Report on A true autonomous vehicle solution

WRO Season 2023

Future Engineering

Using the method of machine learning, a CNN neural network is built to train and predict the pictures of the competition venue.

This is our AI CyberLEGO Car

OCT 4, 2023

Team FEV USA

**Team Presentation**



**Team Members: Fengmo Guo**

**Evan Yu Girard-Sun**

**Vinaya Ayinampudi**

**Coach: Fei Guo**

**FengMo Guo**, a tenth-grade high school student from New York, United States. He had participated in WRO in previous seasons. He loves robots and is a very innovative person.

**Evan Girard-Sun**, Evan is a 10th grade high school student who enjoys 3D design and modelling, soccer, and playing the violin, and was a member of 2022 WRO Junior RoboMission National Championship team!

**Vinaya Ayinampudi，a** tenth grader at Eastvale STEM Academy who enjoys playing volleyball, and coding in her free time. She loves working with new technology and is very enthusiastic about robotics!

**Summary Project A combination of half-AI and half-robot, a real AI autonomous driving robot.**

**Technical solution design**

**LEGO robot and car body structure design**

**AI operating environment design, including structure design, power supply design, communication design, remote control handle.**

**Remote monitoring system design, website design.**

**Communication Design:**

**The handle communicates with the Raspberry Pi and the Raspberry Pi communicates with the LEGO robot car.**

**Control logic design:**

**The remote control robot records the picture and data of the playing field**

**Design neural networks**

**Train a neural network with a field model.**

**During the game, the neural network calls in the trained behavioral model to make actions that mimic human operations.**

**This action data is transmitted to the LEGO robot. There are LEGO robots performing action parts.**

**At the same time, the LEGO robot carries out secondary control. Gyro sensors and a LEGO PIXY2-enabled AI camera verifies AI-made judgments. And participate in judgment, controlling vehicle start and customization at start and end.**

**AI software code design:**

**Take photos and record data sections.**

**SECTION USING WEB REMOTE MONITORING.**

**Run the AI neural network and train the part of the data that yields the result.**

**The part where the neural network model is called in real time to make a prediction of the captured video.**

**The part that transmits the prediction data to the LEGO robot.**

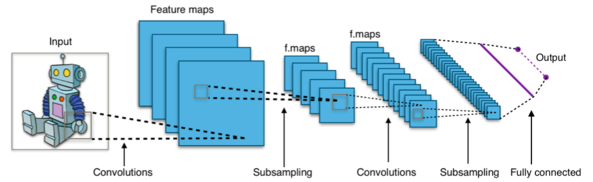
**The LEGO robot receives the data and performs the part of the action.**

**Overall technical solution design**

**Technical Background:**

**Machine learning (ML) is an umbrella term。 It refers to a computer software approach that lets machines find and optimize algorithms on their own.**

**Convolutional neural network CNN is a multi-layered data processing structure. In particular, it has advantages in AI image recognition. Therefore, it has a very good performance in the field of autonomous driving.Convolutional neural network** (**CNN**) is a [regularized](https://en.wikipedia.org/wiki/Regularization_(mathematics)) type of [feed-forward neural network](https://en.wikipedia.org/wiki/Feed-forward_neural_network) that learns [feature engineering](https://en.wikipedia.org/wiki/Feature_engineering) by itself via [filters](https://en.wikipedia.org/wiki/Filter_(signal_processing)) (or kernel) optimization.

****

**TensorFlow is a free and open-source software library for machine learning and artificial intelligence.**

**Lego Mindstorms EV3 is LEGO Education's programmable robot hardware, which is very easy to build structures and very easy to program. AT PRESENT, WORLD-RENOWNED YOUTH ROBOTICS COMPETITIONS SUCH AS FIRST LEGO LEAGUE AND WRO ARE USING LEGO'S ROBOTICS PRODUCTS.**

**Technical ideas:**

**Future Engineering is an autonomous driving competition that avoids obstacles at the venue. We designed to use the Raspberry Pi as avehicle for A-I operations. Under the operating system of Raspberry Pi's Linux OS, the python IDE integrates the runtime environment. Install Google's tensorflow. Install the lens on the Raspberry Pi. Start by running a program that records a video of a real vehicle moving and remote control data. The data is then fed to a CNN neural network, which uses machine learning methods to find connections between video images and remote control data. Then apply this model to the recognition of real lens photos. Let the AI give the vehicle's operating data. This data is then transmitted to a vehicle built with a LEGO EV3. Make LEGO cars self-driving.**

