LASER SYSTEM TO SHOOT DOWN MOSQUITOS

By

Fan Yang(fy10@illinois.edu)

Ruochen Wu(rw12@illinois.edu)

Yuxin Qu(yuxinqu2@illinois.edu)

Zhongqi Wu(zhongqi5@illinois.edu)

Final Report for ECE 445, Senior Design, Spring 2023

TA: Xinyi Xu

21 May 2023

Project No. 12

Abstract

Mosquito-borne diseases and illnesses are caused by bacteria, viruses, or parasites transmitted by mosquitoes. An effective method of protection against mosquitoes is necessary. Since traditional methods like nets, incense, traps, and lotions are not autonomous, we designed an autonomous device to detect mosquitoes based on vision and attack them using a laser. Our device uses real-time camera imaging to scan the surrounding space. The computer vision module, Yolov5, is utilized on Orangepi to process the images and detect mosquitoes. Additionally, an edge detection algorithm is applied to recognize the laser point. Arduino controls the precise movement of the camera and laser to realize attacking mosquitoes.

Contents

1. Introduction

1.1 Problem

Mosquito bites and blood feeding can cause itchy bumps and possibly a horrible infection. Mosquitoes cause at least 2.7 million deaths every year and about 500 million cases of mosquito-borne diseases occur annually. Mosquito-borne diseases and illnesses are caused by bacteria, viruses, or parasites transmitted by mosquitoes. The most prominent mosquito-borne diseases include malaria, West Nile virus, yellow fever, dengue, chikungunya, and Zika virus. Therefore, an effective method of protection against mosquitoes is necessary. Since traditional methods like nets, incense, traps, and lotions are not autonomous, detecting a mosquito and using a laser to kill it may be a feasible solution. We hope to design a system to detect mosquitoes based on vision and attack them using a laser.

1.2 Solution

The design can be divided into four subsystems: a positioning subsystem, an attacking subsystem, a remote-control subsystem, and a power system. The main task of the positioning subsystem is to detect a mosquito in the environment from a camera and locate its position. The attacking subsystem moves and switches the laser to kill the mosquito by emitting a high-power laser. The power subsystem supplies voltage to enable the components in other subsystems.

Firstly, the laser attached to the camera will emit a low-power laser to indicate the drop point, which is the position that can be attacked. We employ the yolov5s on Orangepi to do real-time detection with the input from the camera. Then, we move the camera to diminish the distance between the drop point and the mosquito until they coincide. The laser then emits a high-power laser to destroy the mosquito.

1.3 High-Level Requirements

1. Our software was able to identify mosquitoes that occupy at least 1/20 the size of the total pixels in the image.

2. The center of the attacking point should not be more than 1cm away from the mosquito to ensure that it will hit the mosquito.

3. Image shooting time and detection algorithm delay should not be longer than 3 seconds.

1.4 Block Diagram

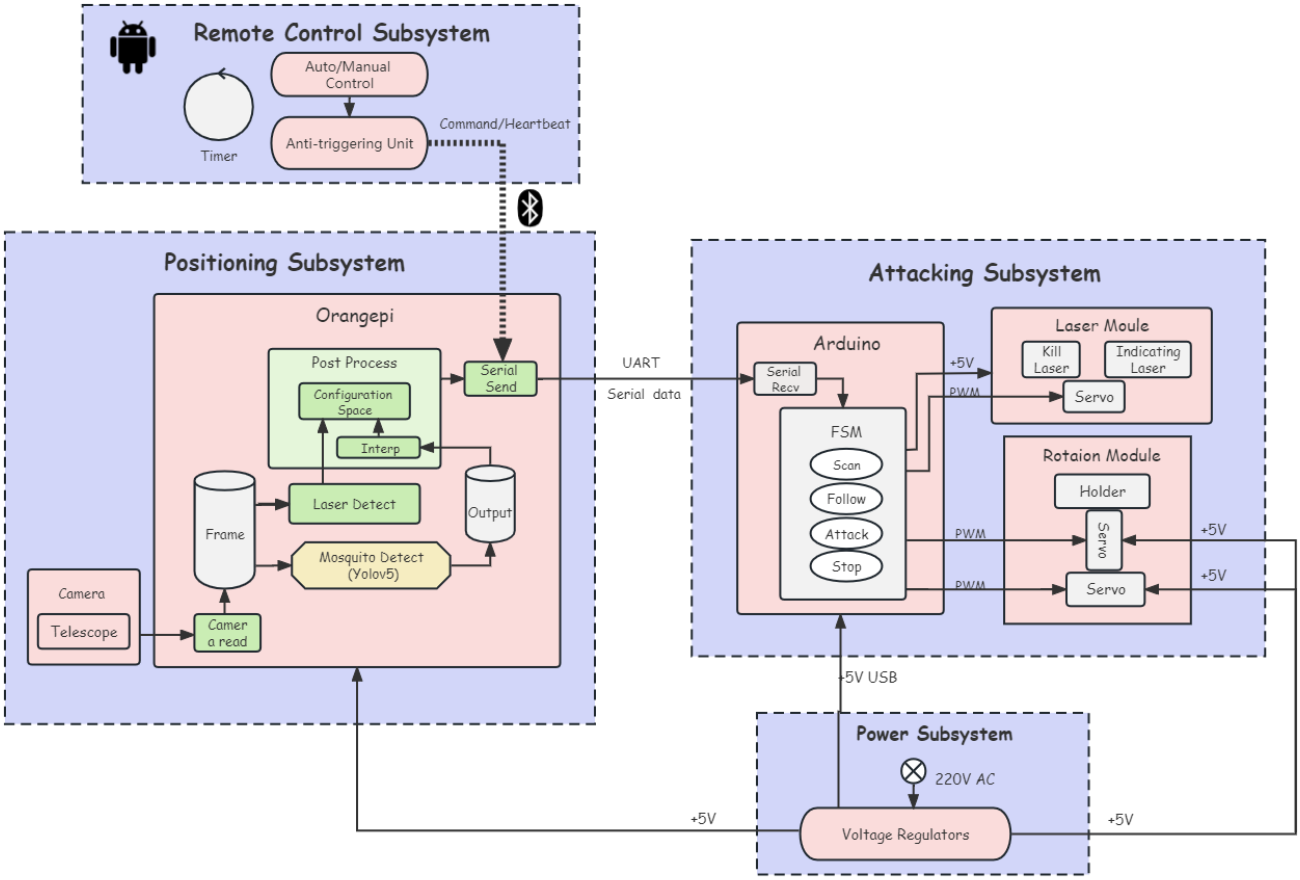


Figure1. Block diagram

Our system is mainly composed of four subsystems. The positioning subsystem will locate the positions of the mosquito and laser point in the photo, calculate the angles to rotate and pass them to the attacking subsystem. The attacking subsystem will then rotate the camera and laser towards the mosquito and use "kill laser" to destroy it. The power subsystem is responsible for the power supply. And there is also a remote-control subsystem that can control the whole system via Bluetooth communication.

