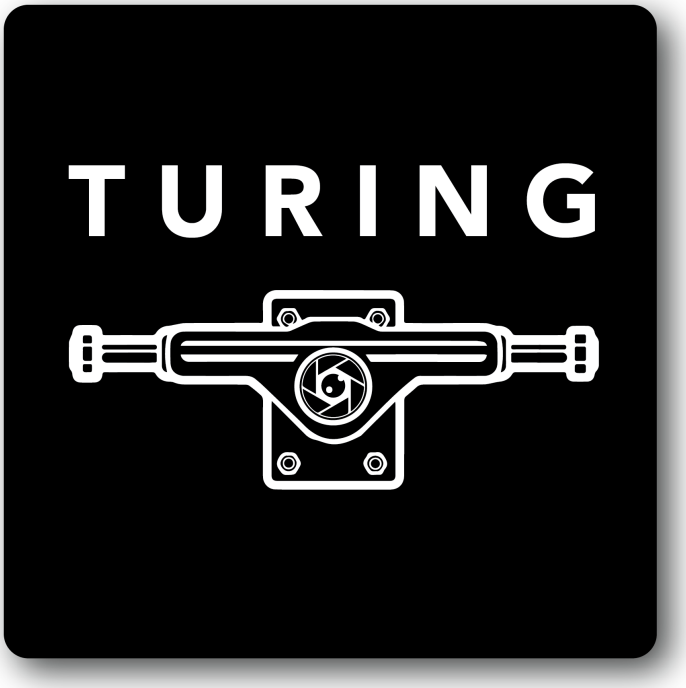
Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4317: Senior Design II
Spring 2022



Runtime Terror
Turing Board

Sahaj Amatya
Sarker Nadir Afridi Azmi
Kendall Buchanan
Keaton Koehler
Happy Ndikumnaa
Lydia Sarver

Revision History

Revision	Date	Author(s)	Description
0.1	1.01.2016	GH	document creation
0.2	1.05.2016	AT, GH	complete draft
0.3	1.12.2016	AT, GH	release candidate 1
1.0	1.20.2016	AT, GH, CB	official release
1.1	1.31.2016	AL	added design review requests

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

The Turing Board is a concept autonomous longboard which is capable of exhibiting self-driving capabilities using computer vision. Users of the Turing Board will be able to take advantage of various features such as having the board follow you autonomously and having the board summon itself to you from a parked location, in addition to functioning as a standard electric longboard capable of recording and analyzing all trip data. Users will also be able to use the Turing Board to function as a load carrier, relieving them from the burden of carrying everyday items like backpacks, boxes, etc. if desired. The main components of the Turing Board include a remote control via an app on the user's phone, the Jetson TX2 controlling the software signals and output to other components of the board, the motorized wheels and turning mechanism, computer vision, human-machine interface, and the board's power supply.

2 SYSTEM OVERVIEW

The main controls layer is in charge of process and sending out the majority of the signals on the Turing Board. It uses the Jetson TX2 to communicate with the microprocessor in the wheels and turning layer to control the wheels and turning mechanism. There are several sensors in the wheels and turning layer that send information via the microprocessor to the Jetson as inputs. The Jetson also receives inputs from the power layer to know how much power is left in the battery. The last input to the Jetson comes from the Human Machine Interface (HMI) layer. This layer consists of items used to allow the Turing Board to understand the world around it and operate more appropriately. Along with the human machine interface layer, the computer vision layer also contributes to the Jetson TX2 receiving feedback from its environment. The computer vision layer uses RGB imagery and stereo and infrared imagery to see obstacles and the user depending on the mode it is. Finally, the remote control for the Turing Board allows the user to control the board as an electric longboard when in the respective mode. The app boasts many features including authentication and ride data analysis.

