Embedded Final Project Report

b02501093林渼芸
b02901150何佳錦

A. Abstruct

In 21th century, the topic of intelligence becomes very important. In this final project, we want to build a machine which combines Raspberry pi and Arduino together to let us use gesture to control the 3C product.

Raspberry pi will use Raspberry Pi Camera module to detect the gesture and it will also use its ability of calculation to distinguish different gesture. After calculating, it will send a signal to Arduino. Arduino will use this signal to decide which IR signal it has to send. In the end, the fan will turn on or do some function. Moreover, we use auto mode to sense the room's temperature and the fan will automatically wind up or down according to it.

In the future, we want to use this method as a reference to the Internet of things. By detecting the behavior of user, 3C product can be more useful and convenient for us to control. Not only bluetooth but also IR light are convenient ways to communicate with the 3C product. We try to build a easy prototype to fulfuill the dream about intelligent 3C product.

B. Motivation

Due to technology improvement, intelligence products become more popular. With simplifying the project, we want to use the 3C product we have and improve it with the modern technology. As we learned Raspberry Pi and Arduino in embedded systems class, we tried to combine Raspberry Pi with openCV and Arduino with the some modules to complete the more useful way to control the 3C product.

After discussing, we decided to use fans as our main object in our final project and improved it with the modern technology — which is media processing. Combining all things together, our final project is using gesture to control the fans.

C. Goal

As we mentioned above, we want to design a machine that can use gestures to control the fans. This work is not only required the use of openCV and IRremote skill but also needed the combination for Raspberry pi and Arduino which needs less delay and high efficiency.

D. Design

Predesign:

The first thing we think is to separate our final project with several parts:

(1) Camera on Raspberry Pi to receive the gesture
(2) Raspberry Pi receive the frames camera took and identify the gesture
(3) Raspberry Pi send a signal to Arduino
(4) Arduino receive a signal and control the fan
(5) Additional function: Auto mode

Camera on Raspberry pi to receive the gesture :
The gestures we define are:

(1) Hand up: Open / Close the fan
(2) Hand down: Shake the fan
(3) Hand left: wind Up
(4) Hand right: wind Down
(5) Flip the hand: Enter/ Off the auto mode

Raspberry identfy gesture via Raspberry Pi Camera Module:

At first, we need to idntify the region of our hand in the frame from camera. We can easily to obtain the region whose color are in skin's color range by transforming the pocture from RGB format into HAV format. However, there are many objects in this range, but not we want. We have to find the object representing hand within these and fiter others out. It's difficuilt to compare shapes of objects to that of hand. Thus, we set a threshold to filter out small objects, which we view as noise at background.

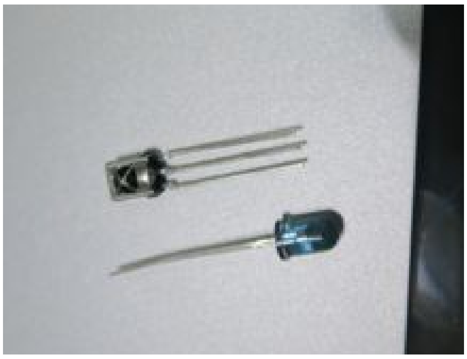
After finding the region of hand, we need to identify different gestures. We draw a rectangle, which contain the region of hand, with minimum area. Then, we find the center of the rectangle as the center of the region of hand. By the position of the center, we can track the hand and know whether it moves or not. Additionally, we use the area of the region of hand to identify whether the hand flips. Thus, we identify the gestures.

But for inaccurate identification, we add 2 LEDs to tell user the state the machine is. If there is difference between the machine and user expectation, user must adjust the position of the machine to revise the difference.