1.5 Physical design

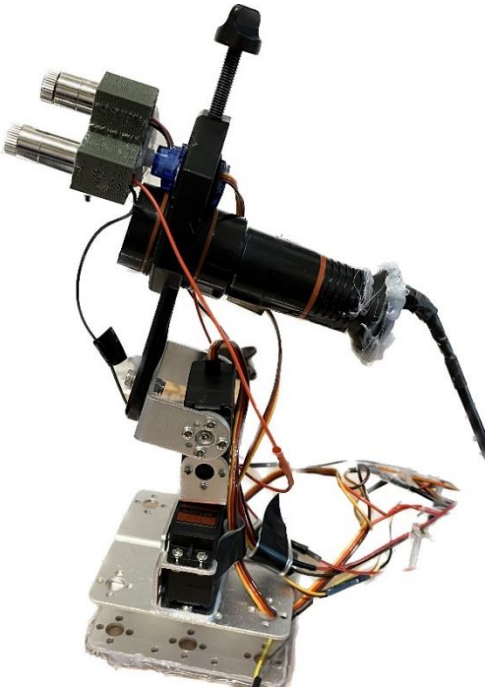


Figure2. Physical design

2 Design

2.1 Computer Vision Module

To position the laser point and mosquito, our project uses computer vision module to recognize mosquito and laser point. Considering taking full advantage of our computing power to improve the processing speed, we choose different computer vision model to detect the mosquito and laser.

2.1.1 Yolov5 model

1) Design procedure

To deploy the Yolov5 model in our project, we have opted for the Orange-Pi embedded development board with NPU integration. Our choice of model takes into account the limited computation capability and memory capacity of the board. After careful evaluation, we have determined that the Yolov5-s model best suits our project requirements. While it may offer slightly lower performance accuracy than other models, its small NPU footprint makes it an excellent fit for our purpose.

In our setup, the Yolov5-s model will receive high-resolution camera images. We have striven to optimize the detection capabilities of our system by configuring the network to identify mosquitoes occupying even the smallest region of the image frame. This configuration allows for a broader scanning horizon that can detect mosquitoes in all areas with minimal loss of accuracy.

2) Design details

We use a large-scale pre-trained model on insects, which can capture the features of insects greatly to enhance our performance of mosquito detecting. This model is trained on two million images using yolov5 release 7.0 as the basic model.

We collect our datasets of mosquito from different source, which contains different species, size and pose of mosquito. Then we do cut, blur, change the lightness of these mosquito images to augment our datasets. The size of mosquito ranges from 20*20 pixels to 80*80 pixels to simulate images captured by camera within 1.6 m.

For the size of camera image, we balance between the model processing speed and the pixels mosquito occupying in an image. Since the larger pixels of the camera image, the more pixels the mosquito occupies in one image, which is easier for the model to recognize the mosquito. However, the large images will be processed at a large expense, which will significantly slow down the processing speed of the model. For the sake of not disrupting the collaboration of all subsystems, we finally choose 1280*780 as the camera image pixels.

To better capture the feature of mosquitos, we change the architecture of our model. We add an anchor, whose tensor is [1,42,20,20], which can capture the 20*20 pixels as a feature.

2.1.2 Edge detection algorithm

1) Design procedure

Since it is difficult for yolov5 model to capture such small object like mosquito, we want to only train yolov5 on mosquito datasets to make it focus on small object detecting. Therefore, we want to choose another model to detect the laser point instead of yolov5 model. Since the laser point is a circle with a clear edge, we choose an edge detection algorithm as a mean to recognize the laser point. The edge detection method will receive the image from the camera, and output the coordinate of the detected laser point.

2) Design details

The edge detection model will receive an image with 1280*780 pixels size and then process it with Gaussian blur. Then it will extract the red and white area of the image and convert it to gray image. Then after some morphological operations on the images, we can use the edge detection to find the collections if all contours in the images. Then we find that the laser point is always the contour that has the largest lightness of all contour's areas. So, we set a threshold of the wanted contour's lightness and use it to filter the contours. Since the area size of contours is fixed, we also set a range to filter the contour with contour area. Since the edge detection model can be run on CPU and the yolov5 model can be run on GPU at the same time, we also increase our computing speed by taking full advantage of computing power.

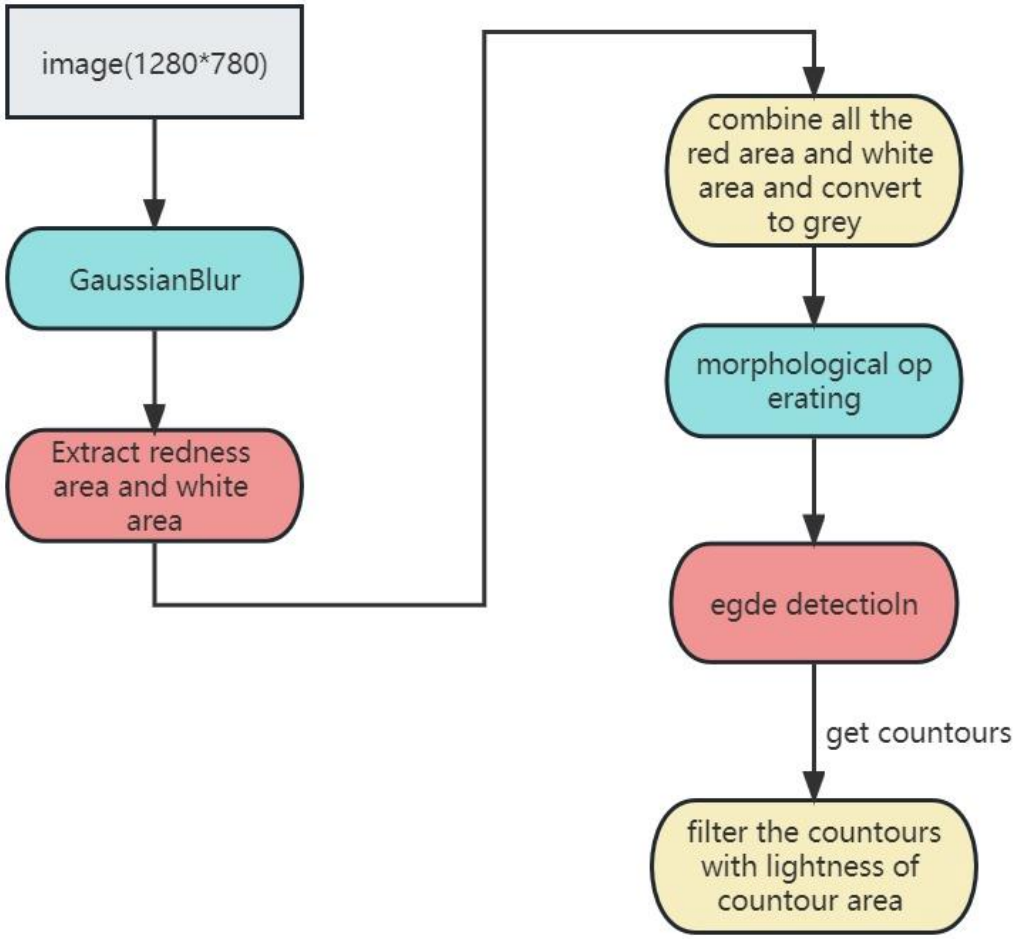


Figure3. Edge detection model

2.2 Inference Acceleration Module

2.2.1 Asynchronous Processing with NPU

1) Design procedure

The computational capacity of peripheral devices such as Orangepi is constrained. And using cloud computing leads to unreliability and significant delays. Based on this consideration, we chose to deploy the Yolov5 model on NPU integrated into the Orangepi.

NPU, or the Neural Network Processing Unit, simulates neurons and neuron connections on circuit layers, doing parallel computation. The NPU also presents a more cost-effective option than the GPU, especially

for embedded development boards. With a computation ability of 6 TOPS (Tera Operations Per Second), the NPU in Orangepi 5 significantly enhances the inference speed of our model.

