吴贤铭智能工程学院
SHIEN-MING WU SCHOOL OF INTELLIGENT ENGINEERING

2021–2022 Design and Manufacturing II

Final Report:

An Obstacle Avoidance & Tracking & Loading Vehicle Based on Openmv

Group members:

Lue Fang, Run He, Rui Qi, Yuchen Song, Ruoyao Tian, Jialong Zeng

Shien–Ming Wu School of Intelligent Engineering

17/06/2022

An Obstacle Avoidance & Tracking & Loading Vehicle Based on OpenMV

Abstract

In this project, a smart robot is required to be designed with the functions to complete all the tasks including obstacles avoidance, path tracking, cargo delivery and unloading. We mainly did as follow:

For obstacle avoidance, a path is defined and moved along with. With the property of Mecanum wheels, translational movements were followed with ultrasonic sensor to detect distances. Moreover, a PD controller is designed to control the wheels.

For tracking part, an OpenMV module is used to fit the car's route. By comparing the route at pixel coordinate, we get both positional and directional error then fed the errors to a PID controllers. Finally, PWM method is used to control the speed of wheels based on the error.

For loading task, vision-based method was used. A learning-based method with neural network and color-based method were developed to detect the object. The position of the object can be estimate by the camera and the required pose and joint position of the gripper can be calculated by forward and inverse kinematics. To better grasp the object, the path of end effector was planned here. Also, a circuit for the loading and unloading was designed.

All in all, requirements of the project were finished with our self-developed smart robot. The robot can overcome all the tasks in the competition with good performance. However, the robustness and stability of our robot is noy high enough, we may improve the stability in future.

Keywords: Mecanum wheels; OpenMV module; PID controller; learning-based method with neural network; Forward and inverse kinematics; Path planning;

Content

Introduction | 1 3
Concept Design | 2 3
2.1 Problem Definition 3
2.2 Concept Generation and Selection 4
2.2.1 Path Tracking 4
2.2.2 Obstacle Avoidance 5
2.2.3 Cargo Detection and Classification 5
2.2.4 Taking Cargo 5
2.2.5 Loading and Unloading 6
2.3 General Mechanism 7
2.4 Material Selection 7
2.4.1 End Effector 7
2.4.2 Manipulator 7
2.4.3 Driving Components & Supports 8
2.5 CAD Designed Works 8
Manufacturing | 3 8
3.1 Obstacle Avoidance 8
3.1.1 Principle for Movement 8
3.1.2 Method for Obstacle Avoidance 9
3.1.3 Speed Control 10
3.2 Tracking 11
3.2.1 Circuit 11
3.2.2 Control Logic 12
3.2.3 Position of Camera 14
3.2.4 Difficulties Encountered 15
3.3 Loading & Unloading 16
3.3.1 Robotic Manipulator subsystem 16
3.3.2 Object Detection Subsystem 19
3.3.3 Unloading Subsystem 22
3.3.4 Subsystems Collaborations 23
Cost Estimation | 4 24
Conclusion | 5 25
Cited References | 6 24
Nomenclature | 7 27
Acknowledgements | 8 28

Introduction | 1

Nowadays, Automatic Guided Vehicles (AGV) are widely used in both the daily life and production. We may see many usages for the AGVs including sorting in logistics, cargo delivery or even the regional antivirus in some places. Meituan has developed a kind of AGV that can bring takeaways to the customers, which we can partly discover the potential of AGVs. As students who major in robotics engineering, we should get to be familiar with AGVs and try to build by ourselves. In this project we got the opportunity.

Figure 1: AGV for takeaway developed by Meituan

In this project, we aim to build our own AGV that can complete the requirements of project competition. What we want to do is to try not to buy any highly mature products related to our project. Also, to overcome the requirements, we set the goals as below:

- Design a proper structure for cargo picking and delivery
- Choose and make good uses of proper sensors
- Use proper method to find the classification and positions of cargo
- Design good algorithm to avoid obstacles and track path
- Fast and smooth control of speed
- Intelligent and highly adaptive to the environment

Concept Design | 2

2.1 Problem Definition

In this project, we are required to build an autonomous cargo delivery robot that can complete the following tasks: obstacles avoidance, path tracking and deliver different cargos including orange, eraser and even a bottle of water and pen in advanced requirement. Tasks are

set in different regions in the map shows in Figure 2 sequentially, we may consider to cover the tasks according to the configuration in the map.

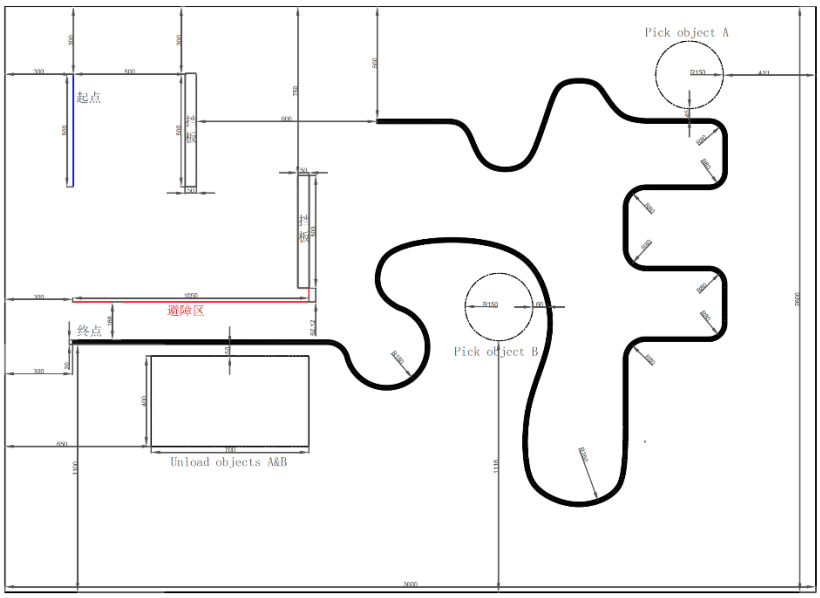


Figure 2: Map for the competition

Then we divide the main tasks are obstacles avoidance, path tracking and cargo delivery. The problem we face can be defined as follow based on the main tasks:

- Path tracking
- Obstacle avoidance
- Cargo detection and classification
- Taking cargo
- Loading and unloading

We try to come up with proper methodologies to overcome the requirements in the next section.

2.2 Concept Generation and Selection

2.2.1 Path Tracking

The AGV should be able to follow the black line on the ground, and try to ensure that the center of the AGV do not deviate from the black line too much, and maximize the speed while ensuring the accuracy of line tracking. Therefore, the ability to recognize black line is the main part of this function.

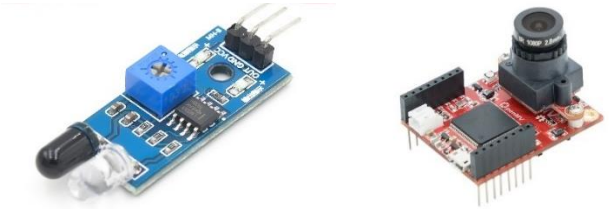


Figure 3: Infrared Sensor (Left) and Visual Sensor (Right)

We considered two schemes of infrared sensor and visual sensor. For infrared sensor, the advantages are simple to use, low cost, fast response speed, and simple calculation process, the disadvantage is the low resolution of black line position. For the visual sensor, and the advantage is the high resolution of the black line position, and the disadvantages are the high cost, the large amount of calculation required, and the slightly slower response speed.

In the trade-off between accuracy and speed, we hope the AGV has better accuracy, therefore the visual sensor is selected to be the solution.

2.2.2 Obstacle Avoidance

The AGV must be able to recognize the existence of surrounding obstacles, change the moving direction according to the position of obstacles. Therefore, the ability to recognize obstacles around the AGV is the core of this function.

Figure 4: Laser Ranging Sensor (Left) and Ultrasonic Ranging Sensor (Right)

We considered two schemes of laser ranging sensor and ultrasonic ranging sensor. For the laser ranging sensor, the advantages are high accuracy and fast response speed, and the disadvantage is the susceptible performance due to the ambient light conditions. For ultrasonic sensors, the advantage is low cost, and the disadvantages are lower accuracy and longer response time for long distance ranging.

Considering the small space of our obstacle avoidance task, and the cost of use and the possible influence of ambient light, the ultrasonic ranging sensor is selected to be the solution.

2.2.3 Cargo Detection and Classification

The AGV should be able to recognize the existence of the cargo, and determine its type and position relative to the AGV. The identification should be as fast and stable as possible, and the type of the cargo should be determined with high accuracy, and the position of the cargo relative to the trolley should be calculated. Therefore, the ability to identify cargo is the core of this function.

Considering the possible arithmetic operation of such classification and position calculation, the visual sensor is selected since its chip is design for these operations.

2.2.4 Taking Cargo

The AGV should be able to pick up cargo of different shapes, weights and materials according to the position of the cargo, and place it on the AGV. Therefore, the ability to grasp the cargo is the core of this function.

There are two parts of the grasping, one is the robotic arm and the other is the end effector.

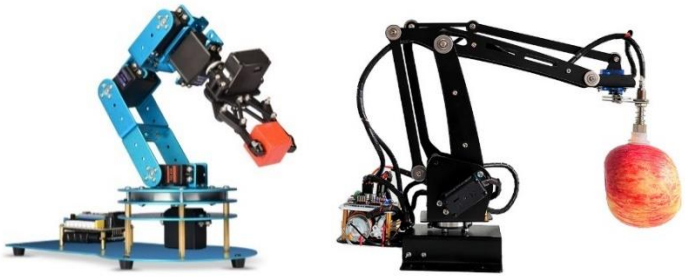


Figure 5: 6-axis Tandem Robotic Arm (Left) and Palletizing Robot (Right)

For the robotic arm, we considered a 6-axis tandem robotic arm and a palletizing robot. Since the palletizing robot has more mature industrial applications and can always maintain the pose of the end effector, it is more in line with our vision, so we chose the palletizing robot as the robotic arm structure.

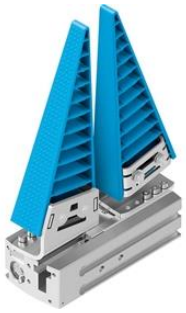


Figure 6: Fin Gripper

For the end effector, we believe that the fin gripper is adaptive to different shapes of objects and is able to hold the cargo firmly. Therefore, the fin gripper is selected to be the end effector.

2.2.5 Loading and Unloading

The AGV should be able to place and unload the cargo.

We considered two schemes, the first is to use the robotic arm to re-place the object from the AGV back to the ground, and the second is to use a cargo “compartment” to directly dump the cargo down the AGV.

Considering the time-consuming and complexity, we suppose the cargo “compartment” is faster and more convenient, so we choose to design a cargo unloading mechanism which is also able to place the cargo on it.

2.3 General Mechanism

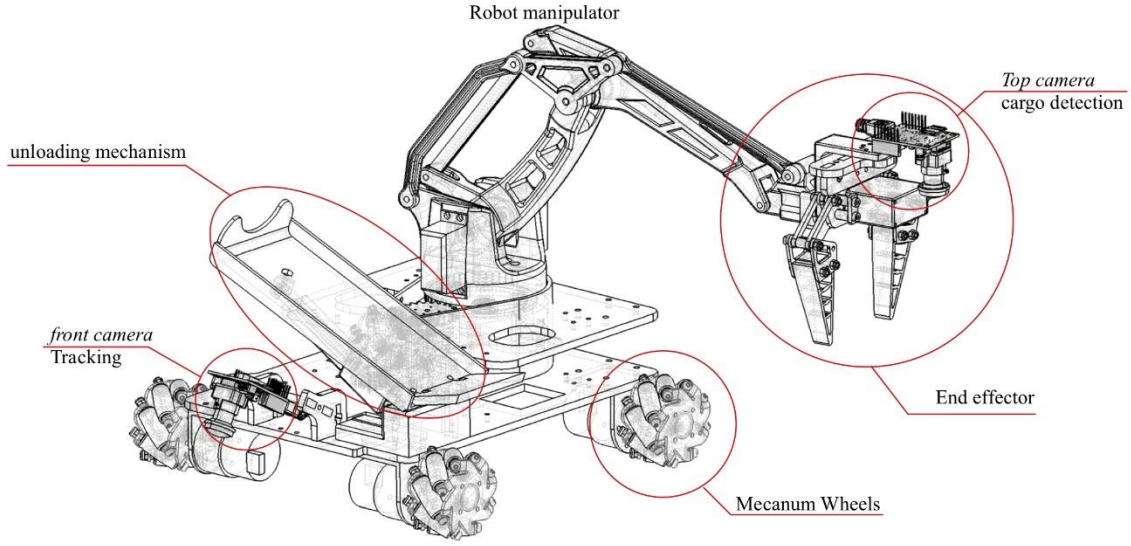


Figure 7: General Mechanism of the Design

2.4 Material Selection

2.4.1 End Effector

In order to enhance the robustness of the grasping behavior, we choose soft and self-adaptive manipulator based on the Fin ray effect®, which can adapt to different object shapes, and achieve higher grasping stability when there is certain error that cannot be eliminated. For this reason, the soft fingers of the end effector are made of soft plastic and manufactured by injection molding, which ensures high manufacturing precision and satisfying surface finish.