**Structural design of LEGO vehicles**

**Please refer to the following four pictures for the design of the body structure.**

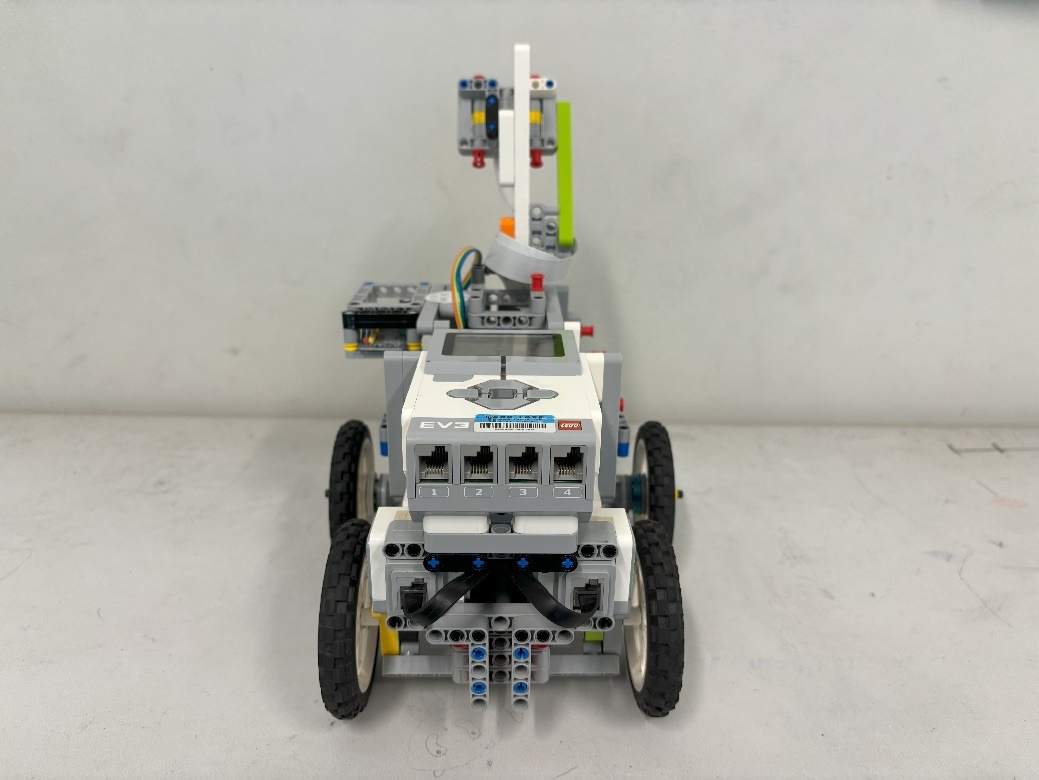


Figure ​1



Figure ​2

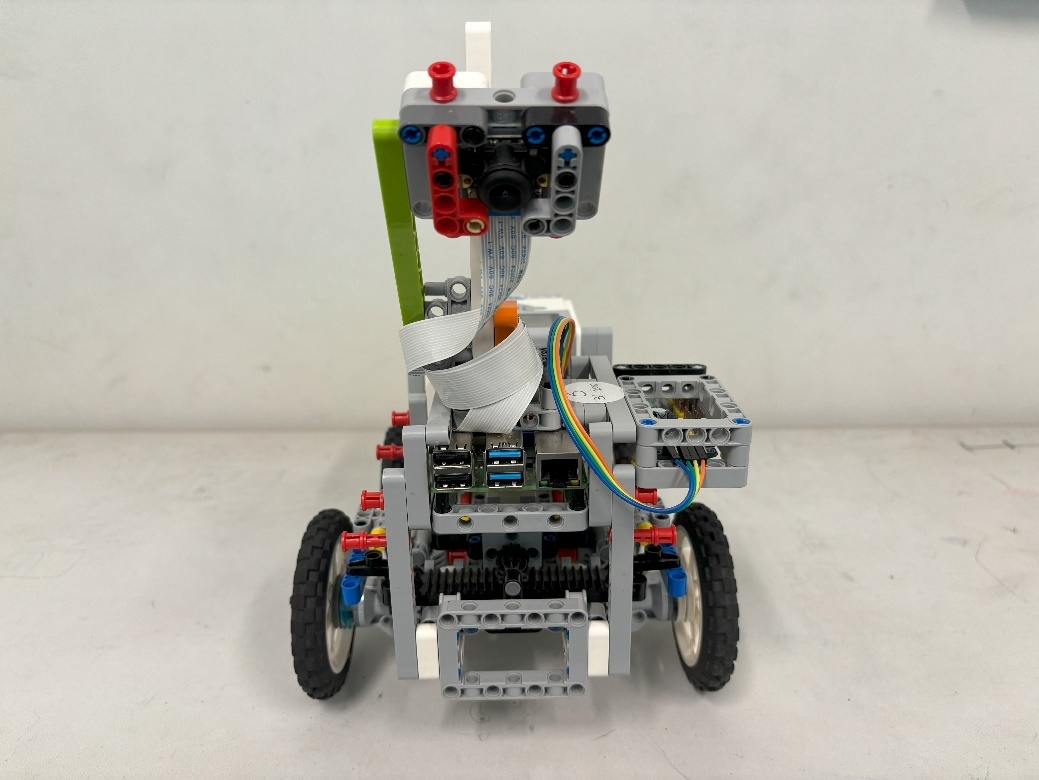


Figure ​3

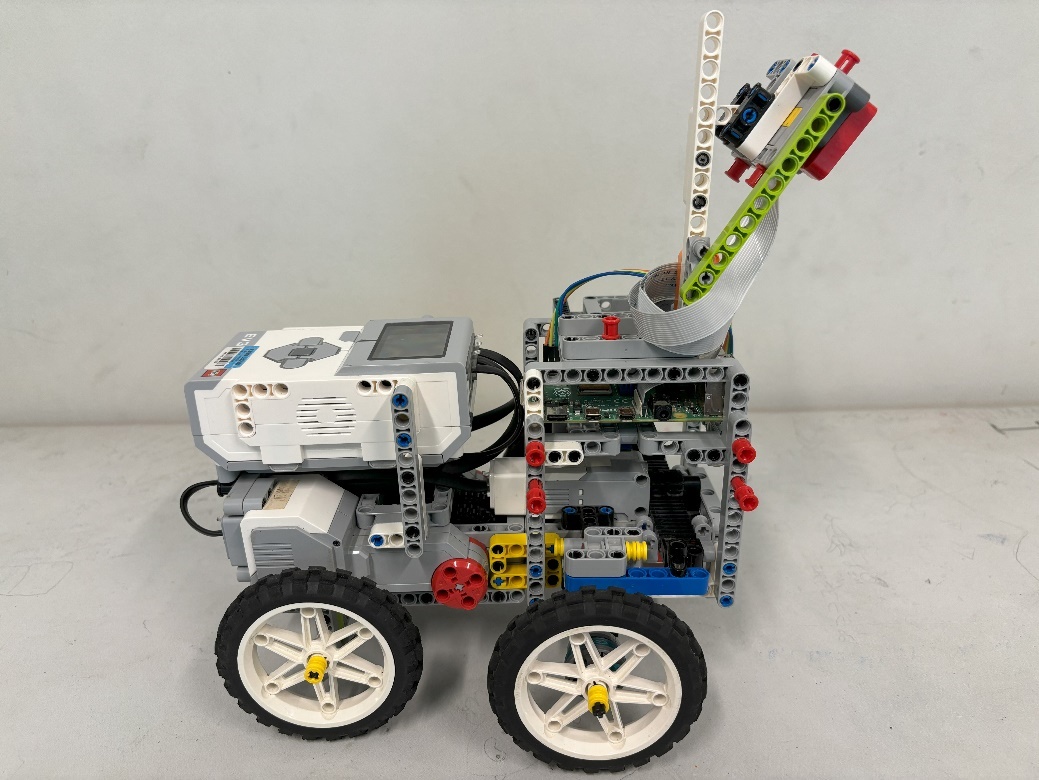


Figure ​4

**LEGO rear wheel power structure design**

**Please note that the two LEGO motors are not differential-driven. Instead, standard vehicle shaft transmission is used.**