The computation task of positioning subsystem mainly consists of three parts, including Camera Read, NPU Reference and CPU Post Process. Intuitively, we can implement the whole procedure in serial order, as shown in Figure4

Figure4. Serial order procedure

This implementation has two flaws. First, the response delay will be increasingly high. The camera capturing rate is much higher than the model processing rate. This leads to an increasing queue of frames waiting to be read, causing the Yolov5 model to lag behind and fail to process the most recent frame in a timely manner. Second, the NPU is not running all the time, interrupted by the post process of CPU. Therefore, the performance of NPU is not fully utilized.

2) Design details

To improve this, we came up with an asynchronous processing strategy. Figure5 shows the timeline of the program with this strategy at the very beginning.

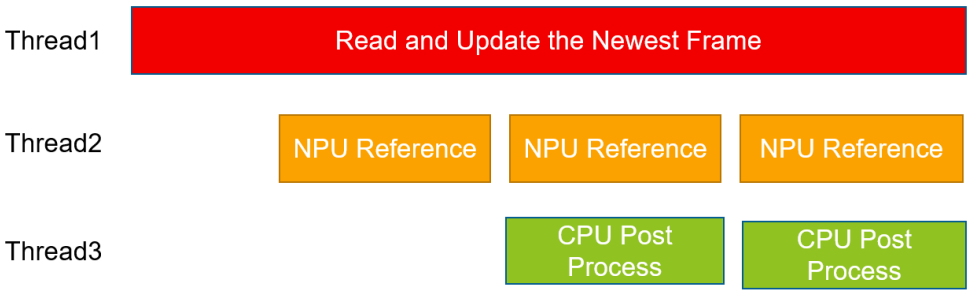


Figure5. Three threads strategy

Thread1 will constantly read from the camera and update the newest frame, making sure that the Yolov5 model always fetches the newest frame. In this case, the system can response in a timely manner.

When the first frame is available, the NPU starts the inference with Yolov5 model. After it is done, the CPU starts the post process. And here is the key point. At the same time, the NPU starts to process the next frame. As a result, we realized the asynchronous processing between NPU and CPU, reducing the execution time of one frame from $t_{cpu} + t_{npu}$ to $\max(t_{cpu}, t_{npu})$.

2.2.2 Model Quantization

1) Design procedure

The default input and output type of Yolov5 model is float16. However, the storage and high precision computation (HPC) of float16 consumes too much. The official toolkit of RKNN provides an acceleration strategy called quantization, which allows the model to do low precision computing.

2) Design procedure

We first map the float16 data to int8 data and input it to the model. Then the model can do the low precision computation (LPC), which is much faster. After we get the int8 output, we inversely map it to float16.

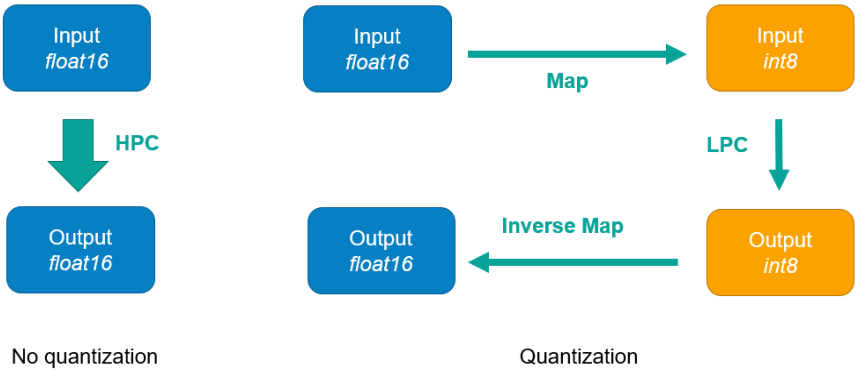


Figure6. Quantization process

However, there is precision loss during this procedure. To reduce this loss to a small scale, we calculate the error between original float16 output and the float16 output using quantization, taking this as the loss function and train the mapping function with machine learning models. Figure7 Shows this procedure.

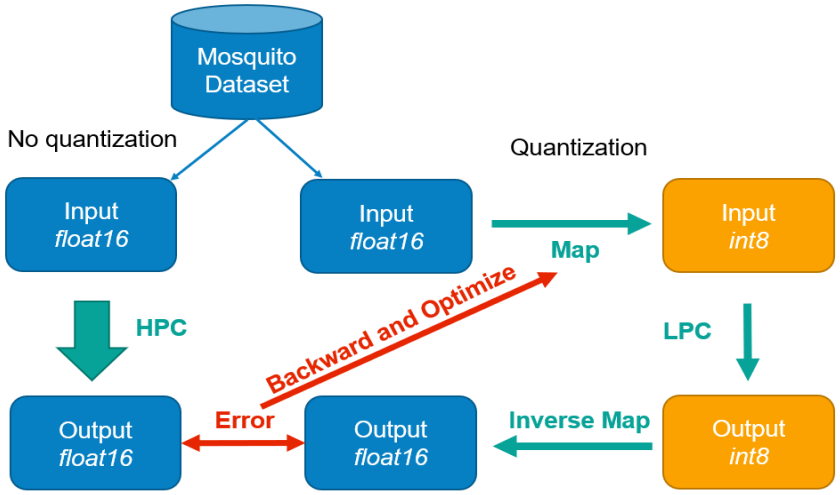


Figure7. Training the mapping method

After training it on our mosquito dataset, we reduced the precision loss to about 20%, which is acceptable in practice.

2.3 Rotation Control Module

This module controls how data transmits from Orange Pi to Arduino, then to servos and lasers and use the information to control the rotation of servos and laser switch.

2.3.1 Configuration Space Calculation

After we get the x and y coordinates of the laser spot and the mosquito in the camera using Yolo5 model and edge detection algorithm, we calculate the distance and write a function to define the angles and the directions for servos to rotate horizontally and vertically.

1) Design procedure

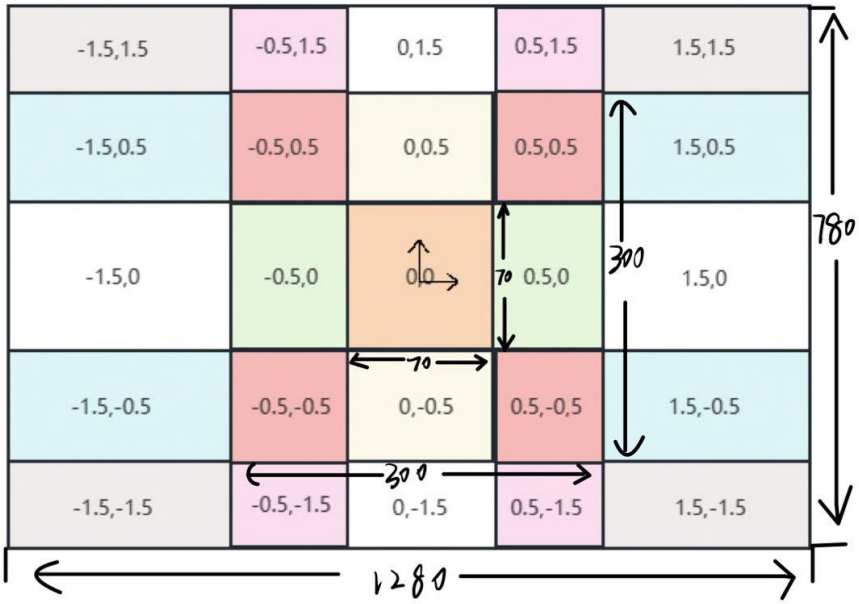


Figure8. Rotation angles and the corresponding distances

Figure8 shows how we get the rotation angles. In every rectangle there are two numbers. The first is the rotation angles in the horizontal direction and the second is the rotation angles in the vertical direction, which are all with degree as unit. The biggest rectangle is in the size 1280 pixels x 780 pixels, which is the size of camera field of view. The orange square is in the size 70 pixels x 70 pixels and the square bigger than it is in the size 300 pixels x 300 pixels. The logic of Figure8 will be illustrated in the next section.