2.4.2 Manipulator

Since the actuators of the manipulator are much heavier, the use of a conventional chain shaped manipulator will lead to a weak dynamic performance. For this reason, the mechanical structure of our manipulator refers to a more mature palletizing robot in the industry, which consists of a series of rigid link, and most of the actuators are set at the base platform of the manipulator, which reduces the Inertial matrix in the manipulator dynamics. Considering the difficulty of obtaining manufacturing materials and related equipment, and considering the effect to the environment due to the manufacturing process, we selected the PLA-based Fused Deposition Modeling (FDM) process, the processed parts have certain toughness and are biodegradable.

2.4.3 Driving Components & Supports

For some driving parts and supports, their assembly requires high machining accuracy to ensure that there will be no jittering or breaking during operation. Considering the cost, we choose polymethyl methacrylate material and use the laser cutting process to manufacture base plate, gears and other power transfer components.

2.5 CAD Designed Works

In order to improve design efficiency, most prototype components are pre-assembled in SolidWorks to verify the rationality of hardware design, since the overall prototype contains many parts (about 600 parts).

At the same time, in order to reduce the difficulty of robotic manipulator debugging, the manipulator is modeled and simulated with the help of Matlab Robotic Toolbox, and the rationality of the robotic manipulator structure and inverse kinematics is verified in the simulation.

Manufacturing | 3

In section 2, we discussed what capabilities our robots should have and how we generated the concepts and selected proper methodologies. In this section, we will discuss how we implemented the methodologies we chose, the problems we encountered and how we overcame the problems. Since hardware, software and control methods are highly bind together in a certain function, we will not separate these three parts but discuss each part with the order of main tasks in this project.

3.1 Obstacle Avoidance

3.1.1 Principle for Movement

Mecanum wheel is a kind of omni-directional wheel which can be used to achieve omnidirectional movement with curtain configuration. The principle is to change the velocity of the rotational axis to the rollers on the wheel, which will generate oblique velocity. Since the Mecanum wheels are usually used in a set of 4 wheels, by proper configuration, the 4 oblique velocities can generate velocity of different directions on the center.

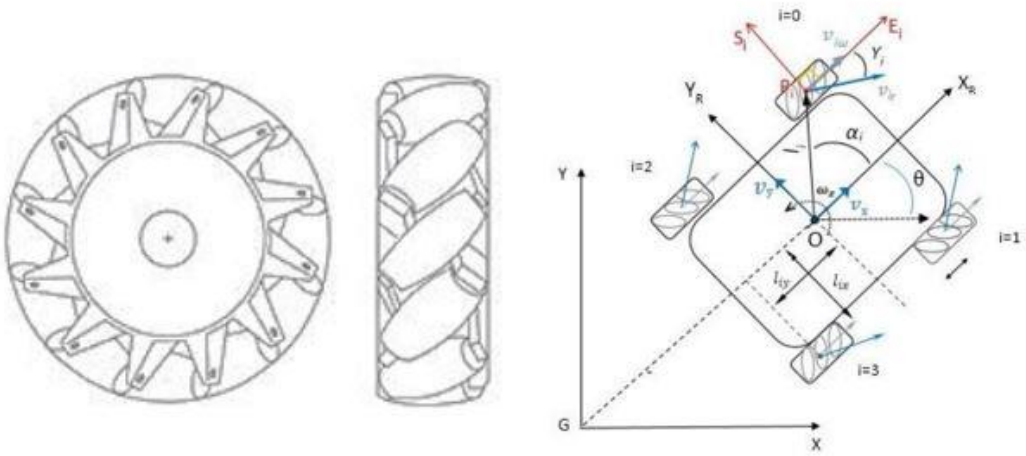


Figure 8: Mecanum wheel and coordinate configuration

Denoted the velocity of the center is $v = [v_x, v_y, \omega_z]$, the angular velocities of four wheels are $\omega = [\omega_1, \omega_2, \omega_3, \omega_4]$, with kinematics analysis, we can have the following relation: [1]

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ \omega_4\end{bmatrix} = \frac{1}{r}\begin{bmatrix}1 & -1 & -(l_x+l_y)\\ 1 & 1 & (l_x+l_y)\\ 1 & 1 & -(l_x+l_y)\\ 1 & -1 & (l_x+l_y)\end{bmatrix}\begin{bmatrix}v_x\\ v_y\\ \omega_z\end{bmatrix}.$$

Where l_x and l_y are half the length of the robot at x and y directions. Also:

$$\begin{bmatrix}v_x\\ v_y\\ \omega_z\end{bmatrix} = \frac{r}{4}\begin{bmatrix}1 & 1 & 1 & 1\\ -1 & 1 & 1 & -1\\ -\frac{1}{(l_x+l_y)} & \frac{1}{(l_x+l_y)} & -\frac{1}{(l_x+l_y)} & \frac{1}{(l_x+l_y)}\end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ \omega_4\end{bmatrix}$$

Then we can set proper speeds of the wheels and make the robot move in wanted direction.

3.1.2 Method for Obstacle Avoidance

Based on the principle discussed in the previous section, we can use the properties of Mecanum wheels and define the basic mode for movement. In the task of obstacle avoidance, the pure translation with moving directions of forward, left and right are defined. To avoid the obstacles with a certain distance, we set ultrasonic sensors at corresponding position to detect the distances between the obstacles at certain directions.

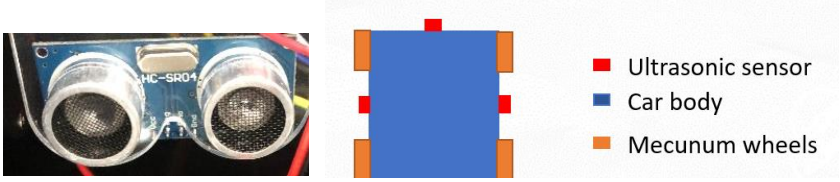


Figure 9: Ultrasonic sensor and configuration of sensors

The algorithm is to change the moving direction when the sensor detects a distance between the obstacle lower than certain threshold. Since the obstacles are set in a fixed way in the map, we define the direction sequence as forward, right, forward and left, then the robot car can avoid the obstacles easily. The working flow can be shown in the Figure 10.

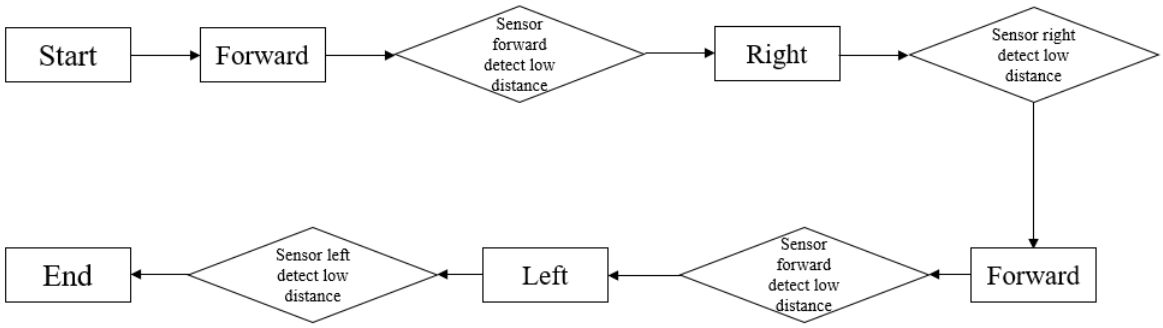


Figure 10: Workflow of obstacle avoidance

3.1.3 Speed Control

As we discussed before, a method for the robot car to avoid the obstacles has been determined: the robot can move in only forward, left and right directions with pure translation by specific sequence. Also, due to the property of Mecanum wheel, the robot can move without rotation only when the speeds of four wheels are equal in magnitude.

However, the speed control can be difficult with only open loop control here.

Considering the requirement of smooth speed control, a feedback control system is built. Encoders are device that can measure the speed of the wheels, which can generate suitable feedback signals in our project. Moreover, a PD controller is designed in the control loop for adjust the error.

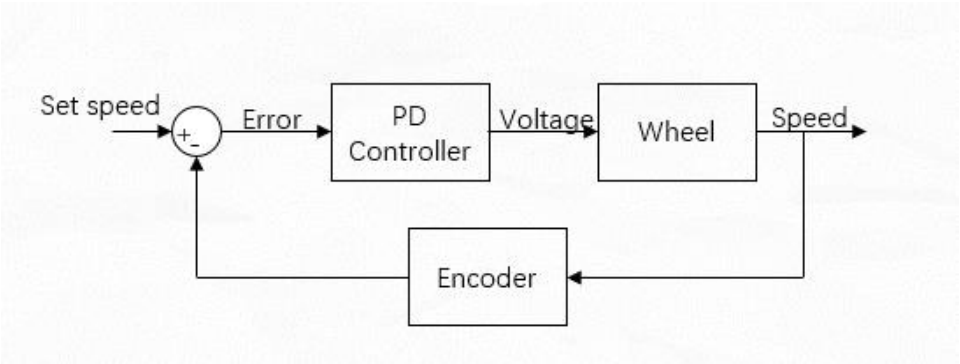


Figure 11: Close loop control system for speed.

The control system can be established as the block diagram in Figure 11. The goal for the system is to adjust the system output to have fast response and lower overshoot since the speed controller will take sample and make adjustment evert 0.1 second as design.

To achieve the goal, the parameters of the PD controller should be tuned properly. The gains for the PD controller were tuned by the method of trial and error, different sets of parameters were used in the controller and the controller was fed with a step input of desired speed, the responses curves could be obtained as follow.

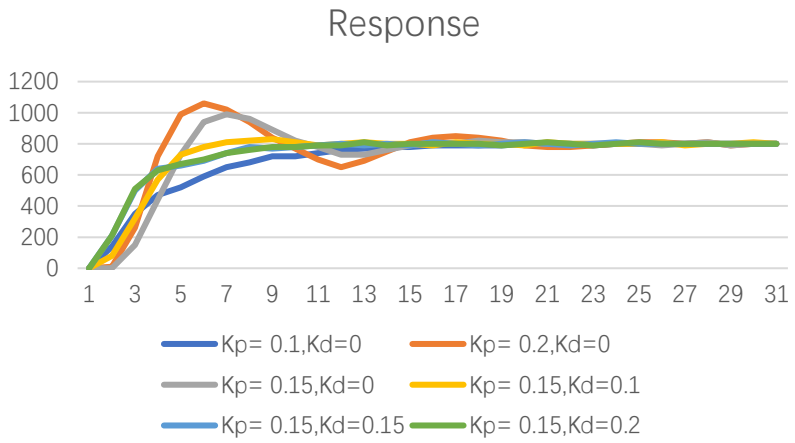


Figure 12: Response curve for PD parameter turning.

In the generated curve, since the goal is to have fast response and less overshoot, a set of $Kp = 0.15$ and $Kd = 0.15$ can be a good tradeoff for the goal. Then the demand of smooth control of speed can be obtained.

3.2 Tracking

3.2.1 Circuit

Firstly, we introduce the construction of the tracking part circuit. As shown in Figure 13:

• OpenMV module is supplied with 5V, two signal lines (TX&RX) are connected with stm32.