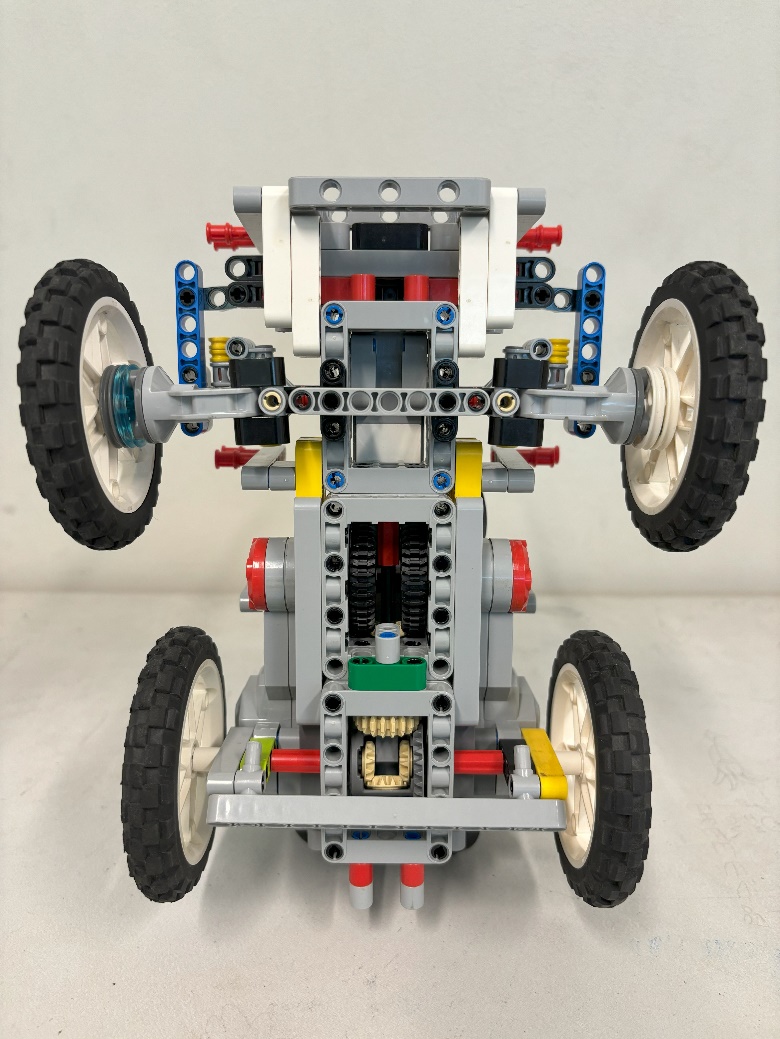


Figure ​5

**Structural design of steering gear**

**The LEGO EV3 medium-sized motor moves the gears left and right.**

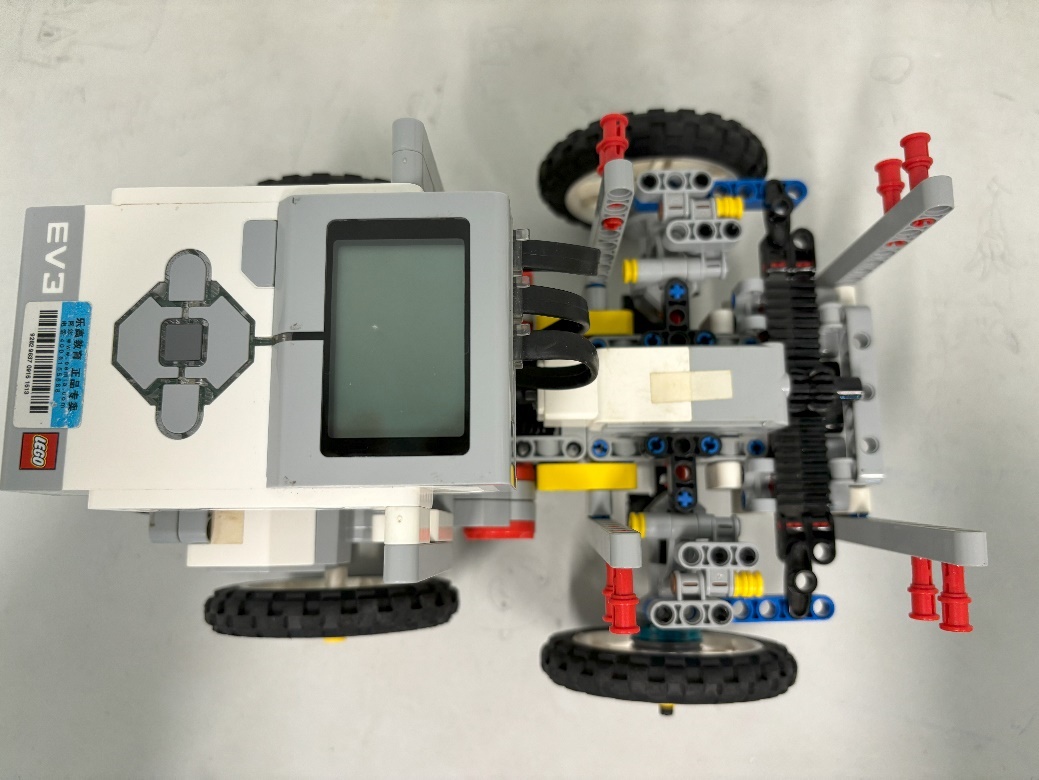


Figure ​6

**Lego structure design for Raspberry Pi**

**This is a very distinctive design, and since the Raspberry Pi itself cannot be connected to the Lego structure, we designed a Lego structure to tie the Raspberry Pi in the middle. The effect of this is that the structural parts on the outside of the Raspberry Pi can be easily connected to the LEGO parts.**

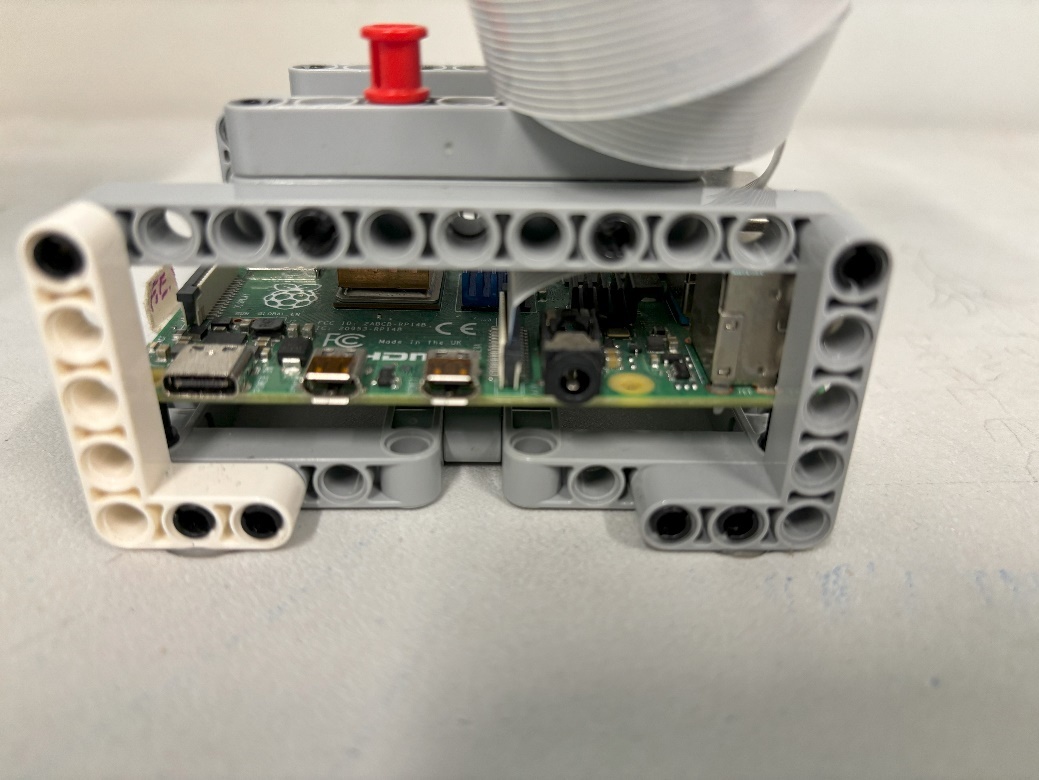


Figure ​7

**Before making AI imitate how humans operate remote-controlled cars, it is necessary to make the vehicles manipulable by humans. We use the Sony Playstation 4 Dual Shock 4 Controller as our vehicle remote.**