2) Design details

A two-dimensional coordinate system is set up on the graph, with the center as the origin, the X-axis to the right, and the Y-axis to the up. The coordinates in the coordinate axis are (the distance between the mosquito and the laser horizontal axis, the distance between the mosquito and the laser vertical axis). For example, if the X distance is within –70 to 70, and the Y distance is within –70 to 70, then the rotation

angles should be (0,0). In the case that the X distance and the Y distance are all with in 300 but out of 70, the rotation angles should be (+/- 0.5, +/-0,5). The positive and negative numbers correspond to the axes. In this way, once the laser spot is far beyond the mosquito, it will move in a far speed towards the mosquito. However, if the laser spot is close to the mosquito, it will move slowly to ensure the accuracy. Finally, once the laser spot is near the mosquito, which means they are close enough, the laser will stop moving and prepare to shoot the mosquito.

2.3.2 UART Serial Communication

After getting the rotation angles, we need to send them and a variable mode, which store the state switch information, to Arduino. For data transmission between Orange Pi and Arduino, we use UART Serial Communication.

1) Design procedure

To implement the communication between the Orangepi 5 and Arduino UNO, there are several approaches. SPI is the Serial Peripheral Interface, whose Bus is composed of 4 signal lines: Serial clock (SCLK), Serial Data Output (SDO), Serial Data Input (SDI), and Chip Select (SS). Therefore, to use the SPI as the communication protocol, we should take care of 4 control signals and 4 wires. I2C is the INTER IC BUS, requiring only 2 wires for SCL and SDA. However, the protocol is even more complex than SPI. Compared to the pervious methods, UART (Universal Asynchronous Receiver Transmitter) only requires 2 signal lines for RX and TX with least programming effort. [12] Since our data exchange has the no requirement on bandwidth and anti-interference, we choose UART as the inter-device communication method.

2) Design details

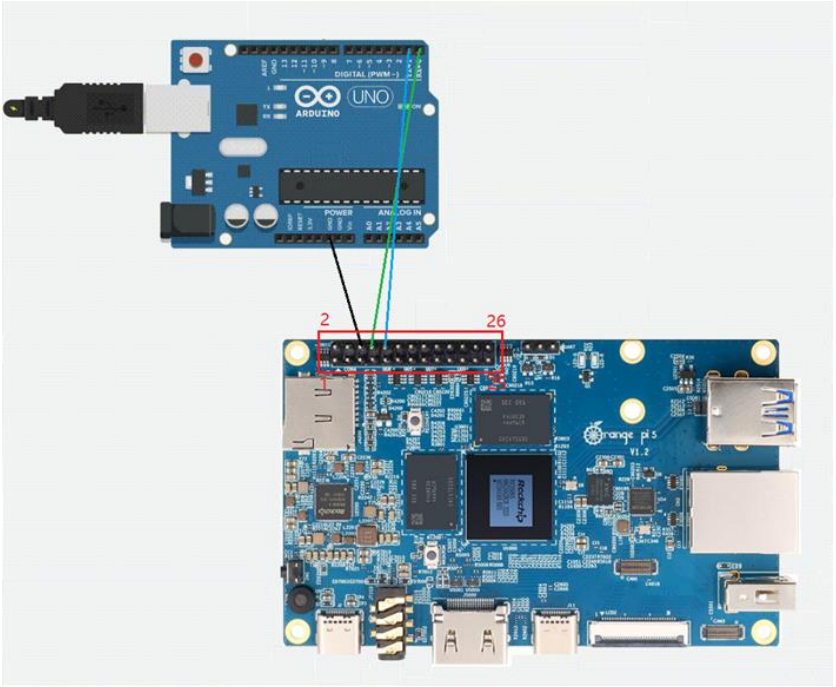


Figure9. Two Boards and the Wiring method for UART

The schematic above shows the wiring method for UART. The black wire connects two GND pins. The green wire transmits the data from Orangepi 5 to Arduino while the blue wire transmits backwards. The red block indicates the 26-pin GPIO area. There are 4 UART interfaces available. UART0 is used here, with pin6 as the Tx and pin8 as the Rx. [13]

In terms of the drive for GPIO, we use the modified version (for Orangepi) of wiringOP-python to use the UART interface. The data transport rate of UART protocol is relatively low, around 230 Kbps to 460kbps. [12] But it is far more sufficient for our demands. The process rate of yolov5 is estimated to be 10 frames/second (FPS=10). After analyzing each frame, we should put a message into the serial channel, composed of two target angles segmented by ";". In other words, the string is in the format of "<angle for servo1>;<angle for servo2>", with the maximum length (like "180;180"). The bandwidth requirement can be calculated in (2-1)

$$BW_{min} = L_{max} * sizeof(char) * FPS = 560bps \qquad (2-1)$$

where BW_{min} is the minimum required bandwidth, L_{max} is the maximum length of a command, FPS is the processing rate of yolov5

Since 560bps << 230kbps, UART is qualified to be used. The following APIs are used to implement the drive. The function wiringpi.serialOpen() is used to open the serial channel with specified pins and bit rate. The bit rate is set as 9600 bits/s. Another function called wiringpi.serialPuts() is used to write strings to the serial channel. And wiringpi.serialClose() aims to close the serial channel. [14]

2.3.3 Finite State Machine

In the Arduino, we design an FSM to switch the state of mechanical structure.

1) Design procedure

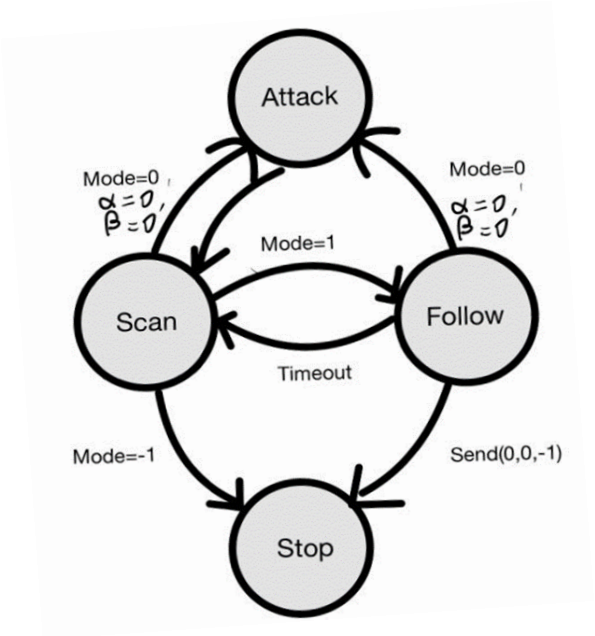
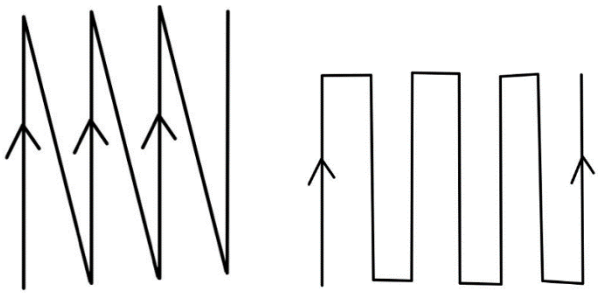


Figure10. The FSM of rotation

The FSM has four states with the data sent from Orange Pi as the condition to switch states.