• Two L298n driver modules are supplied with 12V, one of them supplies 5V to MCU.

• Four signal lines (bule ones) are connected between stm32 and L298n, through which the MCU provides PWM to the driver modules.

• One L298n controls the front motors, the other controls the rear motors. Each pair of motors is divided into left and right motor.

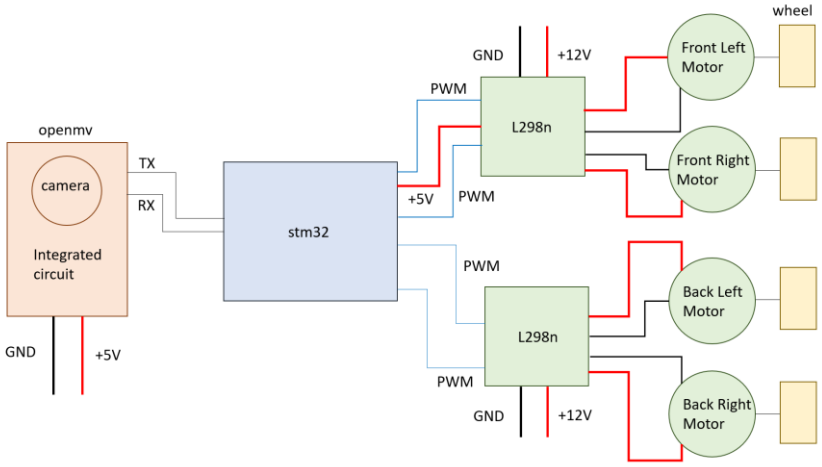


Figure 13: Circuit of tracking part

3.2.2 Control Logic

The control logic of tracking is divided into 8 steps, we will elaborate on each step in detail.

• *Step1-Collect images*: OpenMV module integrates camera and a chip. We use the camera on the module to collect the road condition information of the vehicle. (Figure 14-1)

• *Step2-Binarization*: We use Python to program OpenMV module. Because the original image has a lot of noise, we set a threshold and use it to do image binarization. (Figure 14-2)

• *Step3-Fit the route*: Use linear regression method provided by the module developer to fit the route ahead of the vehicle. (Figure 14-3)

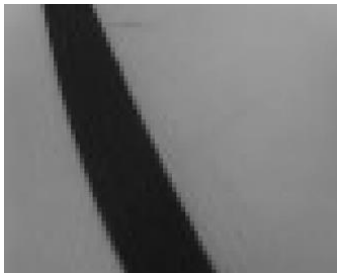


Figure 14-1

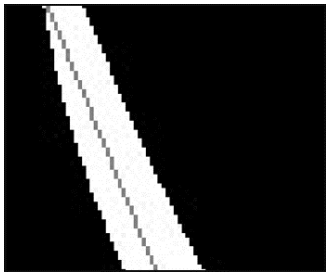


Figure 14-2

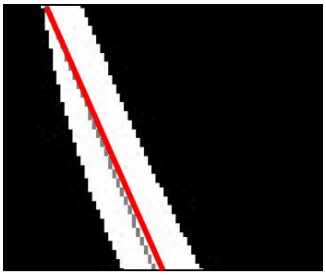


Figure 14-3

Figure 14: Step1-3

• *Step4-Calculate error:* The error of the vehicle has two parts: position error ρ and direction error θ. We define the position error as the abscissa of the intersection point between the route and the x-axis of pixel coordinate system, and the direction error as the angle between the route and the central axis of pixel coordinate system.

For position error ρ: As shown in Figure 15, the intersection point of the central axis (blue line) and the x-axis of the pixel coordinate system is taken as the origin, ρ is negative if the intersection point is to the left of the origin. Otherwise, it is positive.

For direction error θ: The difference between 90 degrees and the actual angle of the route is the error of direction.

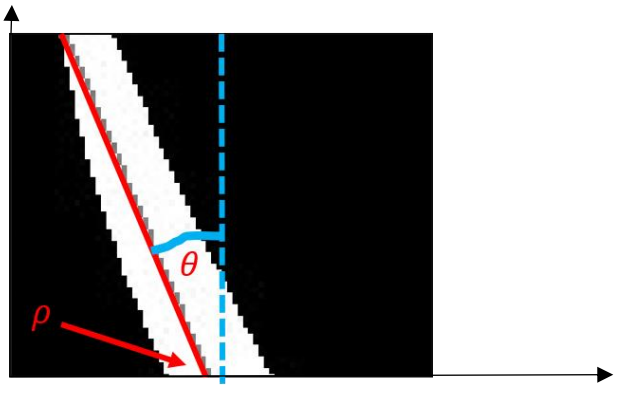


Figure 15: Position error and direction error

• *Step5-Use PID controller to get control value*: PID controller has been taught in two specialized courses, so it will not be covered here. We design two PID controllers to calculate the control value of position error and the direction error respectively. Input the errors obtained in step 4 into the controllers, and sum up the output value, then get the appropriate control value.

$$C_p = K_p \times \rho_{err} + \int_0^t K_i \times \rho_{err}\, dt + K_d \times \frac{d\rho_{err}}{dt}$$

$$C_d = K_p \times \theta_{err} + \int_0^t K_i \times \theta_{err}\, dt + K_d \times \frac{d\theta_{err}}{dt}$$

$$C = C_p + C_d$$

• *Step6-Voltage duty ratio*: In order to make the difference between the two wheels of the car, so that it can turn, we calculate voltage duty ratio of left motors and right motors as follows:

$$VDR_L = base - C$$
$$VDR_R = base + C$$
$$base = 400$$

We also set a basic value $base$, which determines the motor speed. The larger the value of $base$, the slower the speed of motors. Control method of McEnham wheel is explained in part x.

• *Step7- pass value from OpenMV to stm32*: After calculating a pair of 3-bit voltage duty ratios, OpenMV sends the transmission flag through *usart*, and then sends it to stm32 bit by bit. After stm32 receives the transmission flag through *usart2*, it converts the continuously received 6 digits into two three-digit numbers.

• *Step8: stm32 generates PWM to control L298n:* After receiving two values, stm32 judge current situation belongs to going straight, turning left, or turning right according to the voltage duty ratios. In order to improve the stability of tracking, the vehicle should go straight when the difference between the two voltage duty ratios is less than 14.

Therefore, when $L_{PWM} - R_{PWM} > 14$, the left wheel speed is higher than the right wheel speed, the car needs to turn right. Similarly, when $R_{PWM} - L_{PWM} > 14$, the car needs to turn left. Moreover, only when McEnham's motion requires all wheels to rotate at the same speed, the center of rotation is in the center of the car. Therefore, unlike regular wheels, we still need to assign the same PWM value to four wheels when the car is turning left or right.

Based on voltage duty ratios, stm32 uses TIM8 to generate the corresponding 4 PWM waves and inputs them into the B and A phases of a pair of L298n respectively. Thereby, it controls a pair of voltages output of each L298n, then controlling the speed of each wheel.

Apart from controlling speed of wheels, stm32 also needs to control the direction of each wheel. Since the directions of each wheel when car is turning left, turning right or going straight are different (as shown in Table 1), each wheel needs a pair of GPIO ports to control the direction of it. (Table 2)

Table 1: The state of each wheel in different situations

	Forward	Clockwise (turn right)	Counterclockwise (turn left)
Left forward wheel	Forward	Backward	Forward
Right forward wheel	Forward	Forward	Backward
Left back wheel	Forward	Backward	Forward
Right back wheel	Forward	Forward	Backward

Table 2: use front left wheel as an example

	Forward	Backward	Stop
PC 0	0	1	1
PC 1	1	0	1

In short, stm32 decides to turn left, turn right or go straight according to a pair of pwm values given by OpenMV. In each situation, stm32 controls the steering and speed of each wheel.

The schematic diagram of the tracking part is shown in Figure 16.

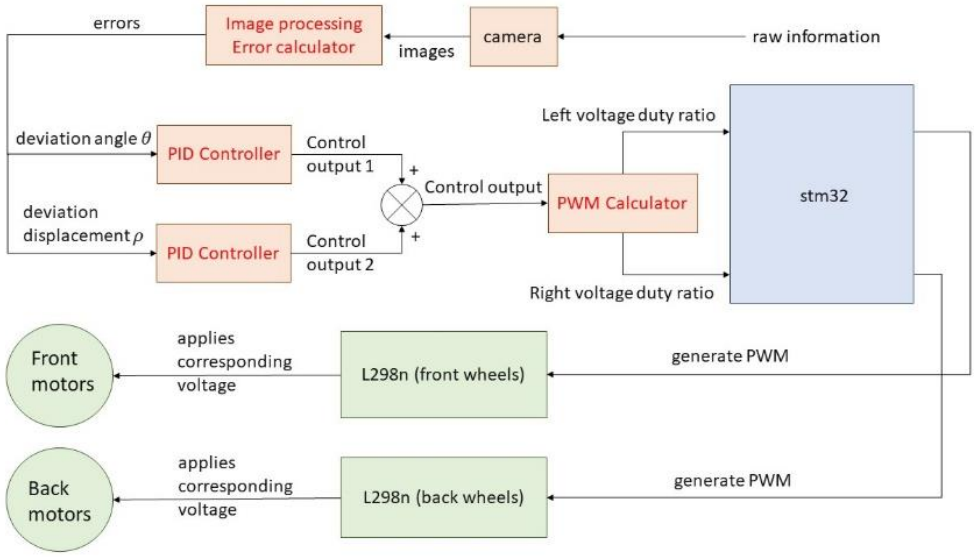


Figure 16: Schematic diagram of the tracking part

3.2.3 Position of Camera

The position of OpenMV can be determined by the four variables: x, y, z, and angle (α) between OpenMV and the horizontal plane.

According to the introduction of McNamulen's movement method above, McNamulen rotates around the center of the car. So, during the car tracking process, it is necessary to ensure that the black line is as close as possible to the center line of the chassis of the car. Therefore, the point that the camera was projected on the horizontal plane must coincide with the midpoint of the line connecting two wheels of the car. That is, the x, y position of the OpenMV coordinate is determined.

After determining the x, y of OpenMV, we still need to determine the z of the camera, and the angle of inclination (α). The angle (α) between OpenMV and the horizontal plane determines the proportion of the tracking route at the far end in length the camera's field of view. When the angle (α) increases, the car gains more foresight (as shown in figure 17). The height (z) between OpenMV and the ground determines the width of the tracking black line in the camera, that is, the proportion of the width of lines in width of camera's field of view. When the height (z) increases, the car is more sensitive to the angle of lines (as shown in figure 18). Since these two variables can only be adjusted through experience and may require constant adjustment, we opted for a flexible and adjustable elastic hose to fix the camera (as shown in figure 19).

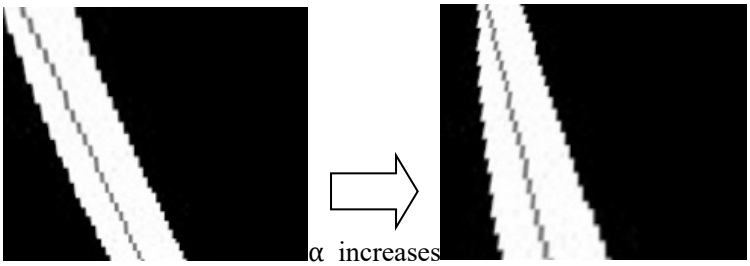


Figure 17: increasing the angle between camera and the horizontal plane, the tracking line at far end is shown more in the camera field

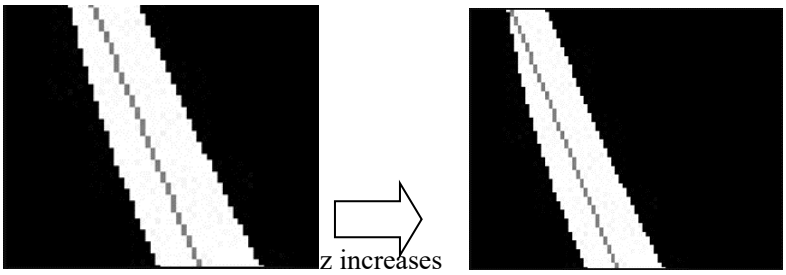


Figure 18: increasing the distance between camera and the horizontal plane, the line is thinner in the camera field

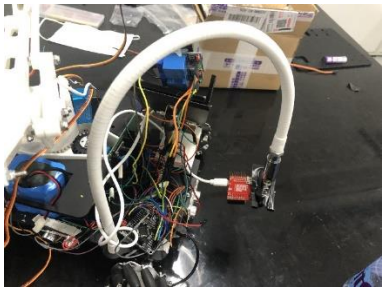


Figure 19: the flexible and adjustable elastic hose we use

3.2.4 Difficulties Encountered

Under normal circumstances, the camera should not be too high or tilting too much, otherwise the lines far from the lines at far end will be seen too much, which will affect the fitting effect. However, sharp turns in the map (as shown in the figure 20) require the camera to have more foresight, that is, the camera cannot be too low, otherwise it will go to a certain position and cannot find the line to turn right.