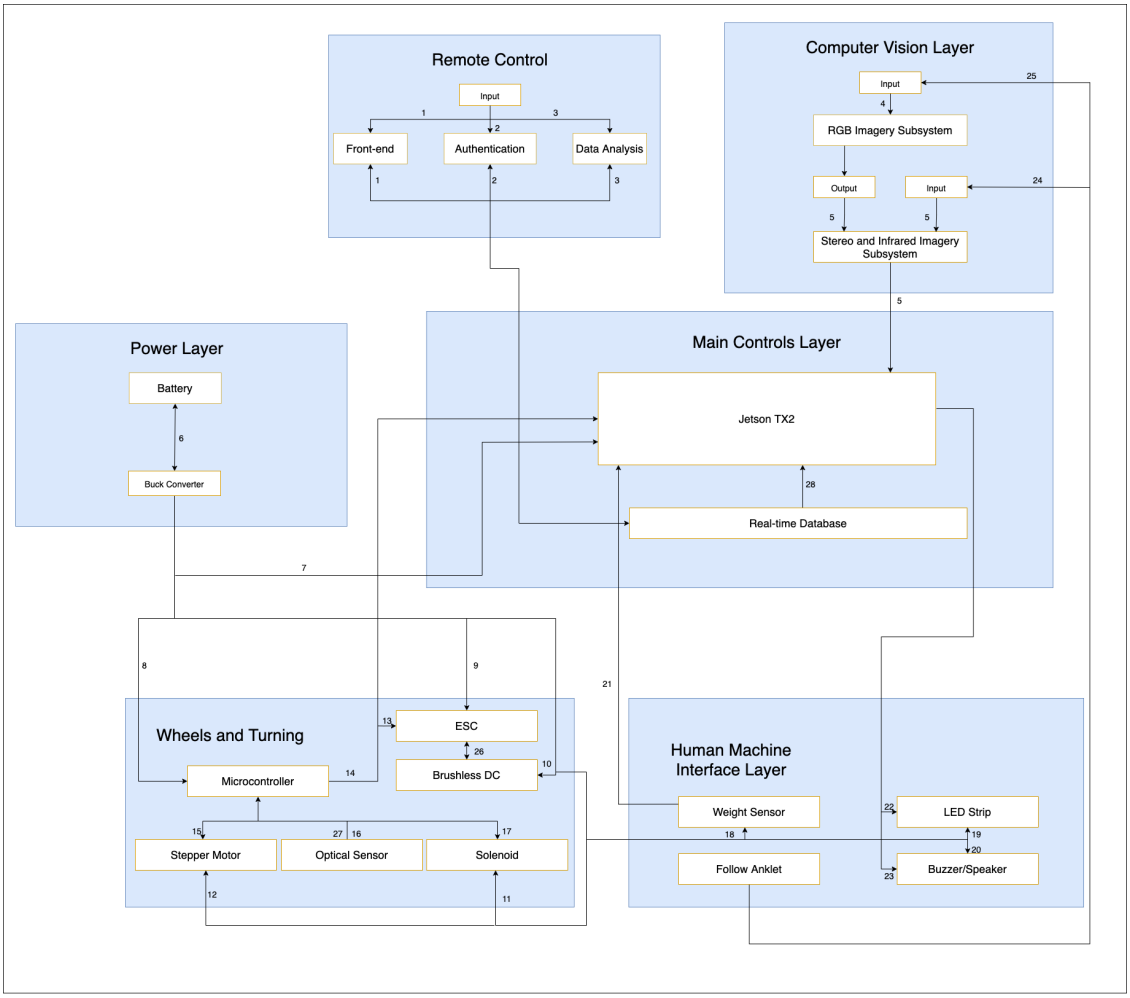


Figure 1: System Architecture

3 Power Layer Subsystems

Each electrical component has a specific power requirement and this layer is responsible for providing that power.

3.1 Layer Hardware

The Turning Board includes an ER 36 volt battery and Songhe LM2596S buck converters.

3.2 Layer Operating System

No operating system is used in this layer.

3.3 Layer Software Dependencies

No software is used in this layer.

3.4 Subsystem 1

This module consists of a Li-Po battery which provides power to a Buck Converter. It is an ER 36 volt battery with a 30A current output.