2) Design details

At the beginning, we are at Scan, where the servos rotate to patrol the wall and cover all the space. The machine has a short delay for 5 seconds and then moves to the next place. The scanning route is shown in Figure11 (b). The reason is that when it reaches the top or the bottom, it will rotate only horizontally instead of moving slash down like what (a) does. (b) can well prevent the shaking of servos and the safety of the camera.



(a) (b)

Figure11. Two ways of scanning

Then, once the mode equals 1, the state will switch to Follow and the servos rotate according to the angles of transmission. If the mode equals 0 and rotation angles equal 0, which means the laser spot is aiming at the mosquito, the state will switch to Attack and the laser will switch to the high power one and shoot down the mosquito. After five seconds, the state will switch back to Scan. Also, if the mode equals negative one, the state switches to Stop and all lasers will be turned off.

2.3.4 PWM

The Arduino sends PWM to servos to control their rotation angles.

1) Design procedure

The larger the proportion of high level, the larger the duty cycle.

2) Design details

The time of high voltage can be calculated using the angle to rotate with (2-2)

$$time = 100/9 * angle + 500 \ \ (us) \qquad (2-2)$$

where angle is the desired rotation angle and the unit of time is μs.
Using (2-3) can we send the PWM to servos

$$servo.writeMicro\,\mathrm{sec}\,o\,nds(time) \qquad (2-3)$$

where time is the value in microseconds.
It is worth mentioning that our servo is carefully selected so that its rotation accuracy is up to 0.4 degrees, which is small enough to accurately move to the mosquito.

2.4 Mechanical Structure

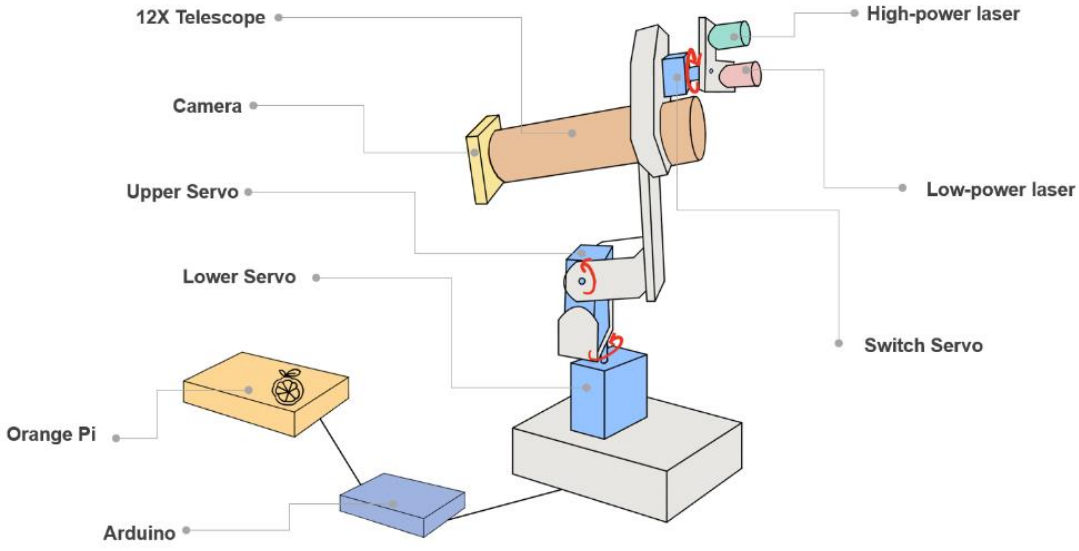


Figure12. Mechanical structure

2.4.1 Camera module

1) Design procedure

We initially tried to detect mosquitoes only with a camera, but that way we could only detect mosquitoes when we were 30 centimeters away. We added a telescope in front of the camera in order to increase the working range and better simulate real life situations, such as using the device in the dormitory.

2) Design details

After experiments, we finally chose attach a telescope with 12 times magnification to camera IMX219-77 to observe mosquitoes 1.6 meters away from the camera, so mosquitoes that are far away can be larger in the camera and can be identified more easily. We can also scan larger areas by moving the camera and telescope.

2.4.2 Rotation module

1) Design procedure

Since the camera needs to be able to scan in space, the moving device needs to be at two degrees of freedom, so we chose a servo that can change the horizontal position and a servo that can change the vertical position.

2) Design details

Two servos are used to control the position of the camera and lasers. The upper servo rotates around a horizontal axis to change the height of the camera and laser. The lower servo rotates around a vertical axis to change the horizontal position of the camera and laser. Therefore, the camera can scan the surrounding environment. And the rotation accuracy of servo DS3115 is up to 0.4 degrees to ensure that we can attack the mosquito accurately.

Figure13. Servo structure with two degrees of freedom

2.4.3 Laser module

1) Design procedure

We should make sure that the low-power laser and the high-power laser can shoot in the same position and can be exchanged in a short time, so we use a rotating way to place the two lasers. According to the laser system built by Rakhmatulin, lasers larger than one watt are lethal to mosquitoes. If we can accurately attack and burn the mosquito's wings, we can use a lower energy laser, about hundreds of watts. However, according to the Center for Devices and Radiological Health (CDRH), laser that has average power of more than 500mW belongs to Class 4. It is able to hurt the eyes and skin immediately, and there is a possibility of burning combustible materials. Therefore, there is a potential that the laser that we might use in our project may harm people. For the sake of limited time and human safety, we used low-power lasers to attack mosquitoes in this project.

2) Design details

One servo and two lasers are attached to the camera and telescope, which is used to switch laser types by rotating 90 degrees clockwise or 90 degrees counterclockwise. The low-power laser is always on for detection. Its drop point indicates which is the position that can be attacked. The drop point should always show up on the camera. When finding the mosquito, the servo needs to be rotated 90 degrees so that the high-powered laser takes up the position of the low-powered laser. The laser will turn back after attacking the mosquito.

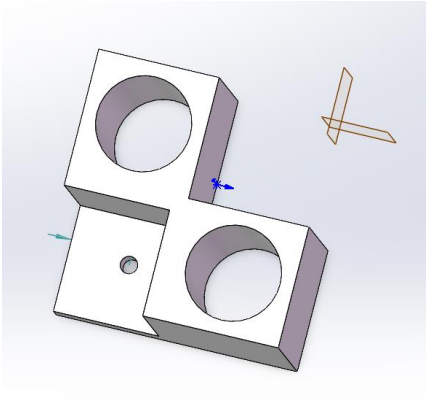


Figure14. Laser Switch design drawing

2.5 Remote Control

2.5.1 Android Application

To enhance the safety of our system, we developed an application on the Android platform, allowing the user to control the system by remote via Bluetooth. The user interface is shown in Figure15.

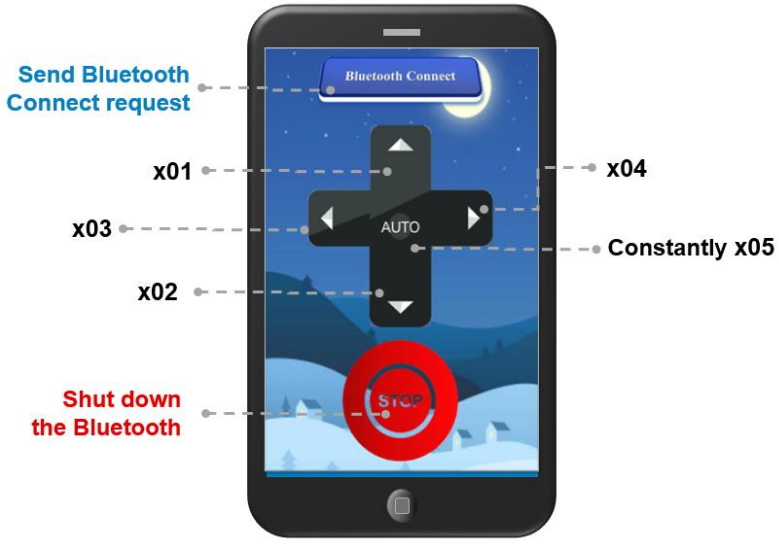


Figure15. Android Application

When the user presses any arrow button, the application enters the “Manual” mode. In this case, the corresponding command will take off the movement of the system.