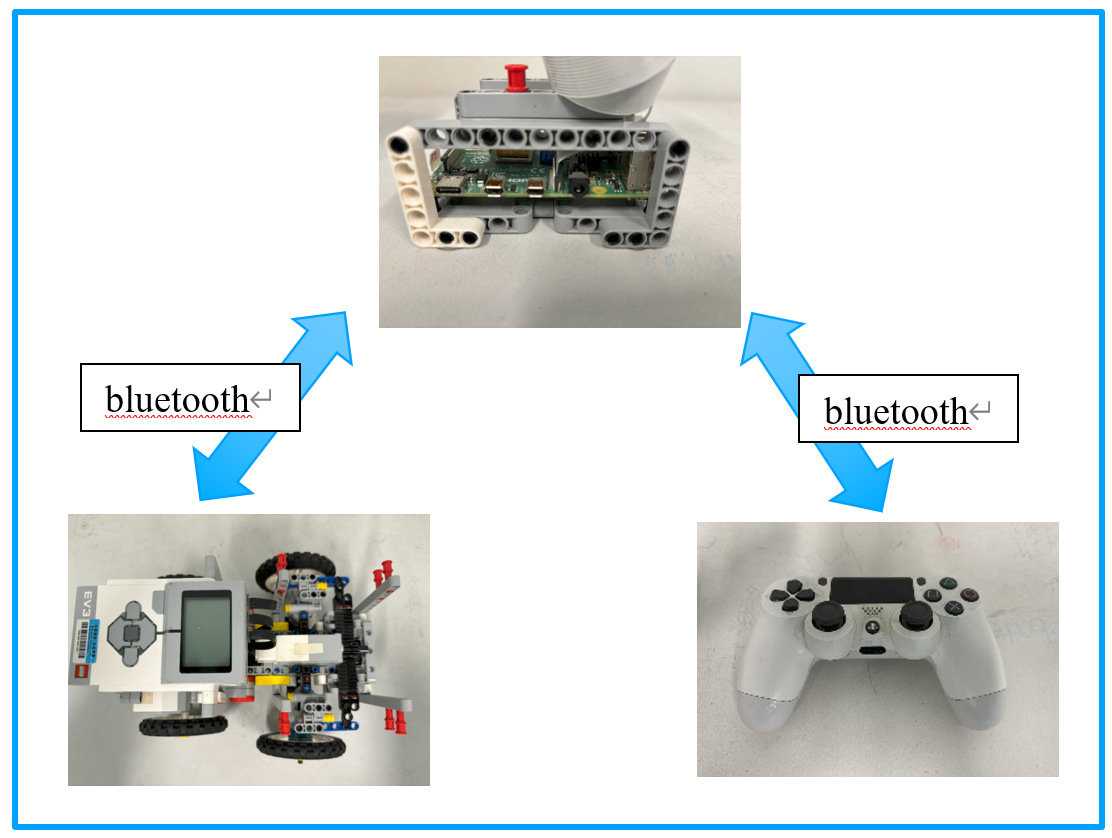


Figure ​8

**In order to display the vehicle data more accurately, we also designed a web server, which also runs on the Raspberry Pi. Each frame of the footage is transmitted in real time by this web server. In this way, the live view of the lens can be accessed through the web page.**

照片

**System communication design**

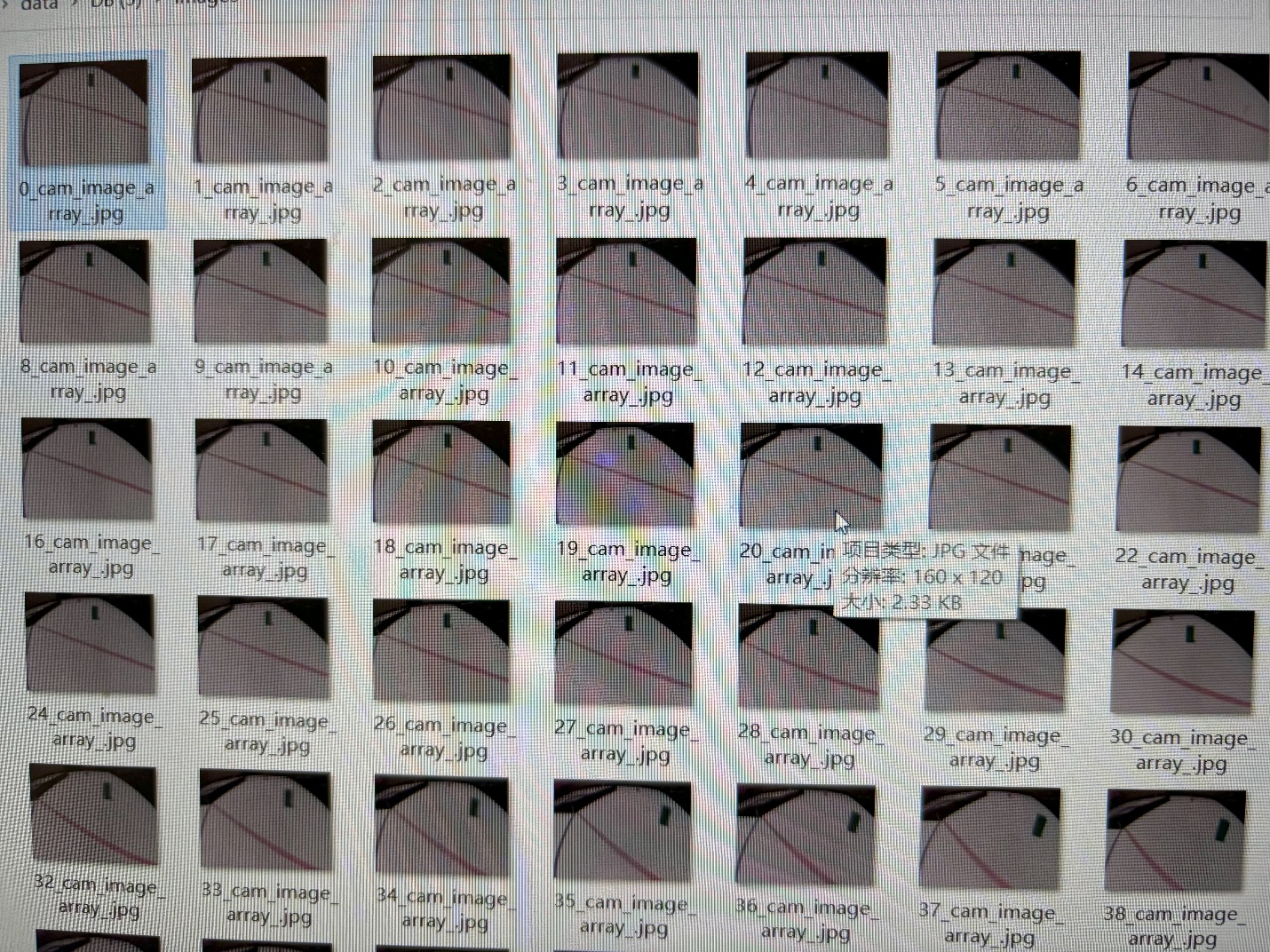
****

After developing our robotic system, we found that it has effectively accomplished the task. The system can simulate real-life navigation by capturing video footage of the canal and making predictive navigational decisions. Although it may require further adjustments in a real-world setting, such as incorporating elements from autonomous car projects, we believe that it holds significant practical value, similar to that of autonomous vehicles.

