for writing, compiling, and uploading code to the microcontroller.

5.4.4 Subsystem Programming Languages

We use a programming language "Arduino C/C++" to write code for the microcontroller. It's a simplified version of C/C++ designed for easy use with Arduino hardware.

5.4.5 Subsystem Data Structures

We often use data structures like arrays and variables to organize and store data. For example, if we're collecting sensor readings, we might use an array to store these values neatly. Data from the microcontroller, such as sensor data or control commands, is usually organized into packets, which are like envelopes holding information. These packets have structures that help us decode and use the data effectively.

5.4.6 Subsystem Data Processing

Data processing in the Arduino microcontroller involves making decisions based on sensor data. For instance, if the vehicle's camera detects an obstacle, the microcontroller might process this information and decide to stop or change direction. While the micro controller doesn't use complex algorithms like a supercomputer, it uses simple logic and calculations to control the vehicle's actions. This processing is fast and efficient, making real-time decisions for safe navigation.

5.5 Subsystem 2:Motor Control Block

Subsystem 2, it consists of motor drivers. Motor drivers regulate motor power for precise control, encoders provide feedback on motor positions and speeds, and the motors physically drive the wheels, enabling the vehicle's movement.

5.5.1 Subsystem Hardware

The key hardware component in Subsystem 2 is the motor control unit. This unit comprises motor drivers, encoders, and the motors themselves. The motor drivers interpret commands from the Arduino microcontroller and regulate the power supplied to the motors, allowing precise control of the vehicle's movement. Encoders provide feedback on the motor's position and speed, aiding in accurate navigation.

5.5.2 Subsystem Operating System

Motors does not require a dedicated operating system. Instead, it interfaces directly with the Arduino microcontroller, which manages its operations.

5.5.3 Subsystem Software Dependencies

Motors relies on software libraries and control algorithms provided by the Arduino platform. These libraries enable the microcontroller to send motor commands and receive feedback from encoders.

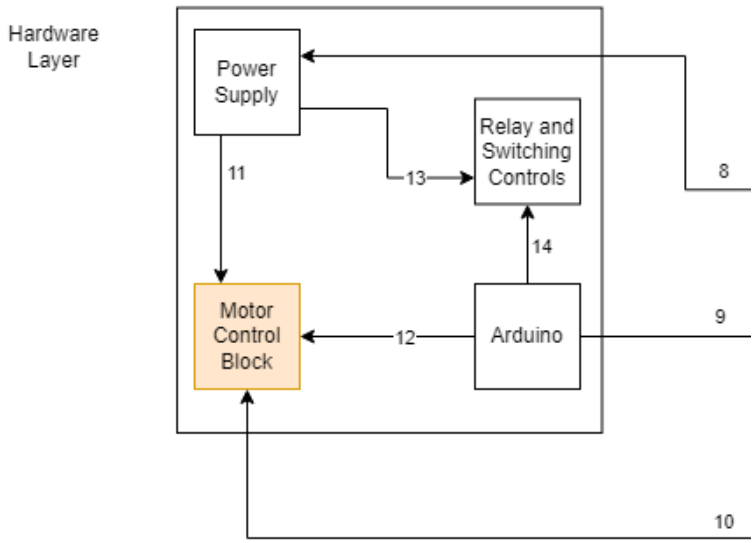

Figure 9: Example subsystem description diagram

5.5.4 Subsystem Programming Languages

Motor Don't have a specific programming language it takes commands from the microcontroller.

5.5.5 Subsystem Data Structures

Data packets are used to transmit information between the microcontroller and motor control unit, containing details about desired motor speeds, directions, and current positions.

5.5.6 Subsystem Data Processing

Processing commands from the Arduino microcontroller to adjust motor speeds and directions. This process ensures that the vehicle moves according to the navigation plan generated by the higher-level control systems. Additionally, feedback from encoders is processed to provide accurate information about the vehicle's movements, aiding in navigation and obstacle avoidance.

5.6 Section 3:Power Supply

The power supply maintains a constant power supply for the peripherals, such as the motors, motor controllers, and microprocessors.

5.6.1 Subsystem Hardware

The power supply interfaces with the relay controls that shut off power to the motor control block if the motors draw too much current. The arduino is an input that sets certain pins high based on the desired configuration of the relay and switching controls.