When the user presses the AUTO button, the application enters the “Auto” mode. In this case, the system will automatically detect the mosquito, move the laser to it and destroy it. If there’s no mosquito, it will automatically scan the workspace.

2.5.2 Anti-triggering

Remote control has potential unreliability. Normally, when the application is idle, it does not send any signal. And if we want to stop the system, we send a "stop" command. However, if the Bluetooth connection is somewhat cut off, there is no way to stop the system, which could be very dangerous

Therefore, we designed an anti-triggering mechanism. Now, even when the application is idle, it sends heartbeats regularly to the system. When the system cannot receive any signal, it will immediately shut down the laser and stop running. This mechanism can prevent the risk of losing control.

2.6 Power Supply

1) Design procedure

Power supply subsystem is determined by the components we used in our project.

2) Design details

Source power is 220V AC. We use AC/DC adapter to convert 220V AC to 5V DC to power Orange Pi, Arduino, upper servo, and lower servo. Camera is powered by Orange Pi. Laser and laser servo are powered by Arduino.

Table 1 Table of Power supply of consumers

Consumer	Voltage	Current	Power
Upper servo and lower servo	5V	1.5A	7.5W
camera	5V	100mA	0.5W
Laser servo	5V	300mA	1.5W
laser	5V	1mA(red)/ 2mA(green)	5mW(red)/ 10mW(green)
Orange Pi	5V	4A	20W
Arduino	5V	80mA	0.4W

3 Design Verification

3.1 Computer Vision Module

3.1.1 Yolov5 model

1) Requirements

a) Yolov5 model can distinguish between mosquito and other items. It should give mosquito a high confidence and other items a low confidence. The difference between the confidence of mosquito and other items should be more than 0.5.

b) Yolov5 model should recognize the mosquito stably, which means it should continuously recognize the mosquito as mosquito with the small rotation of camera.

c) Yolov5 model should recognize mosquito with different species, pose and size at different distances.

2) Verifications

a) Make 10 attempt to test mosquito and other items to make sure that the model give a difference more than 0.5 between the confidence of mosquito and other items.
b) Move the camera slowly and make sure the model can continuously recognize the mosquito successfully.
c) Use mosquito with different colors and size and stick it to the test board with different angle. Make sure in most cases the model can recognize them successfully.

3.1.2 Edge detection model

1) Requirements

a) When there is not laser point in camera image, output laser point coordinate of (-1, -1) to mark there is no laser point.
b) If exists red laser point in camera image, mark it with a red circle and output the correct laser point coordinate.
c) Even there is two detected the red laser point, only output one's coordinate with highest area lightness.

2) Verifications

a) When there is no red laser point, make sure that only coordinate of (-1,-1) will be output as laser point position.
b) Use one red laser point, make sure there is a red circle mark its position and output the correct coordinate of laser point.
c) Use two red laser points and make sure only the one with lightest lightness will be marked as laser point.

3.1.3 Camera & Telescope

1) Requirements

a) The camera should have a resolution higher than 1280*780.
b) The camera with the telescope can focus on the mosquito successfully at 1.6m distance.

2) Verifications

a) Connect the camera to the processor, and input one image. Check whether he resolution of the image is higher than 1280*780.
b) Check that the mosquito at a distance of 1.6 meters can be clearly seen in the picture.

3.1.4 Embedded Development Board

1) Requirements

a) The board needs to have several USB interfaces to receive the data from the camera, mouse, keyboard and the Bluetooth module.
b) The processing rate should be higher than 5 frames per second to reduce delay.
c) The operating system on the board should support the python execution and dependencies of yolov5 model.

d) The board should have the GPIO pins supporting the serial communication.

2) Verifications

a) The Orangepi 5 has three USB type-A interfaces and one USB type-C interface, sufficient for our use.

b) We use some strategies and the NPU to accelerate the running. In our experiment, we measured the processing rate, which is higher than 5 frames per second.

c) The environment on Orangepi has no significant difference with Ubuntu on PC. The python and dependencies of yolov5 are fully supported.

d) There are 4 groups of Rx and Tx with UART protocol. Actually, one group is enough for us to use.

3.2 Rotation Control Module

3.2.1 Servo

1) Requirements

a) It can't shake violently while rotating.

b) The speed must be slower than 0.032 rad/s (1.83 degrees/s).

c) The max rotation angle of the horizontal servo should be 360 degrees. And the vertical one should be able to rotate at least 180 degrees.

2) Verifications

a) Test it by sending signals to it and make sure it will not shake violently especially when the laser spot is close to the mosquito.

b) Fix a mosquito on the wall and make sure it can be reached by the laser spot.

c) Make sure servos can reach any direction in space.

3.2.2 Angles Calculation

1) Requirements

a) When the laser and the mosquito distance is far away, the laser close to the mosquito faster.

b) When the laser is close to the mosquito, the speed of the laser approaching the mosquito is slow.

c) When the laser is approximately at the same position as the mosquito, the laser stop moving.

2) Verifications

a) Move the laser and the mosquito within the scope of the camera and turn on the power. Keep them as far away from each other as possible and calculate the speed of the laser.

b) Keep them close to each other and calculate the speed of the laser.

c) Keep them the same position and make sure the laser will not move any more.

3.2.3 UART Serial Communication

1) Requirements

a) The Arduino can receive the message including three variables from Orange Pi.

b) Make the servo receive the latest message as possible.

2) Verifications

a) Send massage from Orange Pi and make sure the same message can be seen in the serial monitor of Arduino.
b) Create a thread for sending message and a Global variable to store the latest message in Orange Pi so that Arduino can receive the latest message as possible.

3.2.4 Finite State Machine

1) Requirements
    a) Make sure all four states can be reached and the reaching condition is the same as what we illustrated before.
2) Verifications
    a) Send the reaching conditions to Arduino and check the states.

3.2.5 PWM

1) Requirements
    a) Servos can rotate the angles sent from serial monitor.
    b) Arduino can control the Servo to rotate non-integer degrees.
2) Verifications
    a) Input angles to serial monitor and check the real angles read from the servo.
    b) Use myservo.writeMicroseconds(time) to control the rotation angle of the servo and check the real angles read from the servo.

3.3 Remote Control

3.3.1 Remote Control Application

1) Requirements
    a) The application can run on mobile devices like Android phone
    b) The application should have an understandable user interface, and useful features for remote control.
    c) The application should deal with the case when the communication is cut off by accident.
2) Verifications
    a) The application is developed for Android platform and tested on serval devices
    b) The application has beautiful GUI and clear instructions on each button. It provides both manual control and automatic control approaches.
    c) The application has the anti-triggering mechanism as mentioned before. When the communication is cut off, the whole system will stop running immediately.

4. Costs & Schedule

4.1 Parts

Table 2 lists the part costs of our system.