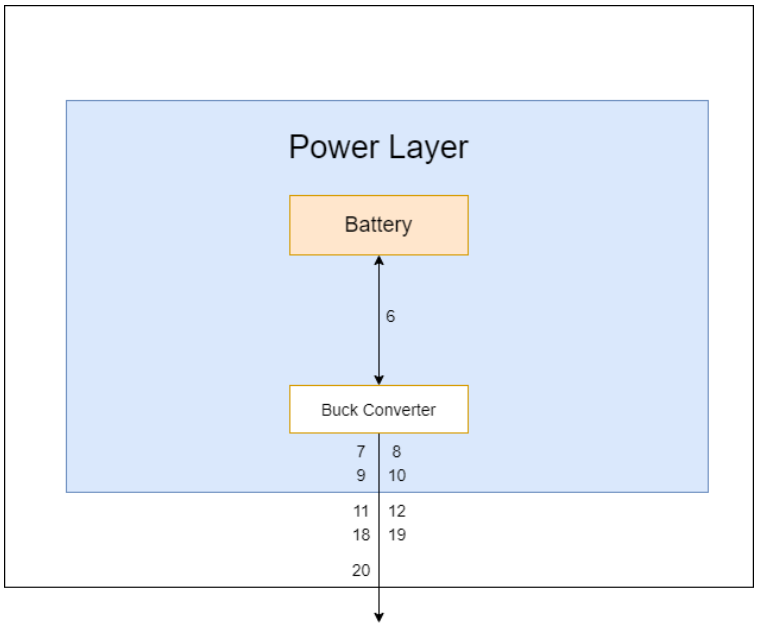


Figure 2: Battery Subsystem in Power Layer

3.4.1 Subsystem Hardware

The battery should will output around 36 volts.

3.4.2 Subsystem Operating System

No operating system is used in this subsystem.

3.4.3 Subsystem Software Dependencies

No software is used in this subsystem

3.4.4 Subsystem Programming Languages

No programming languages are used in this subsystem.

3.4.5 Subsystem Data Structures

No data structures are used in this subsystem.

3.4.6 Subsystem Data Processing

No algorithms are used in this subsystem.

3.5 Subsystem 2

The entire module consists of a Li-Po battery which provides power to a Buck Converter which then powers the rest of the system.

3.5.1 Subsystem Hardware

With an input voltage of 36 volts from the battery, the buck converters should be able to output between 1.25V ~34V continuously.

3.5.2 Subsystem Operating System

No operating system is used in this subsystem.

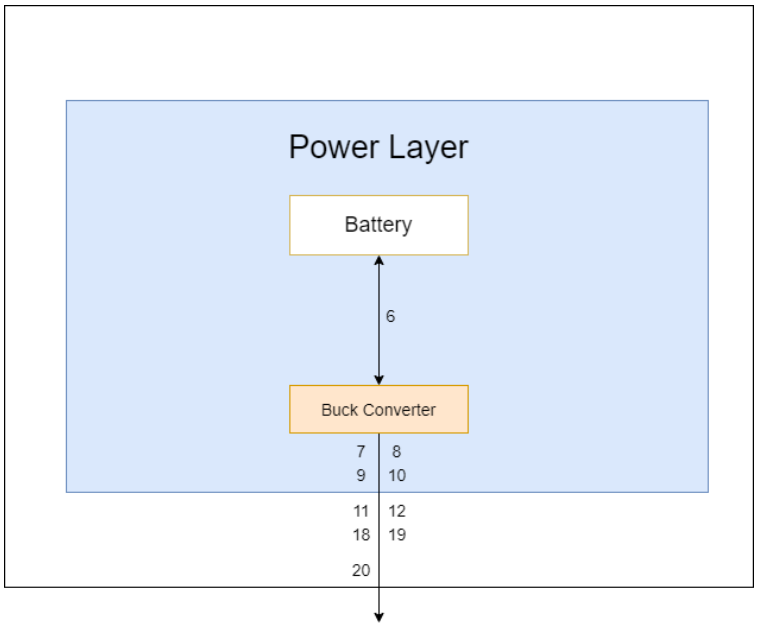


Figure 3: Buck Converter Subsystem in Power Layer

3.5.3 Subsystem Software Dependencies

No software is used in this subsystem

3.5.4 Subsystem Programming Languages

No programming languages are used in this subsystem.