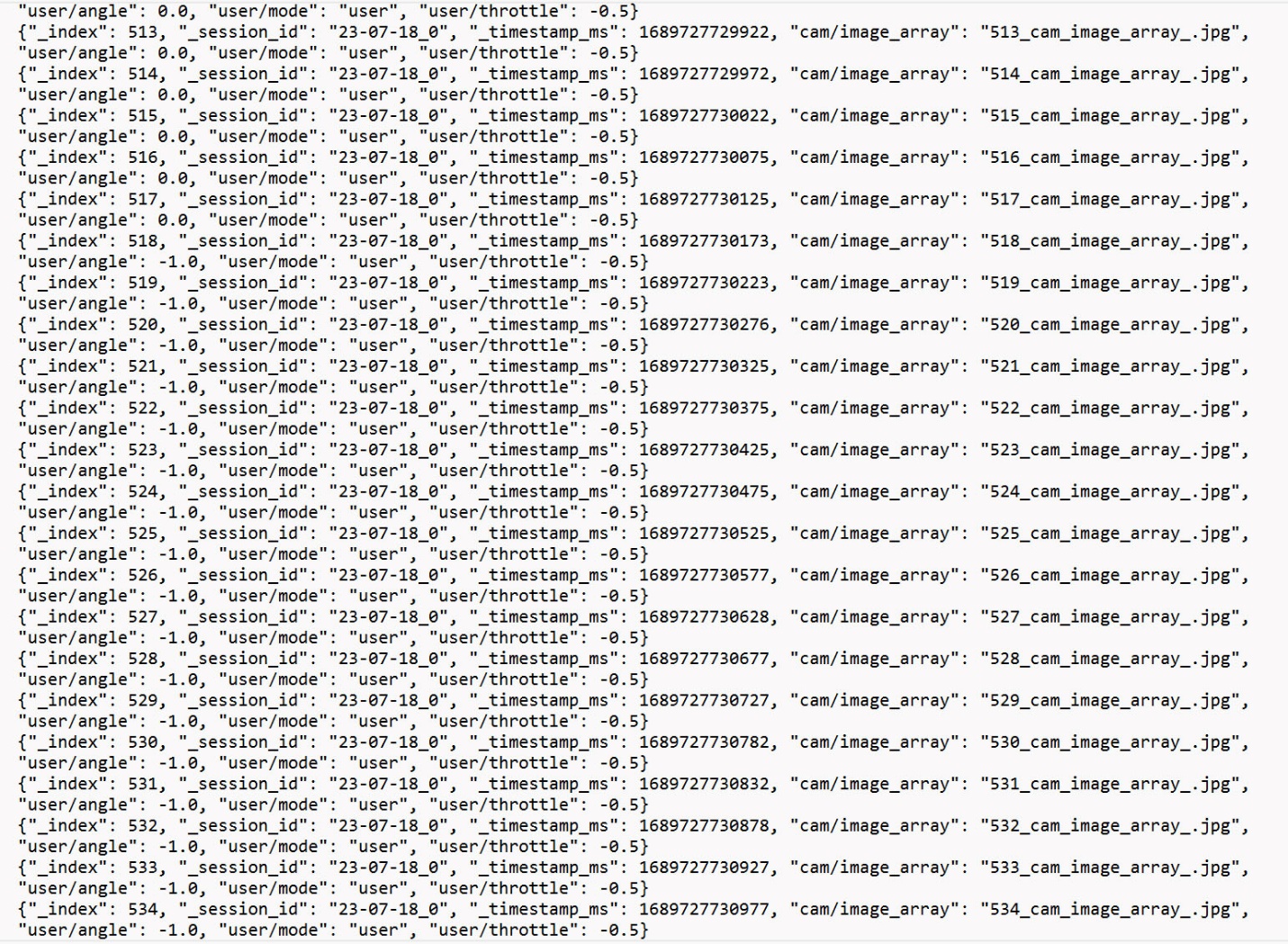
Therefore, we decided to apply end-to-end (END TO END) autonomous driving technology to the task of guiding ships through these areas.

The heart of our system is a single-board computer, specifically a Raspberry Pi 4, coupled with a camera. This setup captures real-time images from the monitor displaying the canal. We also used a joystick that provides data for the learning phase of the project. This combination allows us to pair real images with steering and throttle data, creating a comprehensive learning file that the AI can use to improve its performance.

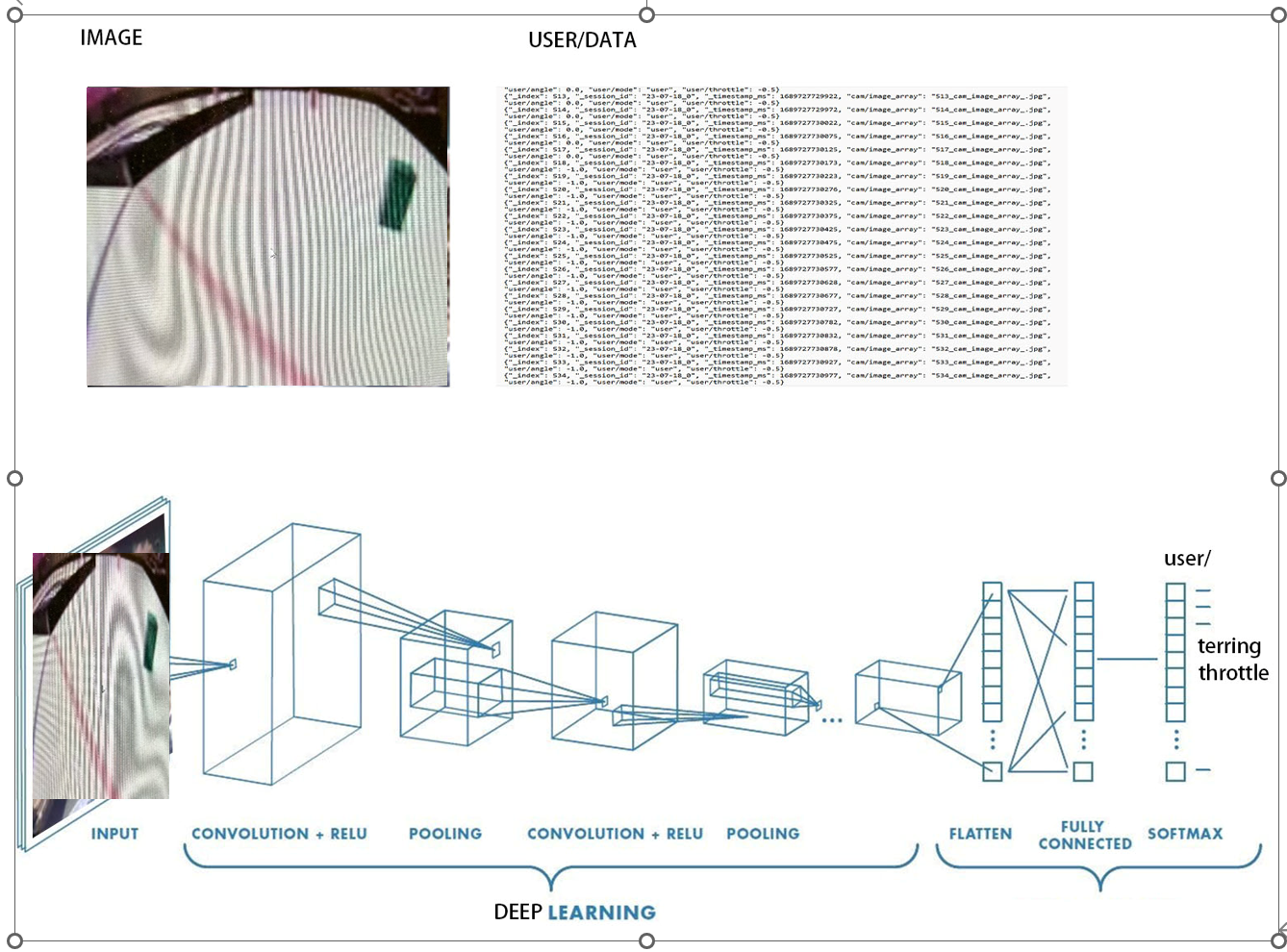
Photos of the Data We Captured



Photos of the Data We Collected



Schematic Diagram of the CNN (Convolutional Neural Network) Recognition and Judgment after Training



In our project, the hardware aspect mainly includes the display, the single-board computer, especially the Raspberry Pi 4, and the camera. The combination of the Raspberry Pi 4 and camera is used to capture live images from the playing field. In addition, we have a joystick that provides data during the learning phase of the project. This combination allows us to pair live images with steering and throttle data to create a comprehensive learning file for the AI to improve its performance.