Table 2 Parts Costs

Part	Manufacturer	Retail Cost (RMB)	Bulk Purchase Cost (RMB)	Actual Cost (RMB)
Camera (IMX219-77)	Sony	78	78	78
12x Mobile telescope	KZKP	47	47	47
Orangepi 5	Orangepi	748	748	748
Bluetooth Module	baseus	5	5	5
Servos (DS3115) and pan-tilt	Xinrui Tech	180	180	180
Laser servo (sg90)	Hq	10	10	10
Acrylic Plate	Knighthut	16	16	16
Arduino UNO	Arduino	17	17	17
Green Laser	RXHC	25.8	25.8	25.8
Red Laser	RXHC	7.14	7.14	7.14
Connecting Piece	Xingtelang	17.5	17.5	17.5
Charger for servos	Zhitan Model	54	54	54
Total	\	**1205.44**	**1205.44**	**1205.44**

4.2 Labor

The team members are expected to work 2 hours per day and 5 days per week, starting from Mar.20th to May.22nd. Totally there will be 45days x 2h/day = 90 h. The salary is 30 RMB/hour.

Therefore, the money spent for the total labor is 30RMB/h x 4 x 90h = 10800RMB

4.3 Schedule

The timeline is shown in the table below. Week 1 refers to the week starting from Mar.20th to Mar.26th.

	Overall Goal	
Week1	Prepare the necessary parts and environment.	● Zhongqi Wu: Train the Yolov5 model on the basic mosquito dataset ● Ruochen Wu: Buy all the needed parts listed above. Build a PWM generation program on RK3399pro ● Yuxin Qu: Designed the mechanical structure based on the functions.

		• Fan Yang: Set up the running environment of Yolov5 on RK3399pro.
Week2	Setup the subsystems	• Zhongqi Wu: Create a dataset for figures of laser drop points and train the model on it • Ruochen Wu: Create the above dataset. Use the PWM generation program to drive the servos • Yuxin Qu: Built the rotation module (a 2-DOF pan-tile) and tested it. • Fan Yang: Deploy the Yolov5 model on RK3399pro
Week3	Optimize the subsystems	• Zhongqi Wu and Ruochen Wu: Train the model on an augmented dataset to simulate the observation from a far distance. • Yuxin Qu: Decided what kind of high-power laser should be used and tested the power supply system. • Fan Yang: Use PWM to drive the high-power laser, use NPU (neural network process unit) to accelerate the inference of the model
Week4	Integrate the subsystems	• Zhongqi Wu and Ruochen Wu: Build the interfaces for the Yolov5 to control the rotation information contained in PWM, based on the target position • Yuxin Qu: changed the mechanical structure and designed the new structure. • Fan Yang: Build the circuit connecting all the hardware. Build the mechanical structure and mount all the subsystems on it (only add low-power laser)
Week5	System evaluation and refinement	• Zhongqi Wu, Ruochen Wu, Fan Yang: Test the performance of the positioning subsystem in normal cases and edge cases, make refinement on logic bugs • Yuxin Qu: Built the new structure. Evaluate the precision of the rotation module. Evaluate the stability of the mechanical structure.
Week6	Launch high-power laser	• Zhongqi Wu, Ruochen Wu, Fan Yang Design a mechanism to avoid shooting when there is potential harm to humans • Yuxin Qu: Built the new laser switch.
Week7	Optimization	• Zhongqi Wu and Ruochen Wu: Label a new dataset with a more complex background. Train model on it.

		• Yuxin Qu: Helped others to do the test and prepare for the mock demo. • Fan Yang: Label the new dataset, optimize the detection algorithm
Week8	Test and refinement	• Zhongqi Wu and Ruochen Wu: Test the positioning subsystem and make refinement • Yuxin Qu: Test the attacking subsystem and make refinement
Week9	Application scenario extension	• All members: Discuss how we can extend the application scenario of our system.

5. Conclusion

5.1 Accomplishments

Our model can detect and attack mosquito autonomously at 1.6m distance, which beyond the distance that can be reached by the existing technology. It is also portable, since it can work without computer and only with an orange pi and an Arduino. More importantly, for safety and convenience, we can use a APP developed by us to control it remotely. Once the device fails to receive the signal from the APP, it will stop working.

5.2 Uncertainties

5.2.1 Positioning subsystem

1) We use the combination of camera and telescope in this project, which is a cheap way to magnify image. If using a telephoto lens, we will get clear pictures with higher pixels.

2) The slow focus process of Camera IMX219-77 slows down our detection speed, which leads to the miss of mosquito detection during the scanning mode.

3) Our device cannot work in a very complex environment since too many items will influence its recognition efficiency greatly.

5.2.2 Attacking subsystem

1) The rotation accuracy of laser servo is not enough, which cause error when attacking.

2) The mechanical structure of switching module is unstable. The focal length of the laser cannot remain constant after adjustment.

5.2.2 Power subsystem

1) Even though we use adapters, the value of current of the upper servo and the lower servo will still change in actual use. It would be better to keep 1.5A at runtime.

5.3 Ethical considerations

Our project aims will detect and mimic killing the mosquito with a laser, it will bring some potential safety problems. For example, a laser with high power that can kill mosquitoes can also hurt people in some way.

According to IEEE Ethics term 1 [7] and ACM Ethics term 1.2 [8], protecting the health of people is our first principle. Therefore, in our project, we should use a model with high accuracy and double check to avoid the laser going out of control to hurt non-mosquito objects. In other words, we have to improve our accuracy in recognizing a mosquito and shoot it after we make sure it is indeed a mosquito.

Another ethical issue is about privacy. Because we have to use a camera to monitor the surrounding environment to identify whether an item is a mosquito a not, it is unavoidable to identify some private items or people. IEEE Ethics [7] and ACM Ethics [8] both focus on the importance of privacy. To avoid private data leakage, we would delete it after finishing the identification.

Since our device will rely on electricity, electrical safety should be considered in our project. According to basic electrical safety at the University of Washington [9], we should prevent electrical shock, and electrical explosions during our device work. Therefore, checking the device to ensure each component is connected properly and not placed in a wet environment is the premise.

Also, during detection, the laser with low power will move freely in the room, it may hurt people's eyes if there are people in the work area. Therefore, we want to keep people away from the work area after the device starts. It means we would use a manipulator to start and end it remotely. Just in case, we will also set up a mechanism that limits the moving trail of the laser to avoid people when detecting the mosquito.

5.4 Future work

Our device will sometimes make mistake during the scanning mode. It will misrecognize other items as mosquito or miss mosquito sometimes. This problem can be mainly contributed by the poor focusing function of our used camera. Focusing speed plays an important role in our device work. The performance of our device can be improved by use a camera with better focusing function.

We only use a low-powerful laser as the signal of the authentic attack laser. The attacking laser should be larger than 1w and should not be divergent at 1.6 m distance. Since our existing algorithm of mosquito detection is not very accurate, we concern it will bring safety problem if using a laser of more than 1w. Also, the laser satisfying our requirement of power and divergence is usually really big and expensive, which is not an ideal choice for our project. The mosquito can be truly killed if we can improve our existing algorithm of mosquito detection and find a small and cheaper powerful laser.

The current device cannot track and predict the movement of mosquitos. Study the movement of mosquitoes, including angle, speed, wind speed, etc. The developing algorithms to predict the future position of mosquitos would improve the usefulness of this device.

Using algorithms to detect human eyes and skin to prevent attacks and protect human safety. Detect special objects, such as inflammables, to prevent hazards to the surrounding environment.

References

[1] MosquitoReviews Editors, “Mosquito Deaths & Mosquito Borne Disease Statistics [2020],” Mosquito Reviews, 2019. https://mosquitoreviews.com/learn/disease-death-statistics

[2] 飞桨 PaddlePaddle, “瑞芯微 RV1126、RK1808 等 AI 硬件 NPU 部署，直达产业落地_bilibili,” www.bilibili.com. https://www.bilibili.com/video/BV1oT411U7Fw/?vd_source=d3e883a246caaa5f767d5caa2a8c2ed6 (accessed Mar. 13, 2023).