3.5.5 Subsystem Data Structures

No data structures are used in this subsystem.

3.5.6 Subsystem Data Processing

No algorithms are used in this subsystem.

4 Controls Software Layer Subsystems

The controls layer is responsible for binding all modules of the Turing Board to be part of the same system. All data coming in is intercepted by this layer and forwarded to the respective modules which are responsible for processing the forwarded data.

4.1 Layer Hardware

The hardware involved in this layer consists of the following:

- Nvidia Jetson TX2 responsible for running the controls code.
- VESC responsible for the speed control of the wheels.
- Intel real sense camera responsible for capturing real time footage for the follow-me feature and autonomous navigation.
- Texas Instruments Tiva C Series microcontroller which controls the turning mechanism.

4.2 Layer Operating System

The Layer makes use of the Linux operating system to run applications such as the control code.

4.3 Layer Software Dependencies

- Python Dependencies
  - PyVESC
  - Pyrebase
  - Pyserial
  - Pycrc
  - Threading

4.4 Controls Subsystem

There are three main components of project which calls for this piece of software which must all be non-blocking in nature to ensure the entire system stays responsive.

- Reading data from the microcontroller.
- Forwarding data to the microcontroller.
- Fetching data from the Firebase Real-time database.

4.4.1 Subsystem Hardware

The controls subsystem does not consist of explicit hardware. All work is done through software to communicate information.

4.4.2 Subsystem Operating System

The controls subsystem does not make use of an underlying operating system. The layer makes use of the Linux operating system.

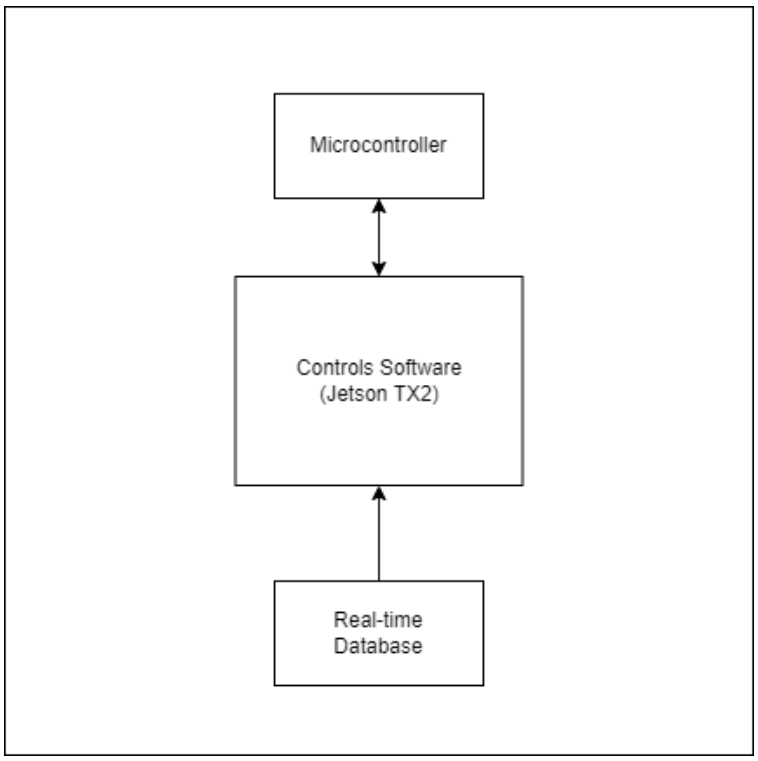


Figure 4: Example subsystem description diagram

4.4.3 Subsystem Software Dependencies

- Python Dependencies
  - PyVESC
  - Pyrebase
  - Pyserial
  - Pycrc
  - Threading

4.4.4 Subsystem Programming Languages

- Python
  - Used for the main controls system.
- C
  - Used to program the microcontroller.