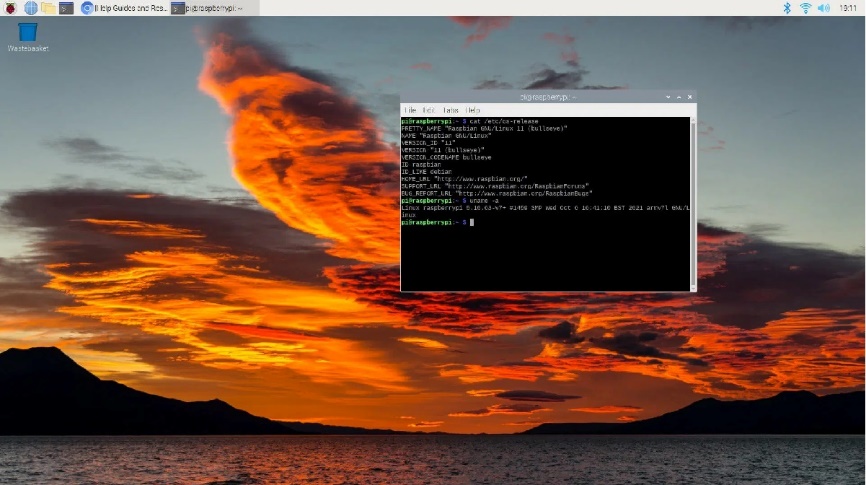
Initially, raw data, such as images, is collected and stored on a Raspberry Pi. Then, this data is transferred to a standard computer for further processing.

On the computer, a series of training programs are run, using the collected data to train an AI self-driving model. This training process results in a file which contains the AI model. This file, essentially our trained self-driving model, is then put back onto the Raspberry Pi.

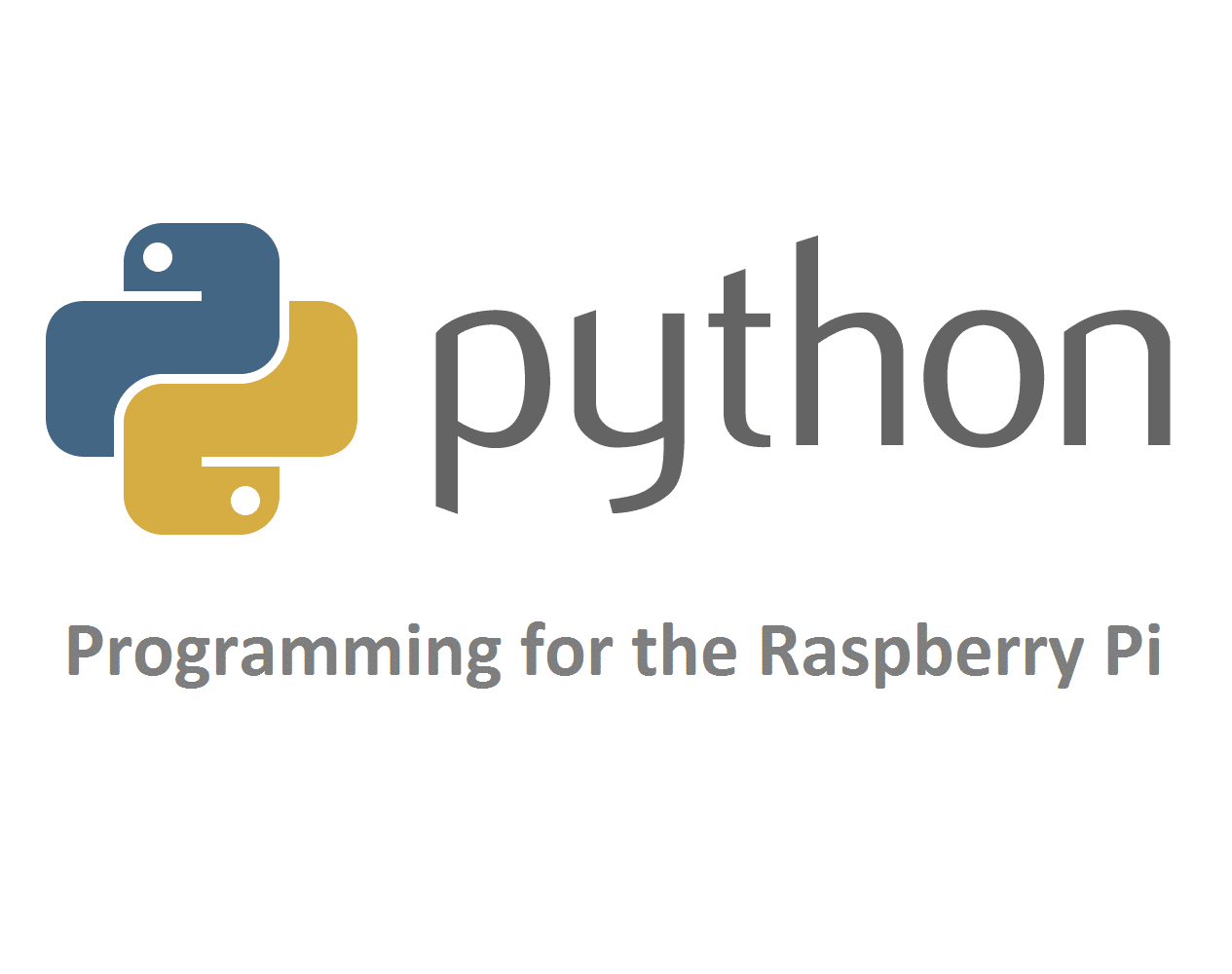
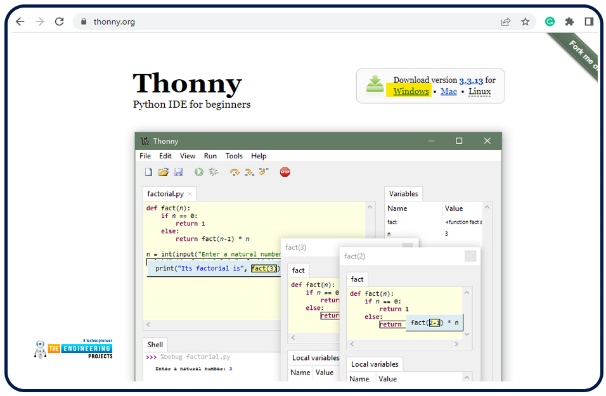
The Raspberry Pi, with the help of the AI model, interprets real-time images and determines specific control instructions, like throttle and steering data. Then, this information is transmitted to the LEGO robot via Bluetooth.

The LEGO robot here mainly plays the role of a model demonstrator. Upon receiving the data, the LEGO robot displays corresponding movements, like moving forward or backward (corresponding to throttle control), and turning left or right (corresponding to the steering wheel). Therefore, the LEGO robot essentially serves as a tangible representation of the AI self-driving model's effects. Basically, that's all there is to the hardware part of the project!

