Hero Pong Vehicle Document

Our vehicle is contained with three parts in total. The base car, the Jetson Nano Developer Kit and the Intel RealSense depth camera D435i. In terms of it can be divided into two parts, the motor part and the Intel RealSense depth camera D435i part.

Here is a summary of our vehicle:

• Onboard four 18650 batteries, 10400mAh large capacity, two parallel and two series output current are larger, and the motor power is stronger

• Onboard HY2120+AOD514 lithium battery protection circuit, with anti-overcharge, anti-over-discharge, anti-overcurrent and short circuit protection functions

• Onboard FP5139 automatic buck-boost regulator circuit, which can provide a stable 5V voltage for Jetson Nano

• Onboard 0.91-inch 128×32 resolution OLED, real-time display of vehicle IP address, memory, power, etc.

• Onboard AINA219 acquisition chip, convenient for real-time monitoring of battery voltage and charging current.

Additionally, we imported the Jetson Nano development kit, the reason why we choose it as the computer of our vehicle is because it equipped with a quad-core ARM Cortex-A57 CPU and an NVIDIA Maxwell GPU with 128 CUDA cores. This combination provides high-performance AI processing capabilities, making it suitable for a wide range of AI applications. Additionally, the Jetson Nano is made to be power-efficient despite its small size. It is appropriate for battery-powered and resource-constrained applications because it only uses a few watts of power. This will enable us to work on our code for a considerable amount of time.

The base car we will be using this year is specially designed for this competition. The motor part adopts brushless DC motor, servo motor and a brushed electronic speed controller. We choose CYS S3006 analog servo as our servo motor. The servo features metal gears, which provide enhanced durability and resistance to wear and tear. This makes it suitable for applications that involve repetitive movements or require reliable performance over an extended period. For the electronic speed controller, The Hobbywing Quicrun-WP 1625 Brushed Electronic Speed Controller were used. It is built with reliable components and undergoes rigorous testing to ensure durability and long-term performance.

To mount our depth camera, we included an L-shaped 3D printed bracket to mount it at the optimal angle. The resin used to print the stand is sturdy and reliable enough to hold the camera. It won't have an impact on the vehicle's performance when driving because the resin is lightweight.

In terms of our program code, it is written in Python. The main idea of ​​vehicle motion is based on Ackermann steering. When our vehicle's camera captures the view of the "traffic lights" (red and green blocks), servo motors turn the vehicle left or right by automatically adjusting the vehicle's front axle. Its acceleration is determined by a programmed brushless DC motor at the rear of the vehicle. The code "car. steering" is used to control the servo motor and "car. throttle" is used to control the DC motor.

For example, if we enter "car. ​​throttle 0.15", the car will move forward at a speed of "0.15", the larger the number, the faster it drives. The range of "car. throttle" is from 0 to 1. For the servo motor, if we enter the code "car. steering 0.2", it will turn right a bit. The range value of steering is from –1 to 1, when the value is negative, the vehicle turns left, and vice versa.

In terms of our camera, The Intel RealSense Depth Camera D435i is installed. This is a depth camera that can identify the color of traffic lights the vehicle is facing. In our program, we set the green to range from ([40, 80, 40]) (BGR) to ([102, 255, 255]) (BGR), and the red to range from ([0, 50, 120]) (BGR) to ([10, 255, 255]) (BGR). We train the AI ​​database by feeding thousands of photos of red and green patches from different angles and orientations into the camera database. Therefore, it can have a correct identification. When the camera captures a pre-set range of size of square, it will recognize the color of the square and perform corresponding actions. For example, it turns left when capturing a green block, and vice versa.

In order to determine whether it is running in clockwise or anti-clockwise, we decided to calculate the slope of the blue line. The code to set the range is just like the previous paragraph, but this time only in blue. The idea is to use colour frame to detect where the blue line lied on, then detect is it a positive or a negative slope. If it is a positive one, then the car with turn left, vice versa. This method can make sure that the light will not affect the accuracy of it. Also, we can just ignore the orange line. The outcome is, it is more accurate and easy way to let the AI know which way it should turn.

In summary, our car uses the Intel RealSense depth camera D435i, the Jetson Nano development kit, a motor that matches every piece of equipment, and self-judgment between traffic signals to guarantee precise and seamless driving. It is a self-sufficient artificial intelligence vehicle that can accomplish tasks without human assistance.