5.6.2 Subsystem Operating System

The power supply doesn't run any code.

5.6.3 Subsystem Software Dependencies

The power supply doesn't run any code.

5.6.4 Subsystem Programming Languages

The power supply doesn't run any code.

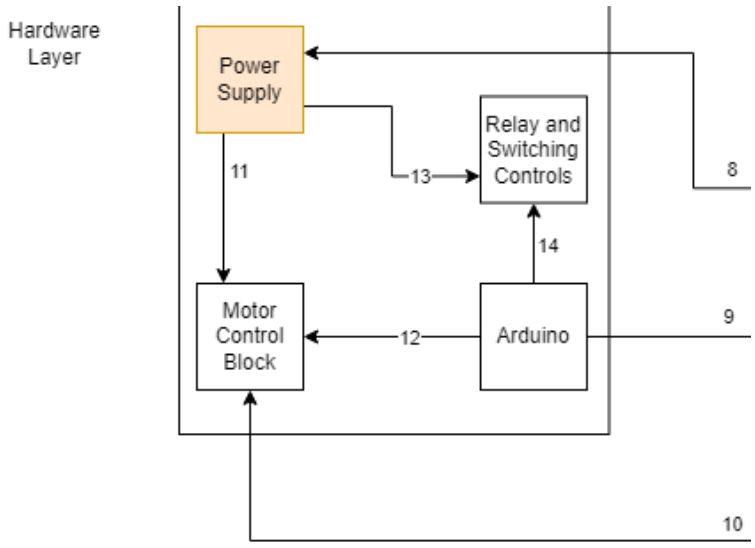

Figure 10: Example subsystem description diagram

5.6.5 Subsystem Data Structures

The power supply doesn't run any code.

5.6.6 Subsystem Data Processing

The power supply doesn't run any code.

5.7 Subsystem 4:RSC

The relay and switching controls cut power off from the motors depending on current draw, power supply health, and configuration provided by the arduino.

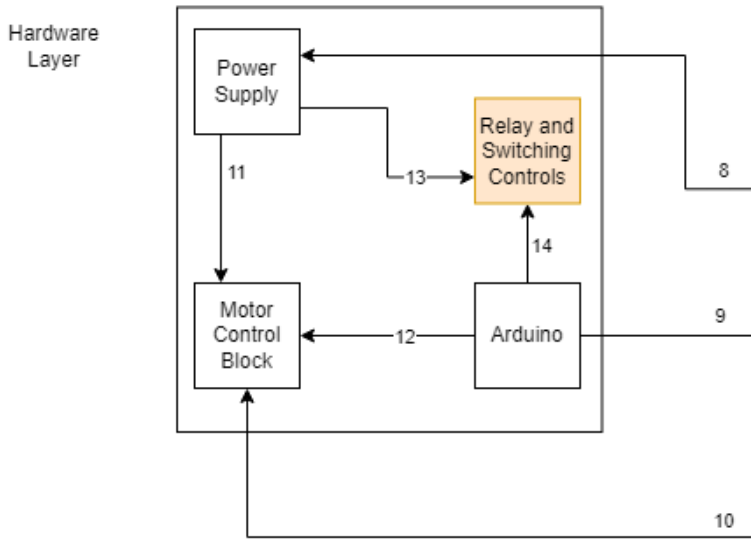

Figure 11: Example subsystem description diagram

5.7.1 Subsystem Hardware

The relay and switching controls interface shut off power to the motor control block depending on current draw. Different configurations can be set by an input that sets certain pins on the RSC high/low. In this case, the configuration is set by an arduino.

5.7.2 Subsystem Operating System

The relay and switching controls doesn't run any code.

5.7.3 Subsystem Software Dependencies

The relay and switching controls doesn't run any code.

5.7.4 Subsystem Programming Languages

The relay and switching controls doesn't run any code.

5.7.5 Subsystem Data Structures

The relay and switching controls doesn't run any code.

5.7.6 Subsystem Data Processing

The relay and switching controls doesn't run any code.

6 APPENDIX A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.

https://ardupilot.org/copter/docs/common-thecubeorange-overview.html

https://cdn.robotshop.com/media/r/rpk/rb-rpk-13/pdf/ld601$_s lamtec_r plidar_d atasheet_s 1_v 1.7_e n.pdf$

REFERENCES