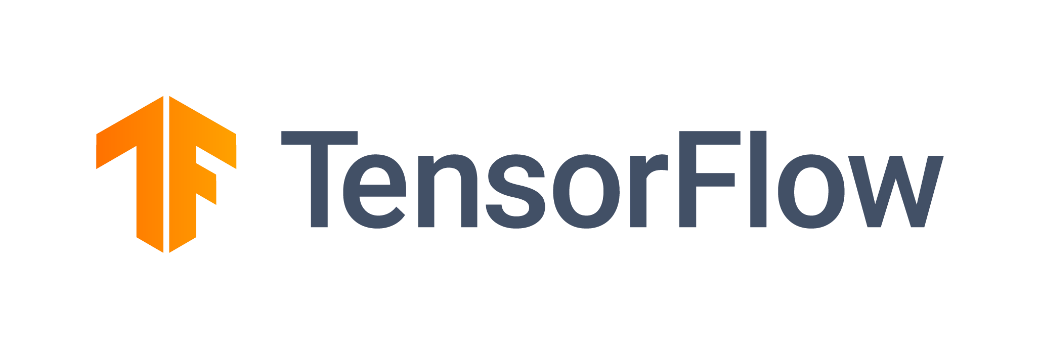
**Pi-os**（**linux**）

****

**Python**

****

**Tensorfolw**

****

In terms of software, there are several important elements to note in our project.

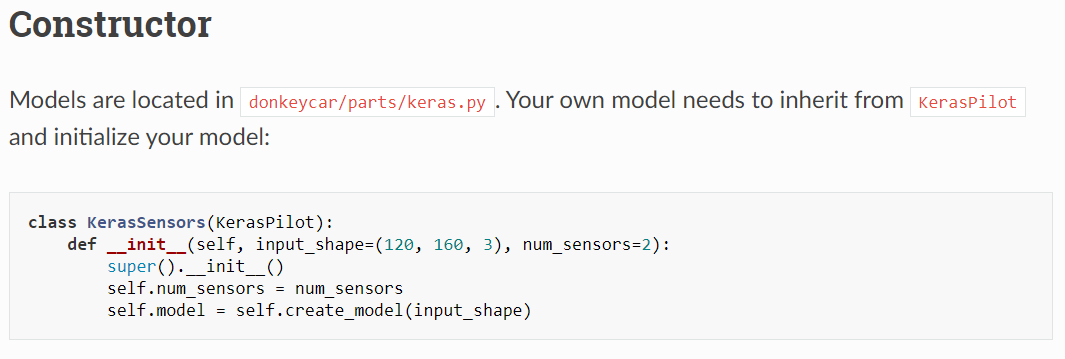
Firstly, we run the Linux operating system on our Raspberry Pi. Within this system, we use Python as our programming language and TensorFlow as the tool for our AI modelling and training tasks.

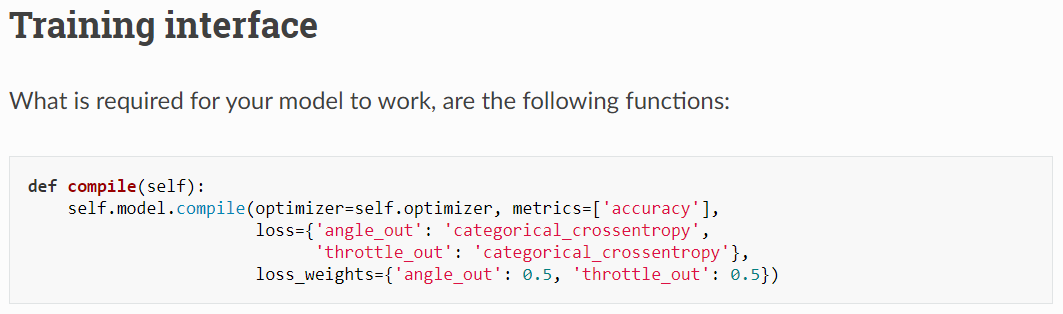
Let me walk you through how the software side works. Using Python, we control the Raspberry Pi's camera to capture images and collect data. Additionally, we connect a joystick via Bluetooth, and both the joystick data and the images from the camera are stored together.

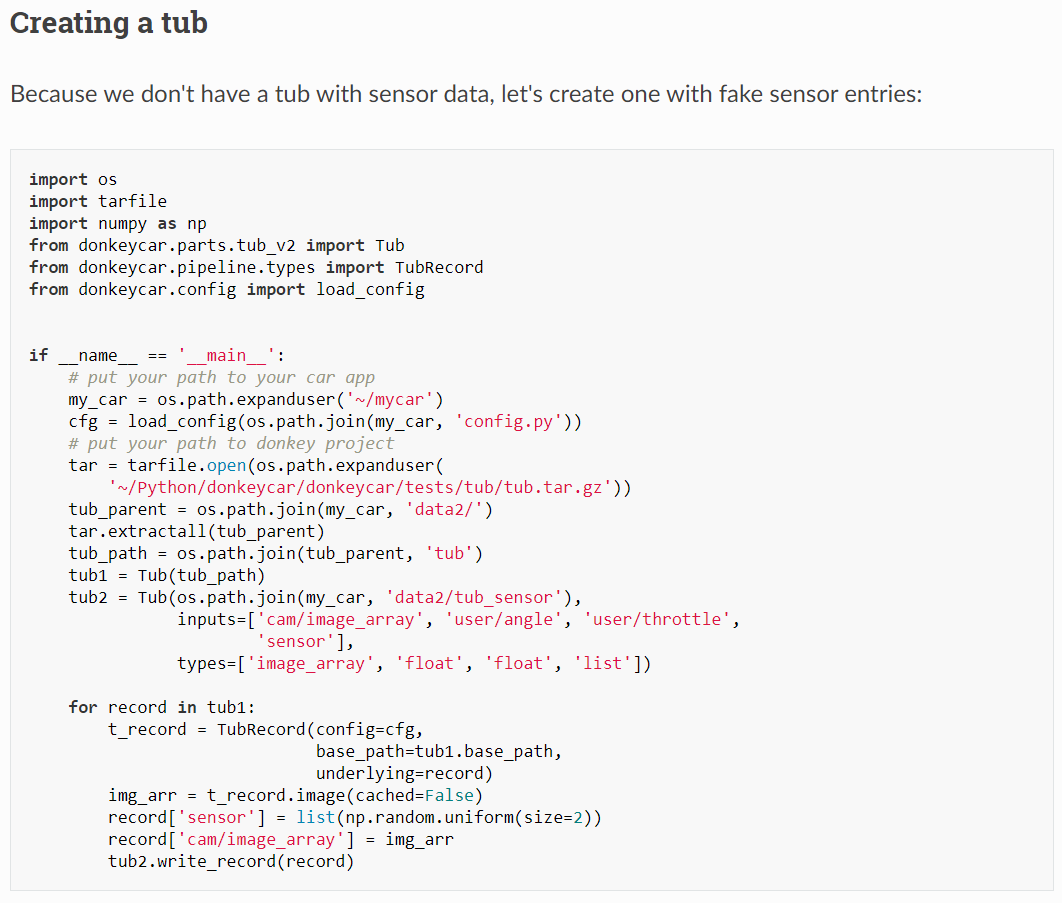
Next, we package all of this data and hand it over to TensorFlow. Inside TensorFlow, we've built a Convolutional Neural Network (CNN) model for training. The inputs to the model are a video matrix, steering control input, and throttle data input. These three types of inputs pass through eleven layers of the network, and in the end, two types of outputs are produced: the steering wheel output and the throttle output. This process creates our AI self-driving model.