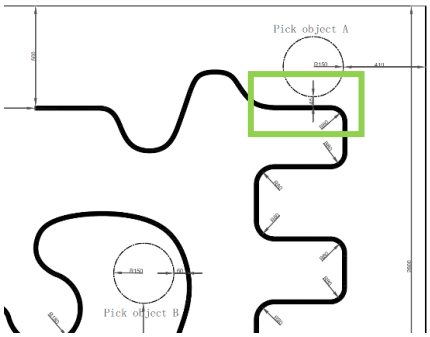


Figure 20: the sharp turn in the map

Since the camera angle needs to be adaptable to various types of curves in the map, the demand of two different situation becomes a trade-off. To sort out this problem, we end up adding an extra case. When the length of the line in the OpenMV field of view is too small, let the car rotate quickly to find the black line in the field of vision again. But this method has great uncertainty.

3.3 Loading & Unloading

The actions of grabbing, loading and unloading are mainly completed by three subsystems, and their configurations are as follows.

3.3.1 Robotic Manipulator subsystem

In order to achieve the grasping goal of the project, we need to design a robotic manipulator subsystem, which must achieve object localization, grasping and loading. For this purpose, the hardware system and auxiliary software of the robotic arm are designed.

This system includes a proportional scaling manipulator (1:7) based on ABB IRB460 palletizing robot, an STM32 RCT6, the corresponding manipulator debugging program and manipulator control program. The system also includes a simulation program based on Matlab Robotic Toolbox.

This system aims to realize the grasping subsystem based on machine vision, and integrates functions including Cartesian coordinate transmission, inverse kinematics solver and motor control. The system can solve the inverse kinematics (joint space) of the manipulator according to the transmitted Cartesian coordinates of target point, and control the end effector to move to the destination.

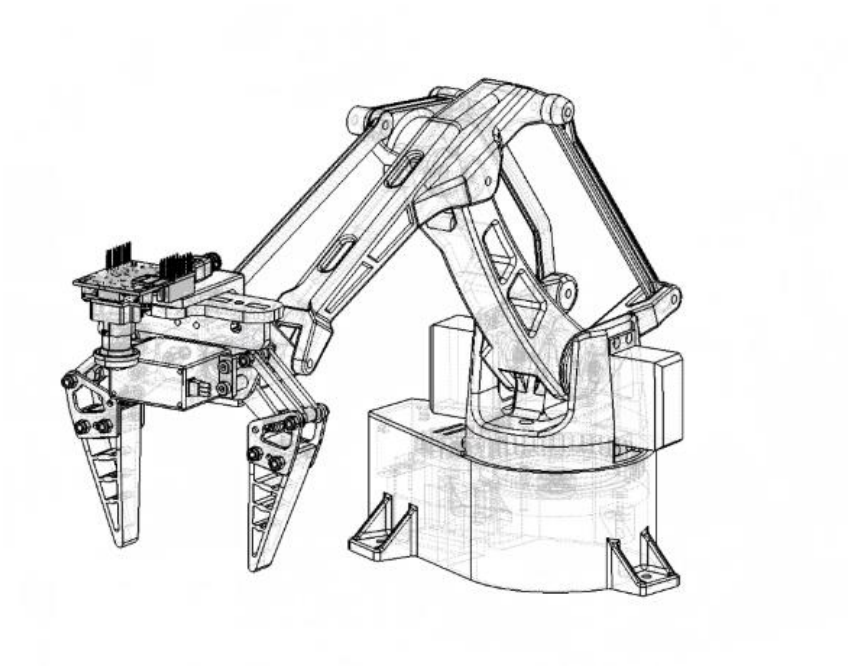


Figure 21: Manipulator overview

3.3.1.1 Mechanical Structure of the Robot Manipulator

The robotic manipulator has a total of 4 degrees of freedom. By simplification, it can be regarded as a standard series linked robot. The novel structure ensures that the z-axis of the end effector frame is parallel or coincident with the z-axis of the base frame. The simplified mechanism diagram is shown below:

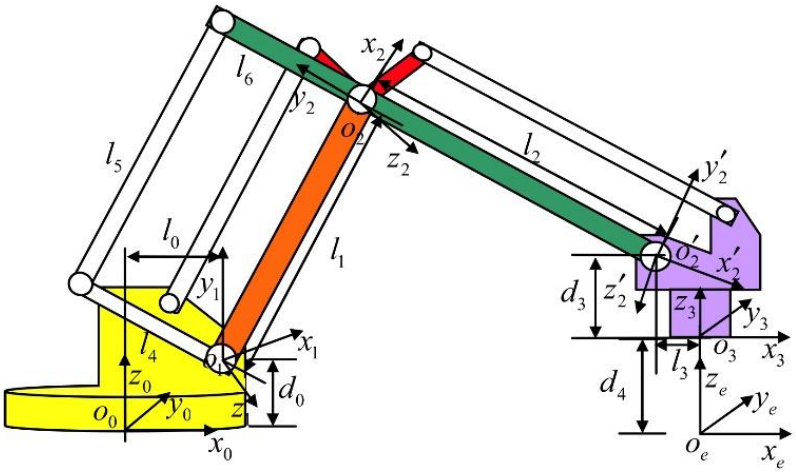


Figure 22: Manipulator diagram (simplified)

The manipulator has a total of 7 critical parameters (non-joint space variables), which are used to fully determines the kinematic characteristics of the manipulator (7. Nomenclature).

3.3.1.2 Kinematic Analysis of the Robot Manipulator

By using the standard D-H parameters, the frames of each linkages can be obtained to describe the corresponding linkage orientations. An homogeneous transformation matrix from base frame to end effector frame is obtained.

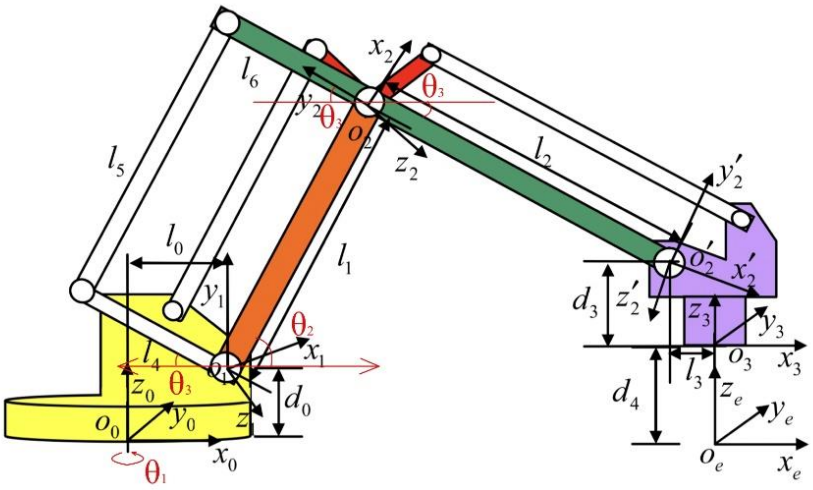


Figure 23: Manipulator frame system

$$T_e^0 = T_1^0 \cdot T_2^1 \cdot T_3^2 \cdot T_e^3$$

$$T_e^0 = \begin{bmatrix} c_{1+4} & -s_{1+4} & 0 & c_1(l_3 + l_2c_{2+3} + l_1c_2 + l_1) \\ s_{1+4} & c_{1+4} & 0 & s_1(l_3 + l_2c_{2+3} + l_1c_2 + l_1) \\ 0 & 0 & 1 & -d_4 - d_3 + l_2s_{2+3} + l_1s_2 + d_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The Cartesian coordinates of the end effector are given based on $\{O_0\}$, where the origin of $\{O_0\}$ is the intersection of the axis of the manipulator shoulder shaft and the ground, and the positive x-axis and y-axis are established based on the following figure.

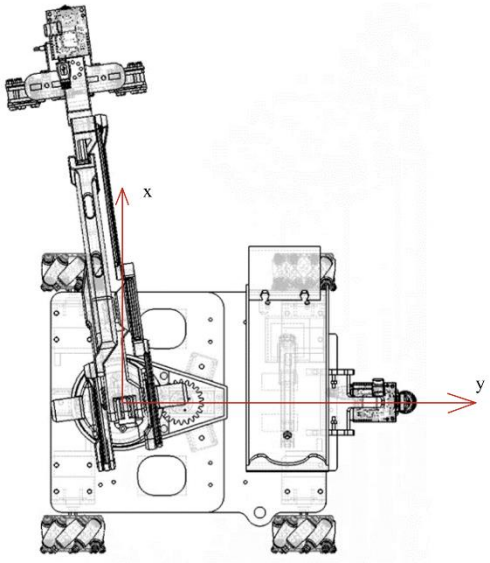


Figure 24: Base frame configuration

Make the position of the target point equal to the position matrix of the homogeneous transformation matrix

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1(l_3 + l_2c_{2+3} + l_1c_2 + l_1) \\ s_1(l_3 + l_2c_{2+3} + l_1c_2 + l_1) \\ -d_4 - d_3 + l_2s_{2+3} + l_1s_2 + d_0 \end{bmatrix}$$

Solving the inverse kinematics analytical solution can be obtained as:

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{y}{x}\right) \\ 2 \times \tan^{-1}\left(\frac{d + \sqrt{c^2 + d^2 - g^2}}{c - g}\right) \\ 2 \times \tan^{-1}\left(\frac{f + \sqrt{f^2 + e^2 - h^2}}{e - h}\right) - \theta_2 \end{bmatrix}$$

Where:

$a = \sqrt{x^2 + y^2} - l_0 - l_1,\ b = z - d_0 + d_3 + d_4,\ c = 2 \times a \times l_1,\ d = 2 \times b \times l_1,\ e = 2 \times a \times l_2,\ f = 2 \times b \times l_2,\ g = -a^2 - b^2 - l_1^2 + l_2^2,\ h = -a^2 - b^2 + l_1^2 - l_2^2.$

3.3.1.3 Simulation

The simulation includes inverse kinematics solution and forward kinematics verification, the results are as follows (random Cartesian coordinates)

```
>> servo = servoOut(200, 200, 250)

servo =

   45.0000   55.8079  -19.8420

>> ArmSimulation
  200.0000

  200.0000

  250.0000
```

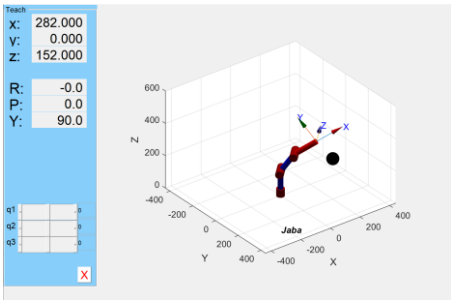


Figure 25: Matlab simulations

3.3.1.4 Robot Manipulator Control

Servo Control & Data Protocol

The motor is controlled by pulse width modulation (PWM), and the pulse width determines the rotation angle of the motor. In order to achieve a certain control accuracy, so that the control can have a resolution of at least 1 degree, the following configuration is used:
PWM frequency 200Hz (servo pulse width 500us ~ 2500us), control accuracy 1 deg, CCR range 45~225 mapping 0~180 deg

Pass in a 12-characters string (integer) through the serial port as the target end-effector coordinates and end-effector clamping force.
It can be seen from the base frame configuration ensures that only the y coordinate can be negative. In order to reduce the complexity of serial port data transmission protocol, the transmission of negative integer is not considered.
The first three characters are the x coordinate (mm), the 4~6th characters are the y coordinate + 300 (mm) to ensure that the y input is positive, the 7~9th characters are the z coordinate, and the 10~12th characters control the clamping force of the end effector. When this message is passed in through serial port, the program will control the PWM waveform of the corresponding pins, and the motor will rotate according to the corresponding command to move the end effector to the destination.