4.4.5 Subsystem Data Structures

- Classes
  - Controls
    - Responsible for running the controls code.
  - SerialCommunication
    - Establishes connection with the microcontroller.

4.4.6 Subsystem Data Processing

After fetching data from the database, the controls software will process the data first to an extent. Since data coming in will be floating point values, it first needs to be translated into value which the microcontroller can understand. So, the entire range of data from the remote control app is taken and the required data is mapped from 0-255 which is then forwarded to the microcontroller which causes the wheels to change speed. As part of the same data packet, angle data from the controls code is also sent to the microcontroller which aids in turning the turning mechanism to a specific angle with respect to the turning mechanisms origin. Any data such as weight values if someone is standing on the long board (an integral part of the design so that the software knows when to turn off the turning mechanism) is received back in the same data format (0-255) which gets translated to weight values inside of the controls code.

5 HMI Layer Subsystems

This layer involves the various sensors and indicators that interact between the Turing Board and the user directly. This includes a load cell or force-sensing resistor for a weight sensor, a piezoelectric speaker, a printed ArUco symbol, a PWM-capable LED strip, a Tiva C Series Microcontroller, and various electrical components. The weight sensor, speaker, and LEDs communicate directly with the Tiva C Series, while the ArUco symbol is recognized by an Intel RealSense camera which sends the data to the Jetson TX2 via the CV Layer. The Tiva C Series is programmed using C and Assembly via Code Composer Studio 10, while the libraries used for the CV layer mainly use C++ and a generic IDE. This is capable of running on both Windows, MacOS, and Linux.

5.1 Layer Hardware

This layer uses a generic piezoelectric speaker (buzzer), a load cell or force-sensing resistor (weight sensor), a printed ArUco symbol generated online (follow anklet), a strip of PWM-capable LEDs approximately 10ft in length, a Tiva C Series, and various electrical components (MOSFETs, resistors, ADC, etc.).

5.2 Layer Operating System

N/A.

5.3 Layer Software Dependencies

This layer depends on code done in Code Composer Studio (v10.2) to program the Tiva C Series and drive the various HMI subsystems. We also used the site https://chev.me/arucogen/ to generate our ArUco symbol.

5.4 Weight Sensor

The weight sensor will be implemented as a safety feature primarily, with future uses in load carrying. When a user is riding on the board, it will detect if they fall off to initiate an emergency stop. When a user attempts to get on the board while in autonomous mode it will trigger an alert noise to notify the user it is not safe to ride.

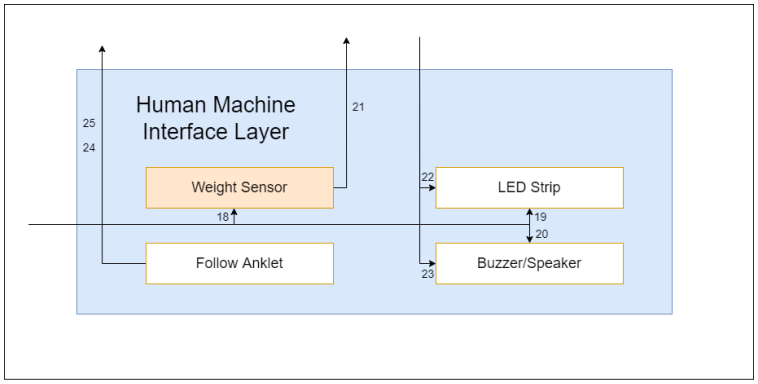


Figure 5: Weight Sensor Subsystem in HMI Layer

5.4.1 Subsystem Hardware

The weight sensor will implement either a generic load cell or force-sensing resistor in or order to determine the weight/pressure being generated by an item/person placed on top. This will detect and send any weight data to a Tiva C Series to parse and decode.

5.4.2 Subsystem Operating System

N/A.

5.4.3 Subsystem Software Dependencies

The weight sensor requires the Tiva C Series to read and parse any data. This, in turn, requires Code Composer Studio 10 to create code to parse/read the data and program the Tiva accordingly.