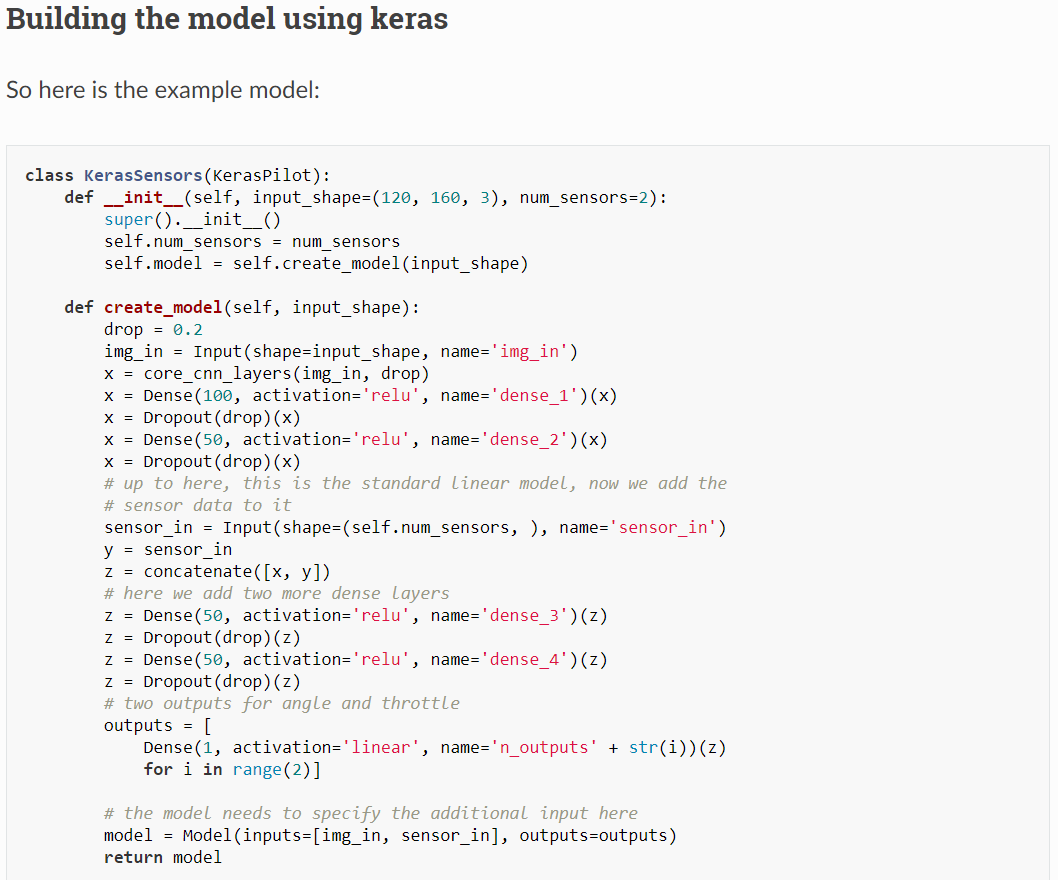
Finally, we transmit the data to our Lego robot via Bluetooth. The Lego robot is programmed using its proprietary language. It primarily serves to showcase the received data, controlling the motors to drive forward or turn left and right. This sums up the software portion of our project.

Python Code for Capturing Photos:



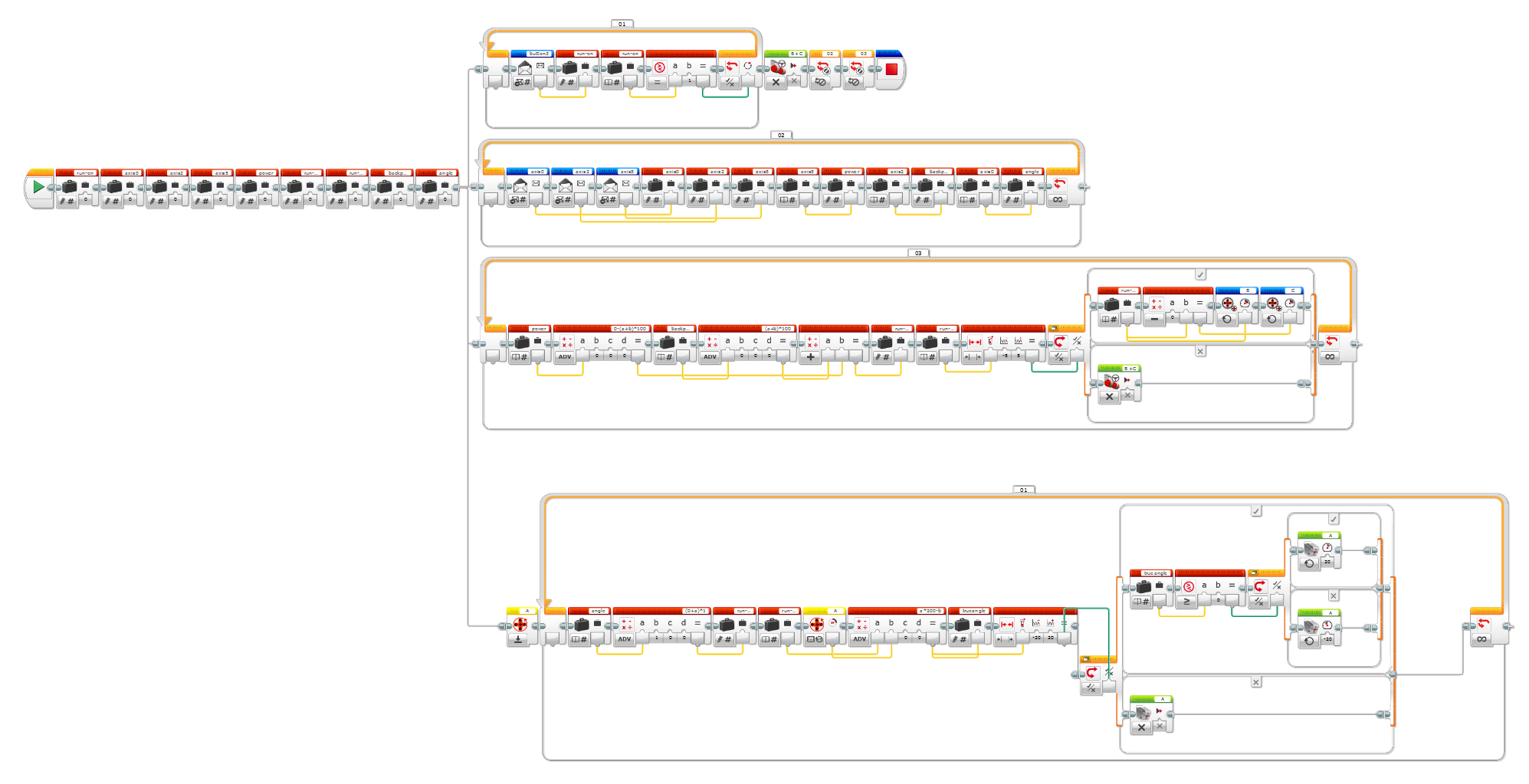








Code for the Lego Robot



Challenges we met: Throughout the development of this project, we encountered a series of challenges. First off, there was the issue of image resolution. If we chose to work with larger image resolutions, our machine computations slowed down significantly, resulting in training sessions that could take longer than 8 hours. To solve this, we downscaled the data and used a resolution of 160 by 120 pixels.

Next, we grappled with a problem of overly lengthy road models. To address this, we segmented a portion of the data model for training. Finally, we faced issues of software compatibility. For instance, we found that a model trained on TensorFlow 2.9 wouldn't work on TensorFlow 2.4. To remedy this, we upgraded all systems to use TensorFlow 2.9 for a more unified approach to training.