3.3.2 Object Detection Subsystem

For loading part, the first task is object detection. OpenMV is a frequently known solution to basic images-based tasks including classification and object detection. There are four embedded functions in OpenMV to deal with our specific task: template-based method (image.find_template), shape-based method (image.find_keypoint), learning-based method (tf.classify) and color-based detection method (image.find_blobs). All these four methods have

their own advantages and disadvantages. For example, template-based method is very sensitive to the size and dip angle of pictures taken from the camera. If the distance and angle of the object and the camera changes a little bit, this method may go wrong. The second method is good at detecting objects without the requirement of keeping the same pose, however, it is prone to be affected by the noise in the environment, especially for eraser and bottle who have a highly reflective interface. Learning-based method sounds cool, and it performs well with tons of images as training set, but we are not able to collect enough pictures and manually labelling all these images can be both time-consuming and error-prone. The very last choice we were left with is color-based one. While it looks like very simple and unstable, it works well in our task with the slack requirement that we can add marks on eraser and choose the color of pen as we want.

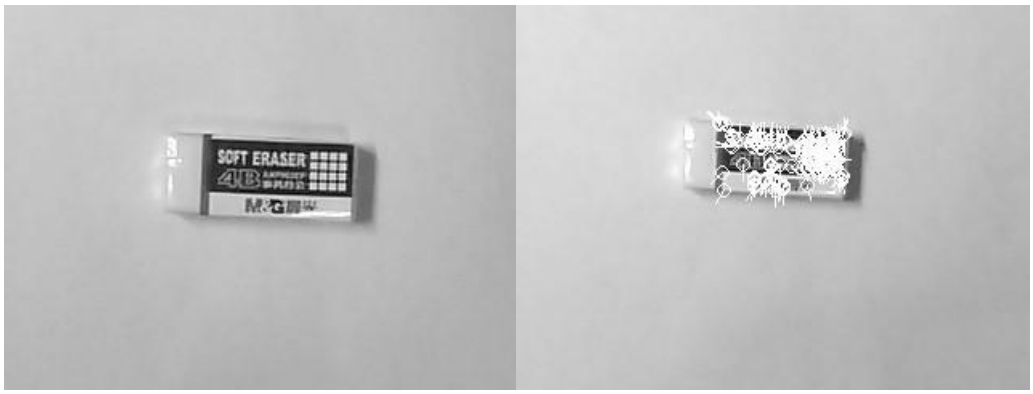


Figure 26: shape-based method (find key points)

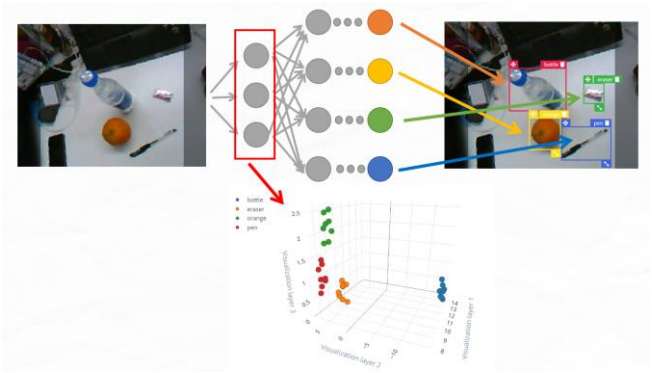


Figure 27: learning-based method (use neural networks)

Color-based method requires setting a proper threshold of different target colors. As demonstrated in the figure, we draw the bars in the threshold editor to make target color shown in white and other colors in black. As the first component L means light, we can make a loose restriction of it since the environment light varies from time to time. By taking record of colors of different objects, we check the snapshot from time to time and if target color is detected, the algorithm performs subsequent procedures.

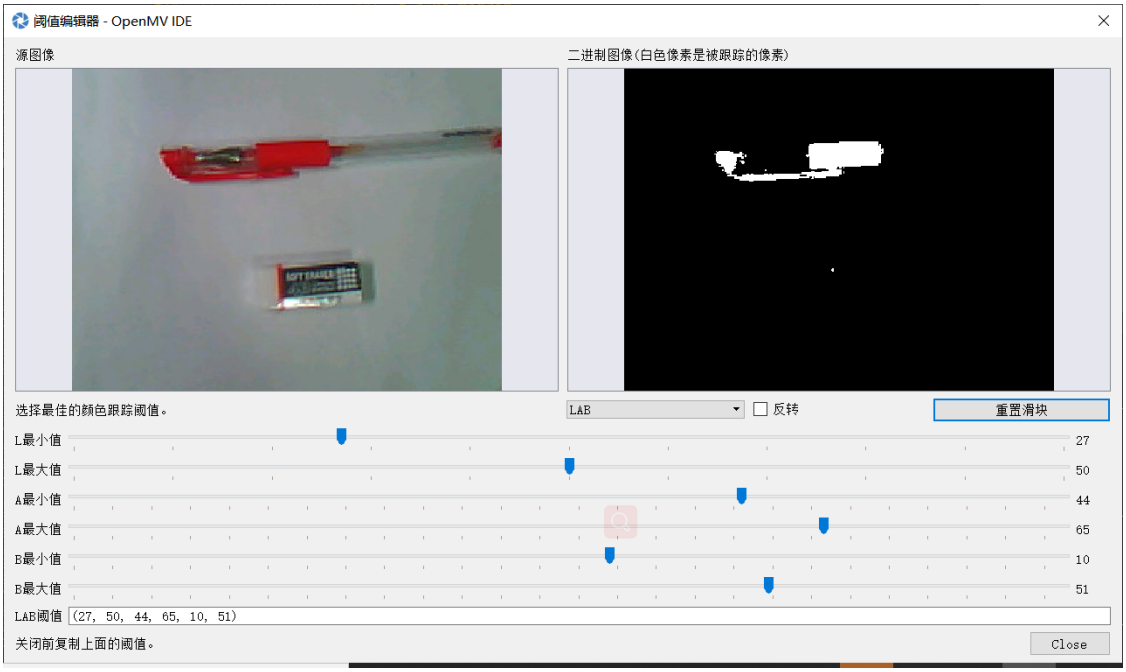


Figure 28: LAB threashoding

Next, when a snapshot meets with the first requirement, we can choose other requirements (e.g., area of target color, number of pixels, roundness, etc.) to further restrict the detection in case of noise. After all these checks, the color left with us is the target object we want to find.

Figure 29: color tracking

Finally, when the object is found and double checked, its coordinates can be acquired as well and then it will be sent to MCU for subsequent tasks, but before that, another procedure should be done, that is, transformation for different coordinate systems. To do this, a very important step is correcting the distortion of the lens. Fortunately, there is a built-in function in OpenMV to perform this job for us. Then, for the coordinate transformation, we set the fixed point in the middle of the image as it does not distort and set it as the origin of the second system. The whole procedure of image processing, object detecting and data transferring is shown below.

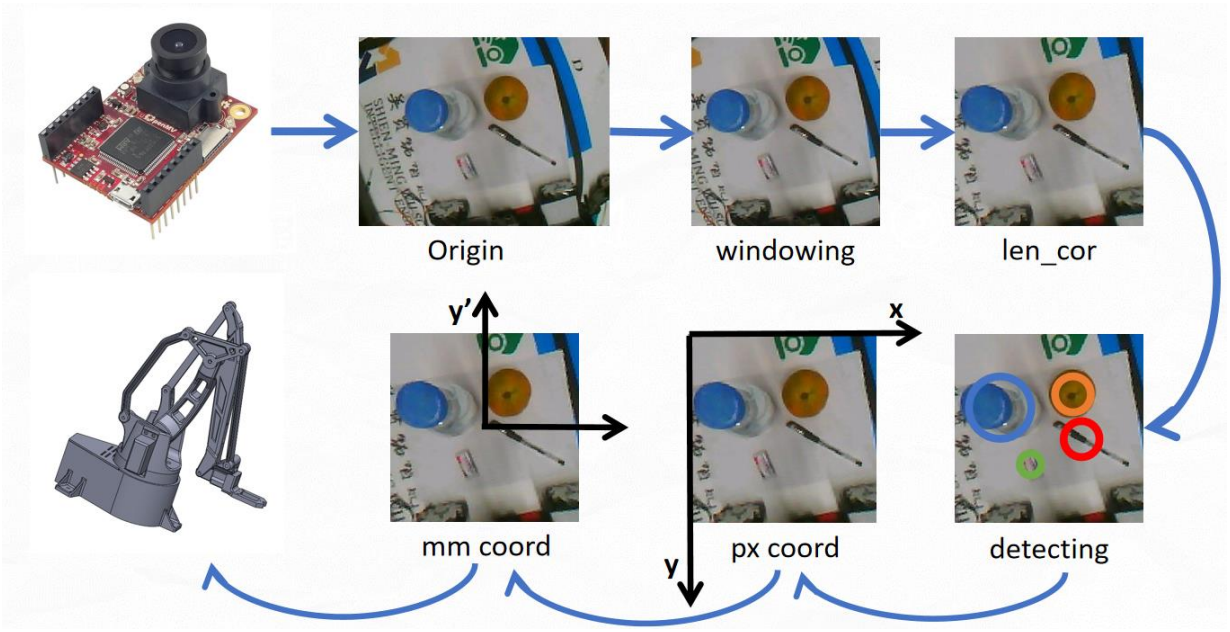


Figure 30: image processing and coordinates transformation

3.3.3 Unloading Subsystem

In order to improve the efficiency of unloading the cargo loaded on the robot, we designed a unloading mechanism driven by a linear actuator.

3.3.3.1 Mechanical Structure

When the linear actuator is in its minimum stroke, the mechanism will be in a horizontal state, waiting for the object to be loaded, when the linear actuator is in its maximum stroke, the mechanism will be in a vertical state to unload the object.

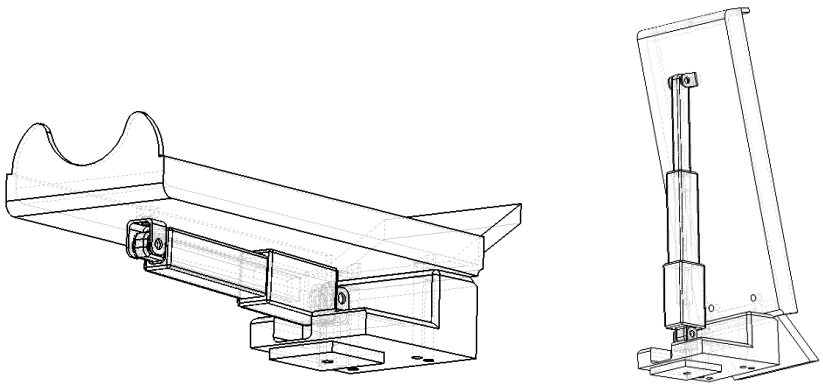


Figure 31: horizontal state and vertical state

3.3.3.1 Circuit & Control Logic

The linear actuator has two external lines, which are connected to the internal DC motor. By controlling the magnitude and direction of the voltage potential difference, the direction and speed of the linear actuator can be controlled.
Considering the limited driving capability of the MCU GPIO and the power required for the operation of the linear actuator, we use two single port double throws (SPDT) relays to indirectly control the external power peripherals (DC-motor).
Control circuit and GPIO configuration is shown (open drain & push up):

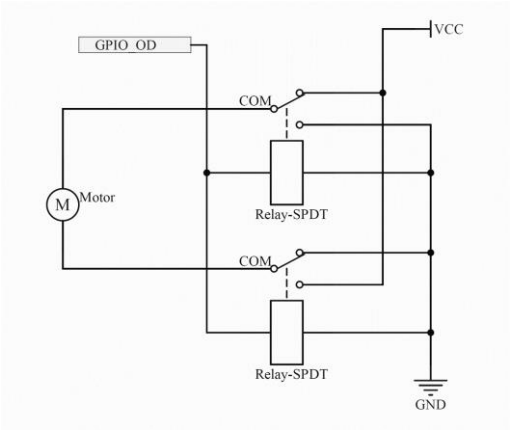


Figure 32: Control circuit and GPIO configuration

3.3.4 Subsystems Collaborations

For loading task, our camera keeps scanning the snapshots it has taken. Whenever it finds the matching object, it will send the command of stopping to the MCU, and then the car stops. After a few seconds, the camera scans again, capture the position of the target object and then send the corresponding data to the MCU through serial port. After receiving the data, MCU processes it, and control the robot arm to reach to the target point of grasping after computing the kinematic and inverse kinematic matrixes. Then, after grasping, the robot arm will move above the cargo box and unload the object. After that, all these two systems go back to their initial state and waiting for the next grasping task. Trajectory planning works during the whole procedure. Finally, when the car reaches the end of the road, the unloading mechanism dumps the cargo box to unloading area.