5.4.4 Subsystem Programming Languages

Code Composer Studio 10 has code written in C and Assembly.

5.4.5 Subsystem Data Structures

The data read in by the sensor is formed into 24-bit "packets" sent along a datastream to the Tiva running at 100kHz. This is done by transmitting one bit per clock using an ADC. This data, which correspond with an external linear force, is then read and parsed by the Tiva and any relevant data is passed to the other subsystems (LEDs and Buzzer/Speaker) to alert of a weight change.

5.4.6 Subsystem Data Processing

N/A.

5.5 Buzzer/Speaker

The buzzer/speaker will be used to alert the user if they attempt to step on the board when it is in an autonomous mode. When the weight sensor detects a person attempting to stand on the board when in autonomous mode, the buzzer will begin to sound an alert to notify the user it is unsafe to stand on it.

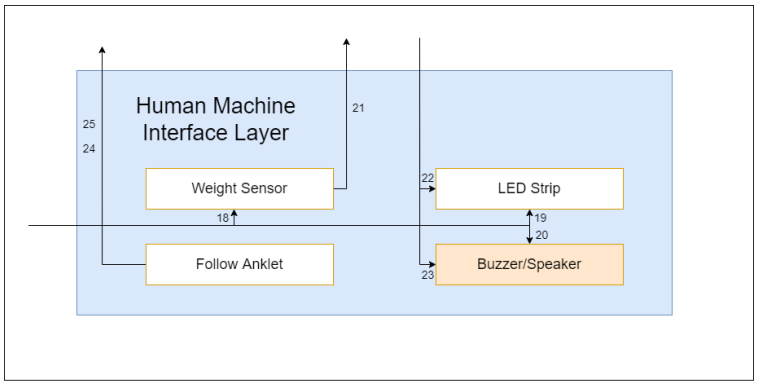


Figure 6: Buzzer/Speaker Subsystem in HMI Layer

5.5.1 Subsystem Hardware

This subsystem uses a generic piezoelectric speaker powered and run by a Tiva C Series to emit a sound as an alert on various conditions.

5.5.2 Subsystem Operating System

N/A.

5.5.3 Subsystem Software Dependencies

Code Composer Studio 10 is used to send signals to the piezoelectric speaker in order to emit a sound of a pre-determined frequency and length.

5.5.4 Subsystem Programming Languages

Code Composer Studio 10 has code written in C and Assembly.

5.5.5 Subsystem Data Structures

A signal is sent from the Tiva to the speaker using a 32-bit value to determine the frequency at which to emit the sound. The Tiva also uses a 32-bit variable to determine how long to emit the signal for before sending a stop command.

5.5.6 Subsystem Data Processing

N/A.

5.6 LED Strip

The LED strip will be used to indicate to the user what mode the Turing Board is currently in. These modes will be differentiated by implementing a different color for each mode.

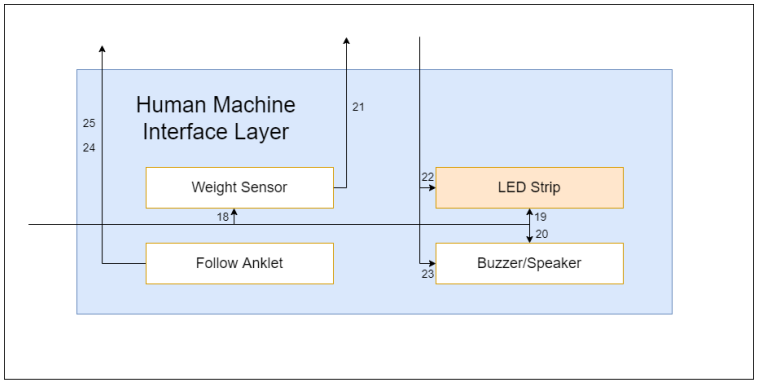


Figure 7: LED Strip Subsystem in HMI Layer

5.6.1 Subsystem Hardware

These LEDs are specifically PWM-capable and rated at up to 12V. It also implements a Tiva C Series to control the LEDs.

5.6.2 Subsystem Operating System

N/A.