[3] Wikipedia Contributors, “Mosquito,” Wikipedia, Apr. 16, 2019. https://en.wikipedia.org/wiki/Mosquito

[4] Waveshare Wiki Editors, “IMX219-77 Camera” www.waveshare.net. https://www.waveshare.net/wiki/IMX219-77_Camera (accessed Mar. 13, 2023).

[5] DFRobot Forum Editors, “FIT0731%20%20-DFRobot.” Wiki.dfrobot.com.cn, wiki.dfrobot.com.cn/FIT0731%20%20 树莓派两自由度云台.

[6] Peterson, Zachariah. “Embedded System Power Supply Guidelines for Power Integrity.” Altium, 24 June 2019, resources.altium.com/p/embedded-system-power-supply-guidelines-power-integrity. Accessed 13 Mar. 2023.

[7] IEEE, “IEEE Code of Ethics,” ieee.org, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html

[8] Association for Computing Machinery, “ACM Code of Ethics and Professional Conduct,” Association for Computing Machinery, Jun. 22, 2018. https://www.acm.org/code-of-ethics

[9] University of Washington, “Basic Electrical Safety | EHS,” Washington.edu, 2018. https://www.ehs.washington.edu/fire-life/basic-electrical-safety

[10] I. Rakhmatulin, “Raspberry PI for Kill Mosquitoes by Laser,” www.preprints.org, Jan. 2021, doi: https://doi.org/10.20944/preprints202101.0412.v1.

[11] R. Ildar, “Machine vision for low-cost remote control of mosquitoes by power laser,” Journal of Real-Time Image Processing, Feb. 2021, doi: https://doi.org/10.1007/s11554-021-01079-x.

[12] “UART vs SPI vs I2C | Difference between UART,SPI and I2C,” www.rfwireless-world.com. https://www.rfwireless-world.com/Terminology/UART-vs-SPI-vs-I2C.html

[13] “Orange Pi 5-Orange Pi” www.orangepi.cn. http://www.orangepi.cn/html/hardWare/computerAndMicrocontrollers/parameter/Orange-Pi-5.html (accessed Apr. 12, 2023).

[14] O. Pi, “Note,” GitHub, Apr. 01, 2023. https://github.com/orangepi-xunlong/wiringOP-Python (accessed Apr. 12, 2023).

Appendix A Requirement and Verification Table

Table 2 System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Requirements for Yolov5 model a. Yolov5 model can distinguish between mosquito and other items. It should give mosquito a high confidence and other items a low confidence. The difference between the confidence of mosquito and other items should be more than 0.5. b. Yolov5 model should recognize the mosquito stably, which means it should continuously recognize the mosquito as mosquito with the small rotation of camera. c. Yolov5 model should recognize mosquito with different species, pose and size at different distances.	1. Verification for Yolov5 model a. make 10 attempts to test mosquito and other items to make sure that the model gives a difference more than 0.5 between the confidence of mosquito and other items. b. Move the camera slowly and make sure the model can continuously recognize the mosquito successfully. c. Use mosquito with different colors and size and stick it to the test board with different angle. Make sure in most cases the model can recognize them successfully.	Y
2. Requirements for edge detection model a. When there is not laser point in camera image, output laser point coordinate of (-1, -1) to mark there is no laser point. b. If exists red laser point in camera image, mark it with a red circle and output the correct laser point coordinate. c. Even there is two detected red laser point, only output one's coordinate with highest area lightness.	2. Verification for edge detection model d. When there is no red laser point, make sure that only coordinate of (-1, -1) will be output as laser point position. e. Use one red laser point, make sure there is a red circle mark its position and output the correct coordinate of laser point. f. Use two red laser point and make sure only the one with lightest lightness will be marked as laser point.	Y
3. Requirements for camera & telescope a. When The camera should have a resolution higher than 1280*780. b. The camera with the telescope can focus on the mosquito successfully at 1.6m distance.	3. Verification for camera & telescope g. Connect the camera to the processor, and input one image. Check whether he resolution of the image is higher than 1280*780. h. Check that the mosquito at a distance of 1.6 meters can be clearly seen in the picture.	Y

4. Requirements for embedded development board a. The board needs to have several USB interfaces to receive the data from the camera, mouse, keyboard and the Bluetooth module. b. The processing rate should be higher than 5 frames per second to reduce delay. c. The operating system on the board should support the python execution and dependencies of yolov5 model. d. The board should have the GPIO pins supporting the serial communication.	4. Verification for Embedded Development Board i. The Orangepi 5 has three USB type-A interfaces and one USB type-C interface, sufficient for our use. j. We use some strategies and the NPU to accelerate the running. In our experiment, we measured the processing rate, which is higher than 5 frames per second. k. The environment on Orangepi has no significant difference with Ubuntu on PC. The python and dependencies of yolov5 are fully supported. l. There are 4 groups of Rx and Tx with UART protocol. Actually, one group is enough for us to use.	Y
5. Requirements for servo a. It can't shake violently while rotating. b. The speed must be slower than 0.032 rad/s (1.83 degrees/s). c. The max rotation angle of the horizontal servo should be 360 degrees. And the vertical one should be able to rotate at least 180 degrees.	5. Verification for servo m. Test it by sending signals to it and make sure it will not shake violently especially when the laser spot is close to the mosquito. n. Fix a mosquito on the wall and make sure it can be reached by the laser spot. o. Make sure servos can reach any direction in space.	Y
6. Requirements for remote control a. The application can run on mobile devices like Android phone. b. The application should have an understandable user interface, and useful features for remote control. c. The application should deal with the case when the communication is cut off by accident.	6. Verification for remote control p. The application is developed for Android platform and tested on serval devices. q. The application has beautiful GUI and clear instructions on each button. It provides both manual control and automatic control approaches. r. The application has the anti-triggering mechanism as mentioned before. When the communication is cut off, the whole system will stop running immediately.	Y

7. Requirements for Angles Calculation a. When the laser and the mosquito distance is far away, the laser close to the mosquito faster. b. When the laser is close to the mosquito, the speed of the laser approaching the mosquito is slow. c. When the laser is approximately at the same position as the mosquito, the laser stop moving.	7. Verification for Angles Calculation s. Move the laser and the mosquito within the scope of the camera and turn on the power. Keep them as far away from each other as possible and calculate the speed of the laser. t. Keep them close to each other and calculate the speed of the laser. u. Keep them the same position and make sure the laser will not move any more.	Y
8. Requirements for UART Serial Communication a. The Arduino can receive the message including three variables from Orange Pi. b. Make the servo receive the latest message as possible.	8. Verification for UART Serial Communication v. Send massage from Orange Pi and make sure the same message can be seen in the serial monitor of Arduino. w. Create a thread for sending message and a Global variable to store the latest message in Orange Pi so that Arduino can receive the latest message as possible.	Y
9. Requirements for Finite State Machine a. Make sure all four states can be reached and the reaching condition is the same as what we illustrated before.	9. Verification for Finite State Machine x. Send the reaching conditions to Arduino and check the states.	Y
10. Requirements for PWM a. Servos can rotate the angles sent from serial monitor. b. Arduino can control the Servo to rotate non-integer degrees.	10. Verification for PWM a. Input angles to serial monitor and check the real angles read from the servo. b. Use myservo.writeMicroseconds(time) to control the rotation angle of the servo and check the real angles read from the servo.	Y