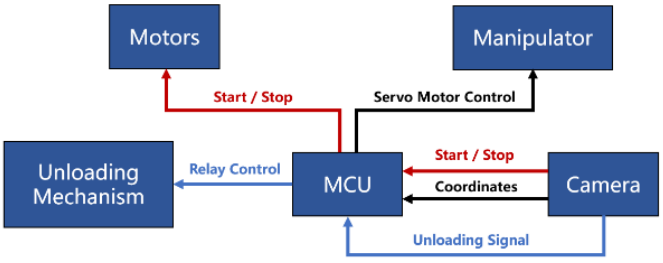


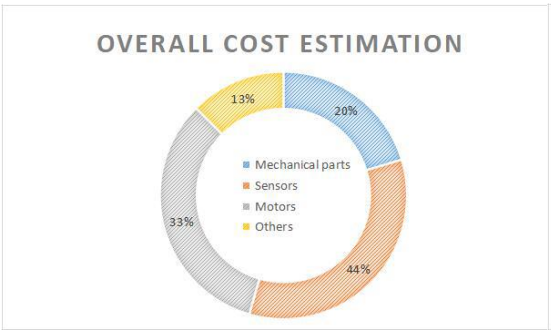
Figure 33: pipelining of subsystems operations

Cost Estimation | 4

In this project, we spend about ¥2887 altogether, which exceeds the expenditure provided by our school. Basically, there are three different parts: mechanical part, sensors, motors, and others. The mechanical part includes 3D printout (¥215), acrylic boards (¥205), fasteners (¥112.88). Sensors include ultrasonic modules (¥41.28), laser module & IMU (¥104), OpenMV & TF card (¥749.59). Motors include encoder motors (¥254), servo motors (¥462), and linear pusher module (¥150). Others include MCU (¥159), power modules (¥122.23) and others (¥46.58). Pie charts of all these three parts are shown below.

Table 3: cost in details

Subject	*Cost* (¥)
3D printout	215
Acrylic boards	205
Fasteners	112.88
Mechanical parts (altogether)	532.88
Ultrasonic module	41.28
Laser module & IMU	104
OpenMV & TF card	749.59
Sensors (altogether)	894.87
Encoder motors	254
Servo motors	462
Linear pusher module	150
Motors (altogether)	866
MCU	159
Power modules	122.23
Others	46.58
Others (altogether)	327.81



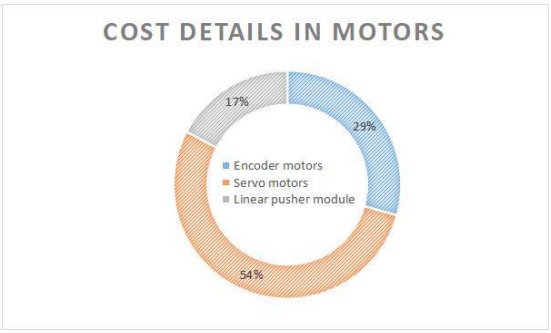


Figure 34: cost estimation

It can be seen from the charts above that motors and sensors takes the largest parts of overall expenditure, which suggests that we could replace them with cheaper ones to save money. For example, a potential action is to replace one of the line tracking OpenMV with regular infrared sensors. Also, we could alter the way that we manufactured the robot arm to save money as well. Shrinking the model size or using laser cutter is a good way to mitigate this problem.

Conclusion | 5

In this project, we built a robot car which can perform the task of obstacle avoidance, line tracking, object detecting, grasping, and unloading.

Our innovation points include:
- Following the line based on vision system
- Avoiding obstacles with feedback loop control
- PID control of speed
- Robot manipulator kinematics and inverse kinematics
- Design of unloading mechanism

Nevertheless, there are still some parts of management that we can perform better:
- Code version management
- More reasonable time schedule
- Tightened expenditure management

For technical parts, we can do better as well:
- Using omnidirectional robot arm
- Designing a better grasping mechanism for both heavy object like a bottle and thin object like a pen

Overall, we applied what we have learnt in class on this hands-on project, and we definitely made some progress during the project.

Cited References | 6

[1] Hamid Taheri, Bing Qiao and Nurallah Ghaeminezhad. Article: Kinematic Model of a Four Mecanum Wheeled Mobile Robot. International Journal of Computer Applications 113(3):6-9, March 2015.

Nomenclature | 7

The primary notations are listed in Table 4.

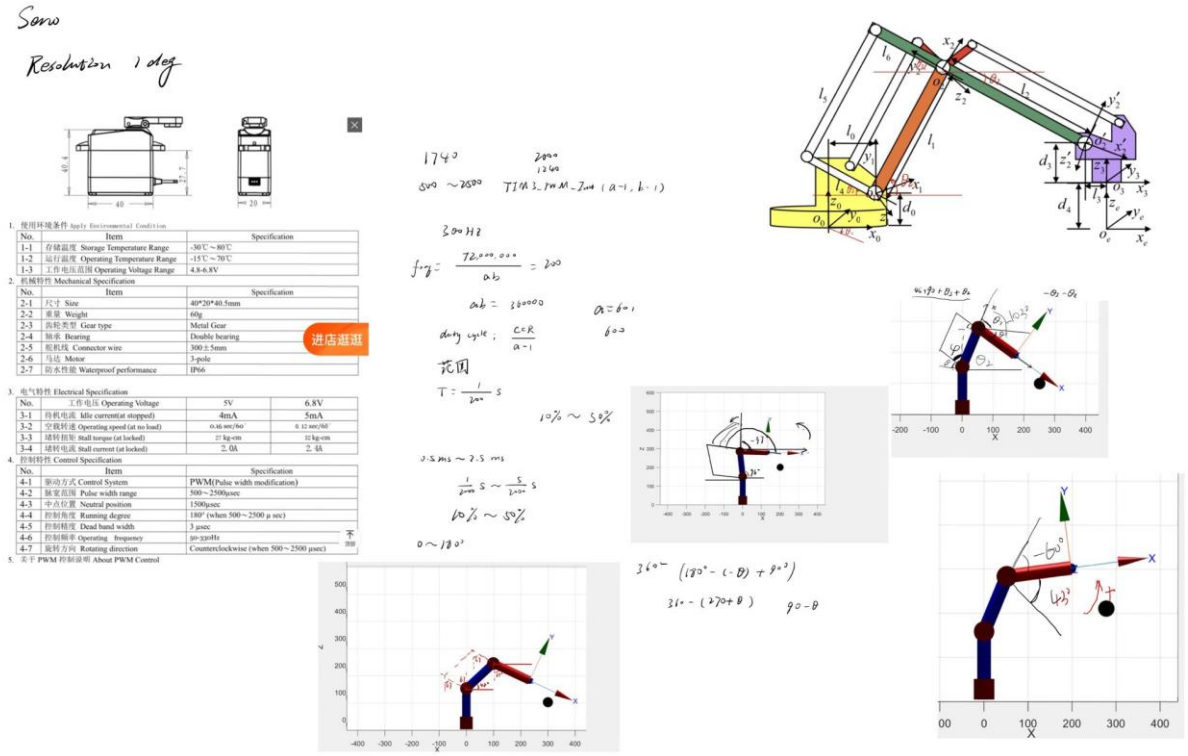
Table 4. Notations

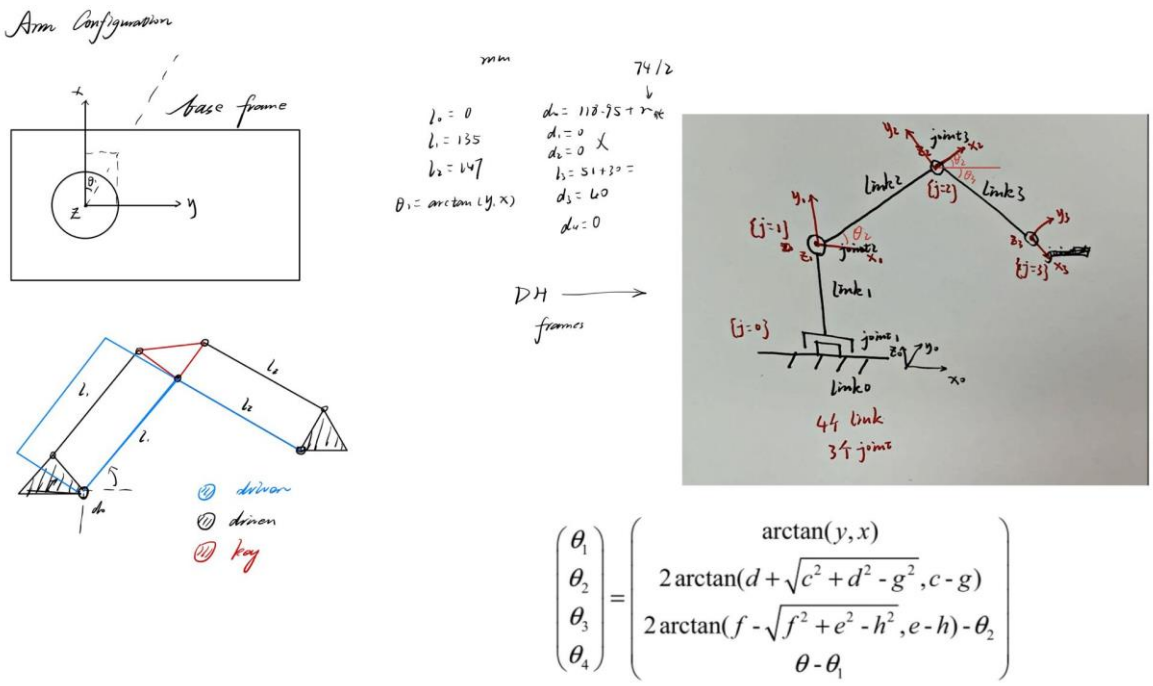
Symbol	Definition
l_0	The vertical distance between the main shaft of the rocker arm and the base shaft (mm)
l_1	main rocker arm length (mm)
l_2	The effective length of the secondary rocker arm (excluding the part of the horizontal holding mechanism at the end) (mm)
l_3	The horizontal distance between the end effector reference position and the end axis of the secondary rocker arm. (mm)
d_0	The distance between the main rocker arm rotation axis and the ground (the main coordinate system takes the intersection of the base rotation axis and the ground as the origin) (mm)
d_3	Vertical distance between the end effector reference position and the end of the secondary rocker arm (downward) (mm)
d_4	does not include the last joint, the parameter is set to 0 (mm)
ω_i	Angular velocity of i-th
ω_z	Angular velocity of robot car
v_x	Translational velocity along the x axis (forward direction)
v_y	Translational velocity along the y axis (side direction)
ρ	position error (pixel)
θ	Direction error (degree)
C_p	Control output from position error PID controller
C_d	Control output from direction error PID controller
C	Control output
VDR_L	Left voltage duty ratio of motor
VDR_R	Right voltage duty ratio of motor
$base$	Base value of voltage duty ratio
L_{PWM}	Left PWM
R_{PWM}	Right PWM

Acknowledgements | 8

We would first like to thank our course teachers Dr. Lei and Dr. Zhou, whose lectures were inspiring for us to complement this project, and also make the project more reasonable and challenging considering suggestions from the class. We would also thank the TAs for their assistance in laboratory managing and project testing.