5.6.3 Subsystem Software Dependencies

The LEDs are powered and controlled by the Tiva using Code Composer Studio.

5.6.4 Subsystem Programming Languages

Code Composer Studio 10 has code written in C and Assembly.

5.6.5 Subsystem Data Structures

The LEDs are each fed a 256-bit value corresponding to the Red, Blue, and Green data lines from the Tiva. This, in turn, activates the respective LED color to create various combinations based on each R/G/B input values.

5.6.6 Subsystem Data Processing

N/A.

5.7 Follow Anklet

The Anklet will be a worn feature used for the CV to track and follow the rider in follow-along mode.

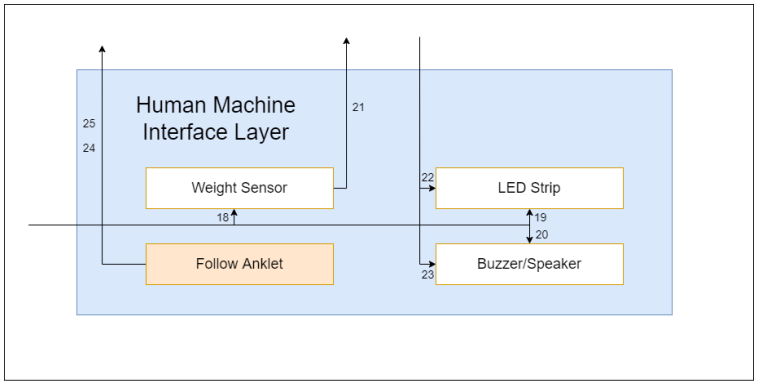


Figure 8: Follow Anklet Subsystem in HMI Layer

5.7.1 Subsystem Hardware

This subsystem simply uses an ArUco symbol printed out on a strip of paper long enough to wrap around one's ankle.

5.7.2 Subsystem Operating System

N/A.

5.7.3 Subsystem Software Dependencies

This depends on the ArUco library based on the OpenCV library used to detect and track a generated symbol.

5.7.4 Subsystem Programming Languages

This uses C/C++ in the provided libraries required to detect and track the symbol.

5.7.5 Subsystem Data Structures

N/A.

5.7.6 Subsystem Data Processing

Uses an algorithm defined in the ArUco/OpenCV libraries to recognize and track the generated symbol.

6 Computer Vision Layer Subsystems

This layer is the heart of the core autonomous functionalities of the Turing Board. We use computer vision and depth imagery to determine the boardâs surroundings and calculate the best path to move forward. The user will have, strapped around their ankle, an anklet-like contraption consisting of a pattern of ArUco markers for the Follow Along feature. This layer tracks the movement of the user through the anklet to determine how to instruct the combination of motors to move so as to follow the user at an appropriate pace. It is also responsible for detecting possible obstacles when operating on its own to find the user as part of the Summon feature.

6.1 Layer Hardware

The required hardware components include the NVIDIA Jetson TX2 as the main compute module and the Intel RealSense D432 Depth Camera.

6.2 Layer Operating System

The NVIDIA Jetson TX2 is running Ubuntu 18.04.

6.3 Layer Software Dependencies

This layer requires the OpenCV v4.2.2 or higher compiled with cuDNN enabled. A more detailed and exhaustive list can be obtained from. https://github.com/TuringBoard/turing-board-vision/blob/main/Configuring

6.4 RGB Imagery Subsystem

RGB Imagery of the front of the board is used as input in making various position specific calculations pertaining to navigation.