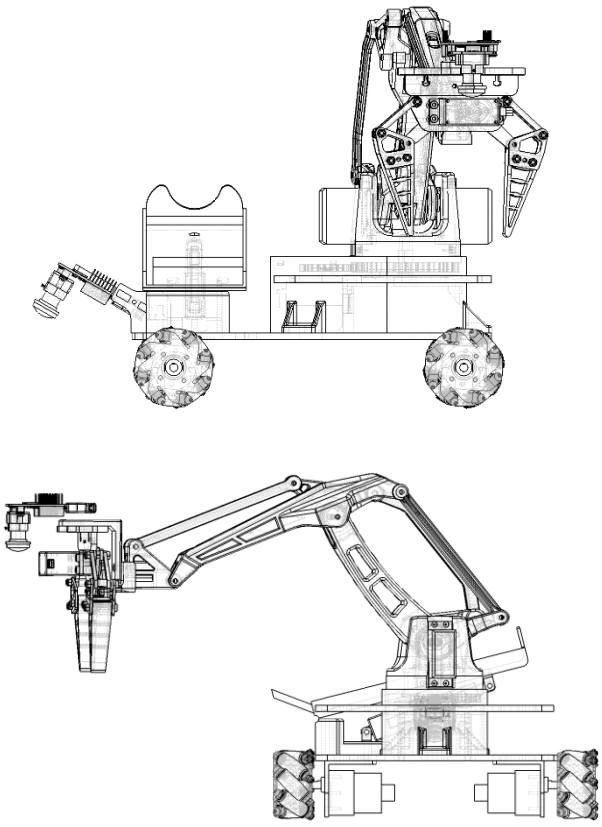
Appendix I. Sketches in Concept Design

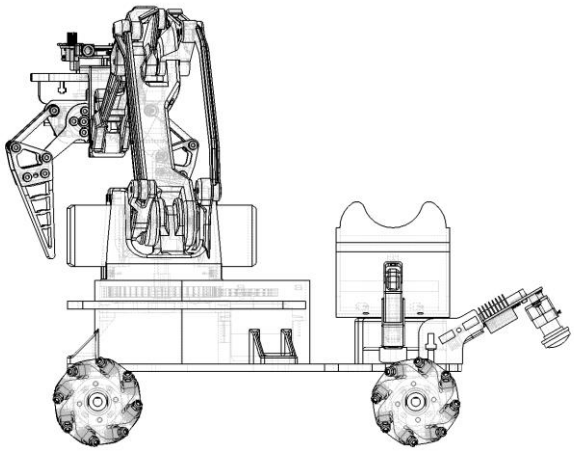


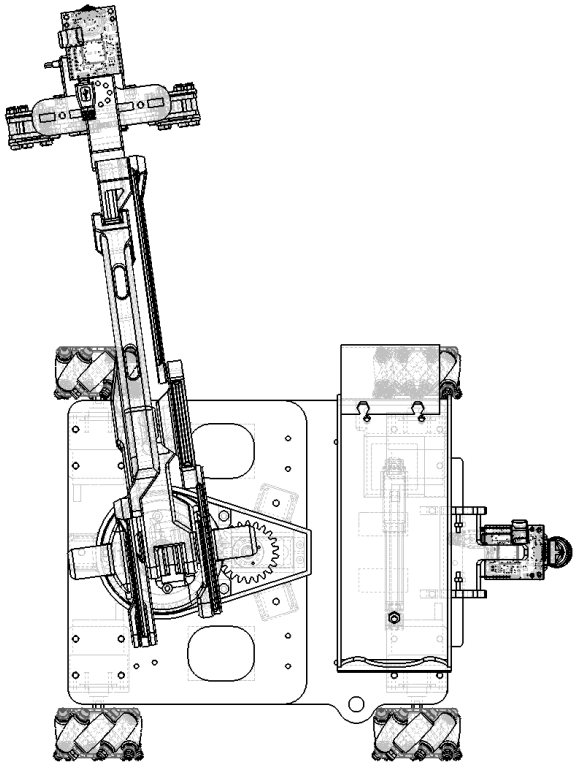


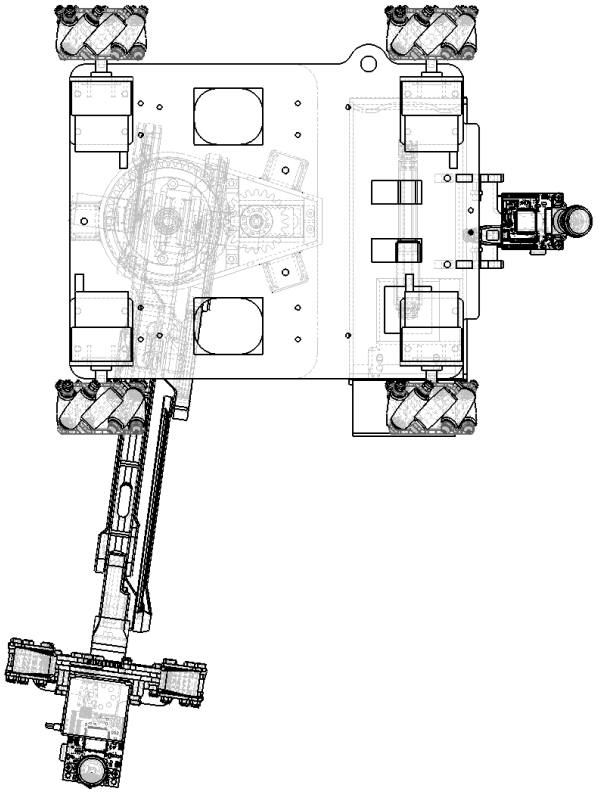
$$\begin{pmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{pmatrix}=\begin{pmatrix}\arctan(y,x)\\ 2\arctan(d+\sqrt{c^2+d^2-g^2},c-g)\\ 2\arctan(f-\sqrt{f^2+e^2-h^2},e-h)-\theta_2\\ \theta-\theta_1\end{pmatrix}$$

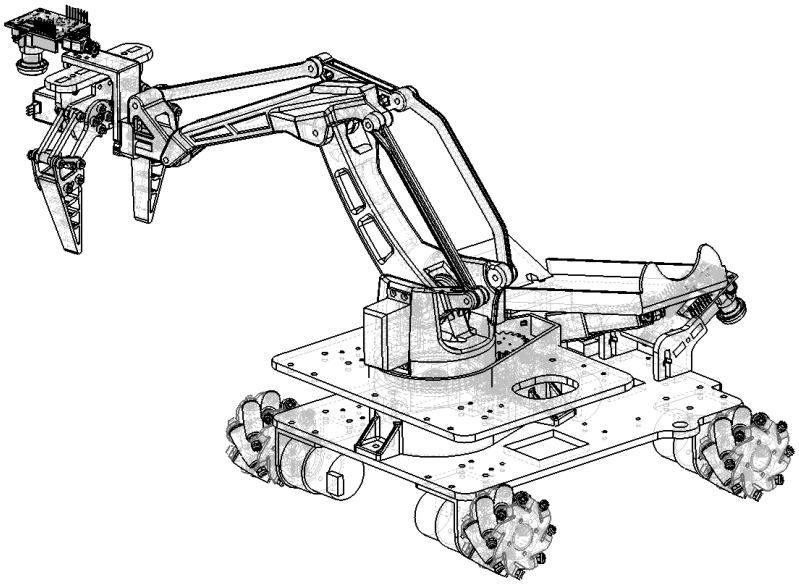
Appendix II. Engineering Draws with SolidWorks

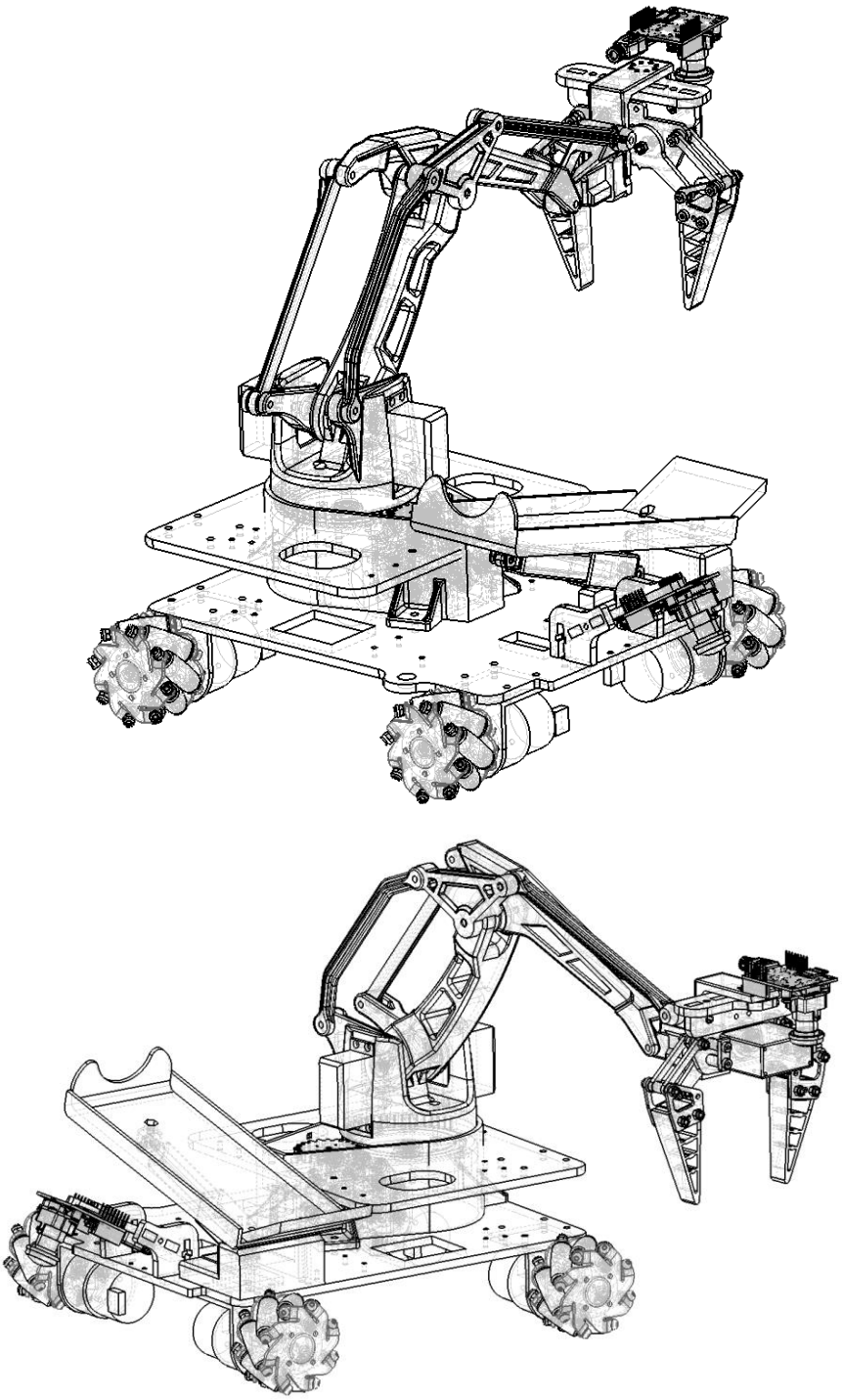


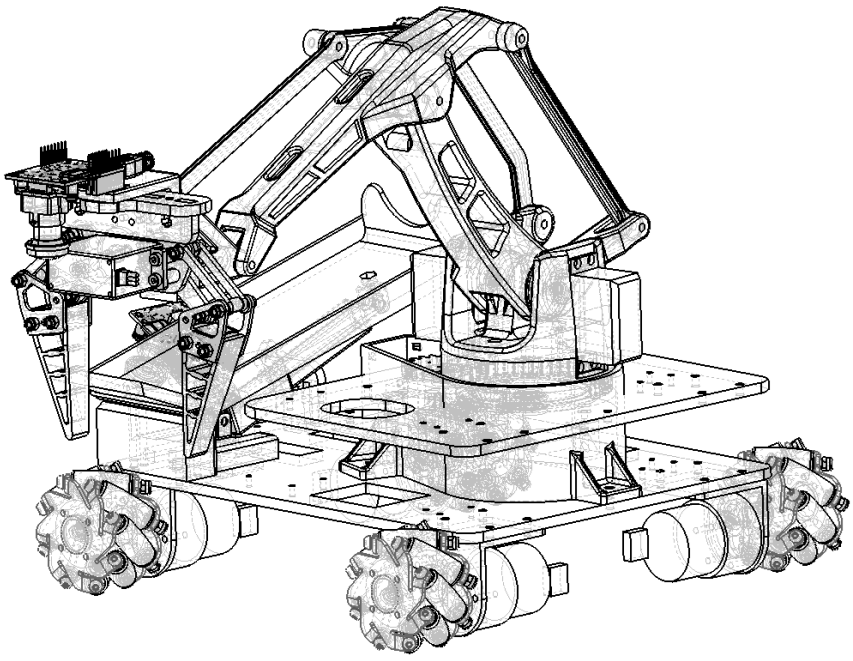


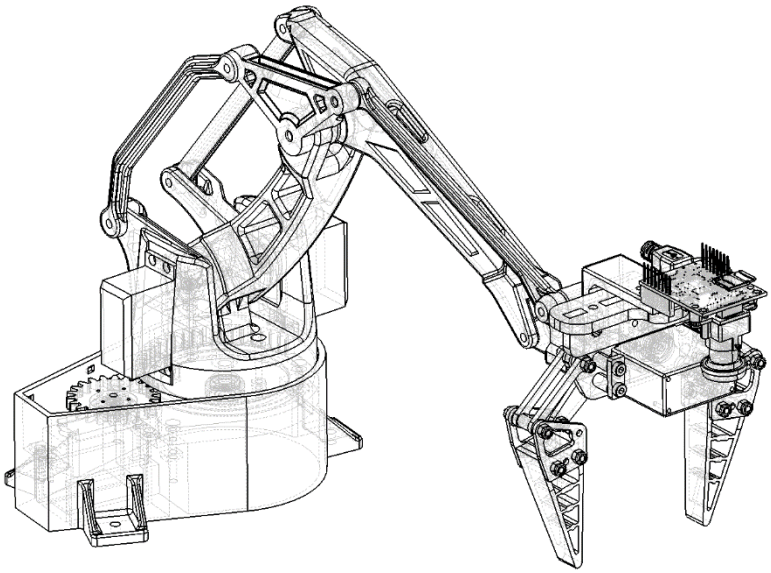


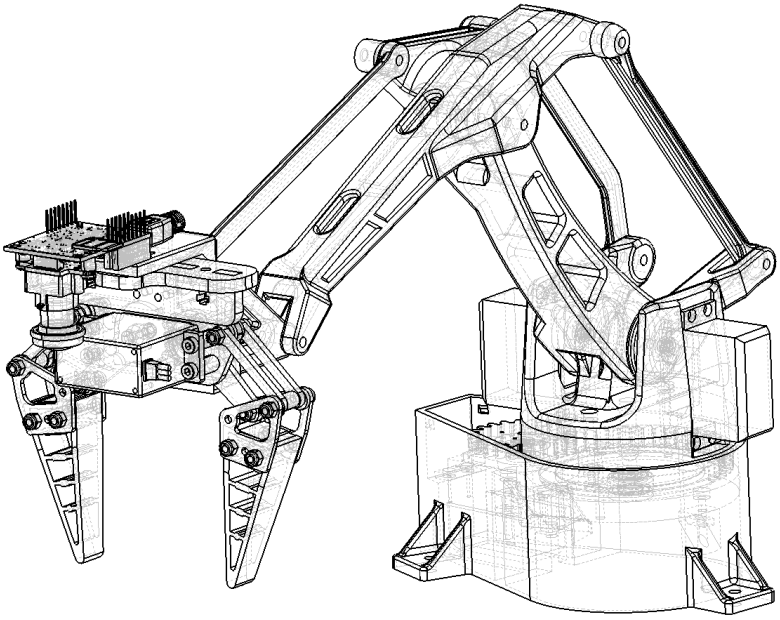


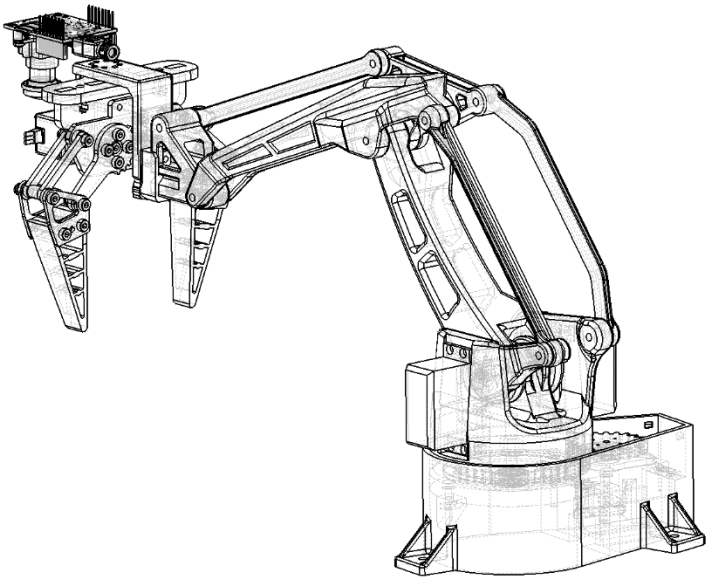


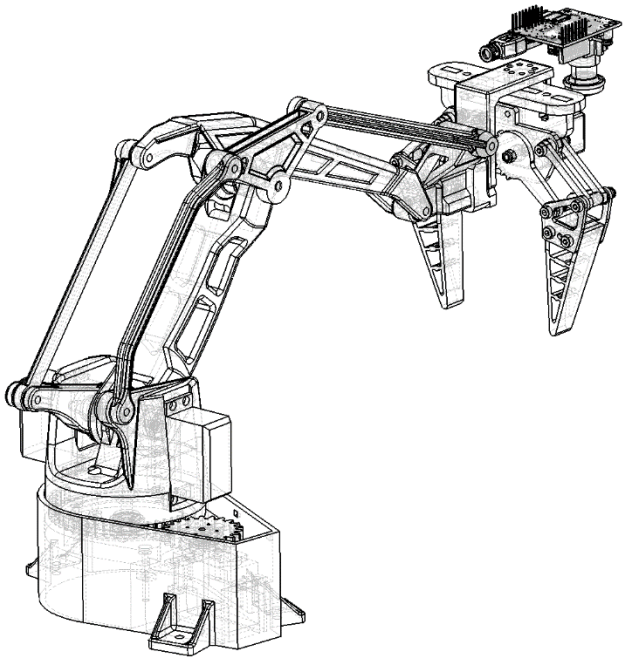


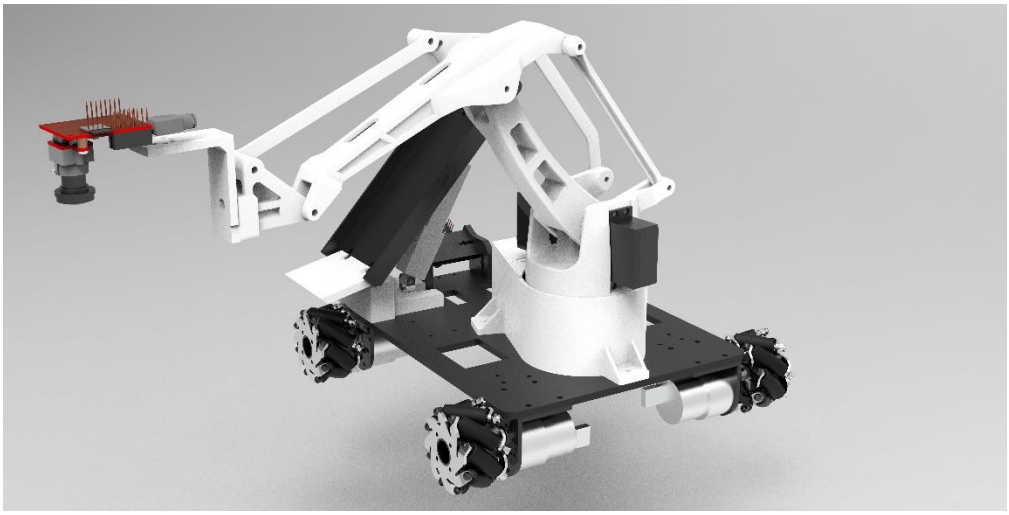












Appendix III. Product Design Specifications

An Obstacle Avoidance,Tracking and Loading Vehicle Based on OpenMV

Project Design Specification

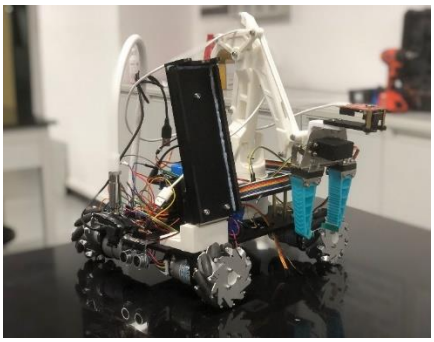
Product Identification

- Autonumous Obstacle Avoidance,Tracking and Loading Vehicle
- Function: Move on certain path, avoid obstables and deliver cargo
- Special features:
  - Highly adaptive to environment
  - Accuarate positining for cargo
  - Good loading/unloading ability
  - Smooth and fast movement
- Performance target: Can track the path accurately and avoid the obstacles,accurate classify, position and pick the cargo then finally unload at certain position.
- Using environment: Used indoors and in the situation in the game of Design and Manufacturing II project .
- User training required : No requirements.
- Key Project Deadlines:
  - Project Competition: 8:00 AM 11 June
  - Presentaion : 14:00 PM 11 June
  - Final project report deadline: 23:59 PM 17 June
- Physical Description:
  - Size: 35x20x35 (cm)
  - Detection: The prototype detects the path and cargo by two OpenMV.
  - Deliver:The prototype has a manipulator to grap the cargo to a loading box in the car then unload at certain position.
  - Movement: The prototype move with four Mecanum wheels with encoders
  - Materials: acrylic board, PLA and PBU for 3D printing
  - Financial Requirements: Expenditure: 2500 Yuan
- Life Cycle Targets:
  - Useful life and shelf life : One year
  - Cost of installation and operation:Every elements consume the energy in the battery especially when moving or grasping. Glue and welding may be used.
  - Maintenance schedule and location: Maintain in lab for electricity at D1C every 3days.
  - End-of-life strategy: All acrylic sheets, screws and nuts of all sizes, all kinds of electronic components, four Mecanum wheels, 3D-printed pieces can be reused.

Market Identification

- Market Size: No completed product in the market right now
- No need to work in the market
- Competing product: Self-delivery car by Meituan, product in smart car competition
- Branding strategy: Put the stability of the product in the first place, and then develop the flexibility (smooth and fast movement) and other additional characteristics.

Appendix IV. Details of Prototyped Machines



• Motor: CHR-GM37-520 12V 1:30 • Battery: 12V

Appendix V. Other Related Works

Details of the division of labor of team members in the whole project is shown in Table 5.

Table 5. Division of labor

Name	Task
Run He & Jialong Zeng	• Path planning for obstacle avoidance • Utilize ultrasonic sensors • Design controller for turning around with IMU • Design and tune PD controller for wheel control with encoders • Configure pins for stm32 with specific peripherals • Write codes for ultrasonic sensor, IMU and encoders. • Design circuits for power transformation • Debug issues related to Stm32
Ruoyao Tian & Lue Fang	• Debug OpenMV module • Communication between OpenMV and stm32 • Build tracking circuit • Write and modify OpenMV image processing program • Write and modify PID controller • Write stm32 program (motor.c, ustart2.c), make chip pins output PWM wave • Control the movement of McNum wheels • Adjust parameters related to tracking
Rui Qi & Yuchen Song	• Implementation of serial port-based servo debugging program • Overall mechanical structure optimization • Components manufacturing and assmbling • Inverse kinematic analysis and simulation • Implementation of inverse kinematic in reality • Designing object detecting algorithms • Implementing car stopping and gripping logic • Debugging and testing