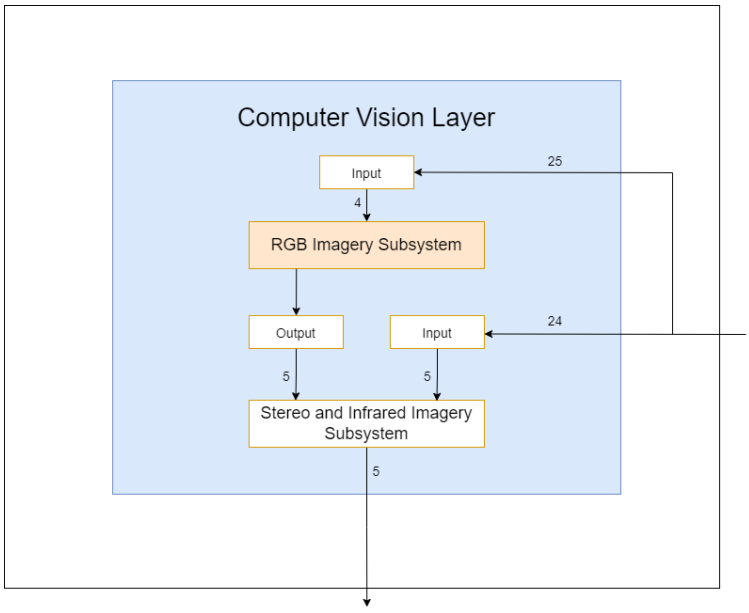


Figure 9: CV Layer Subsystem

6.4.1 Subsystem Hardware

This subsystem requires the NVIDIA Jetson TX2 and the Intel RealSense D432 Depth Camera.

6.4.2 Subsystem Operating System

Ubuntu 18.04.

6.4.3 Subsystem Software Dependencies

This layer requires the OpenCV v4.2.2 or higher compiled with cuDNN enabled. A more detailed and exhaustive list can be obtained from. https://github.com/TuringBoard/turing-board-vision/blob/main/Configuring

6.4.4 Subsystem Programming Languages

Python 3.x.

6.4.5 Subsystem Data Structures

Arrays, HashMaps, Graphs, Trees.

6.4.6 Subsystem Data Processing

SIFT, RANSAC, Canny Edge Detection.

6.5 Stereo Imagery Subsystem

Stereo and Infrared Imagery of the front of the board is used as input in making various depth specific calculations pertaining to navigation.

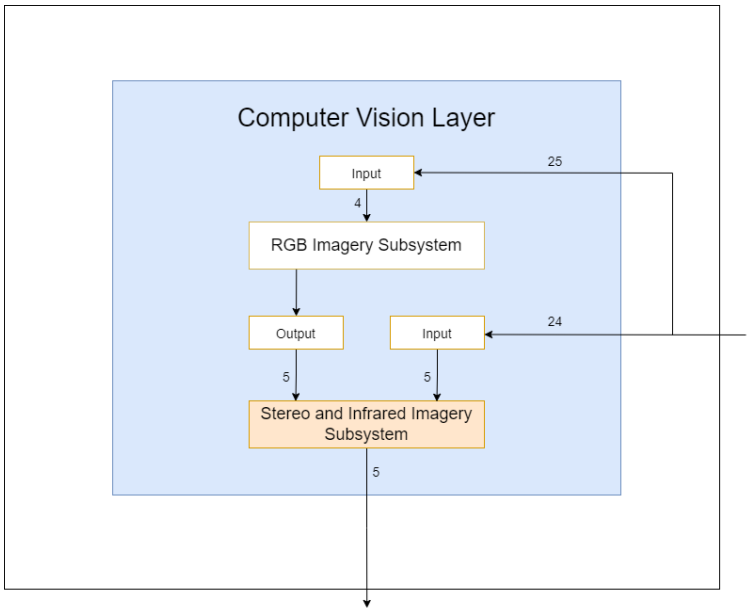


Figure 10: CV Layer Subsystem

6.5.1 Subsystem Hardware

This subsystem requires the Intel RealSense D432 Depth Camera.

6.5.2 Subsystem Operating System

N/A

6.5.3 Subsystem Software Dependencies

pyrealsense: https://pypi.org/project/pyrealsense
librealsense: https://github.com/IntelRealSense/librealsense

6.5.4 Subsystem Programming Languages

Python 3.x.

6.5.5 Subsystem Data Structures

Arrays, HashMaps, Graphs, Trees.

6.5.6 Subsystem Data Processing

SIFT, RANSAC.

7 APPENDIX A

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

REFERENCES