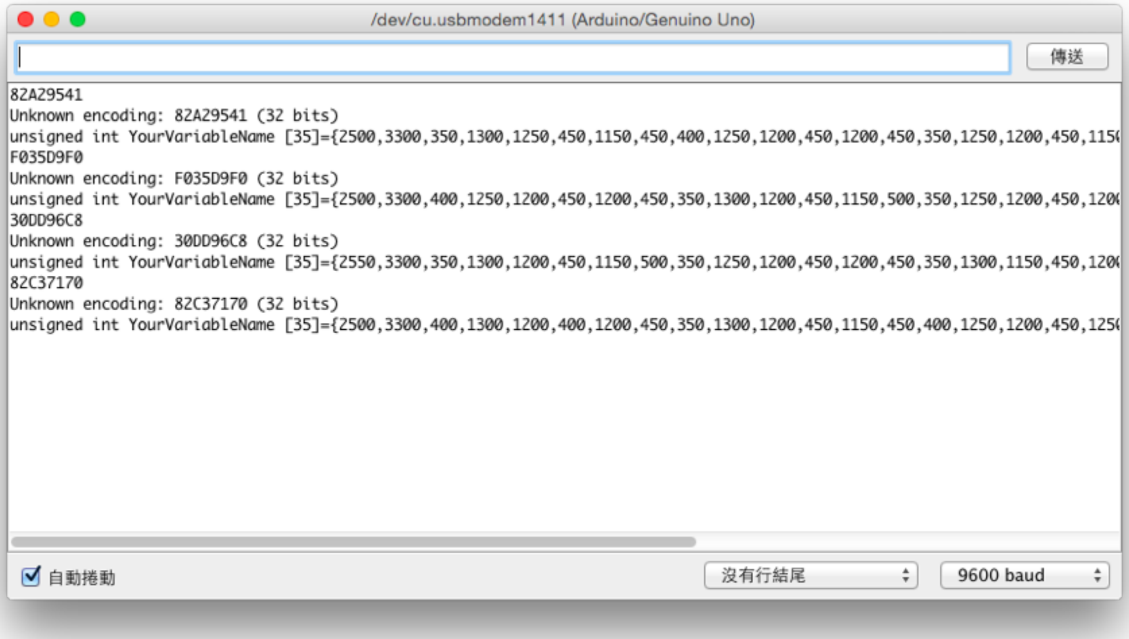
Arduino receive a signal and control the fan :

At first, we want to use bluetooth as the communication with the RPi, but this is too hard for project because we have to disassemble the fan and find the way it progress.

However, we find a substitute way which is easier — IRremote. IRremote use infrared light to communication with 3C products. Most 3C products use IR to control it, for example, fan, air conditioner and TV. IRremote can provide us a convenient way to control the fans without disassembling. This is the reason why we use Arduino. We buy some IR related modules , such as IR receiver and transmitter.



The first thing we have to do is recording the infrared signal which producer defined during manufacturing. We use IR receiver connecting with Arduino and press the button which remoter of fan has. We use the code imported IRremote libraries to record the infrared signal. This library will decode the IR signal, but we notice that it will fail when some protocols are too hard to decode.



```
82A29541
Unknown encoding: 82A29541 (32 bits)
unsigned int YourVariableName [35]={2500,3300,350,1300,1250,450,1150,450,400,1250,1200,450,1200,450,350,1250,1200,450,115
F035D9F0
Unknown encoding: F035D9F0 (32 bits)
unsigned int YourVariableName [35]={2500,3300,400,1250,1200,450,1200,450,350,1300,1200,450,1150,500,350,1250,1200,450,120
30DD96C8
Unknown encoding: 30DD96C8 (32 bits)
unsigned int YourVariableName [35]={2550,3300,350,1300,1200,450,1150,500,350,1250,1200,450,1200,450,350,1300,1150,450,120
82C37170
Unknown encoding: 82C37170 (32 bits)
unsigned int YourVariableName [35]={2500,3300,400,1300,1200,400,1200,450,350,1300,1200,450,1150,450,400,1250,1200,450,125
```

☑ 自動捲動 沒有行結尾 9600 baud

The second thing is transmitting the IR signal. We use the IRremote libraries to complete the task. IRremote libraries provide some protocol to transmit which will make code easier. However, it is not suitable for this case because the remoter we use don't apply to any protocol this libraries provide. Luckily, we have *sendRaw()* function which can help us transmit the IR signal.

The last thing is to make the Arduino process more effienct and intelligent and we will explain in the next part.

Additional function: Auto mode

Because we want our fan to be more intelligent, we design an auto mode to reach this goal. We add a temperature sensor to sense the room's temperature and transfer the data to Arduino to adjust the fan. The original base temperature we set is 25℃. The fan's wind will go up if temperature goes high, vice versa. The base temperature will change if the fan sense higher or lower temperature. In addition, it can be noted that we can control the open and close for fan when we are in auto mode.

Postdesign:

During the design, we find several points we have to deal with:

(1)The efficiency of IR transmitter

After predesign, we find that our original IR transmitter has little range for communication. We think it may be related to power of IR transmitter so we buy the one that is more expensive but has more power to transmit which work well at the end.

(2)How to connect the Raspberry pi and Arduino

We use I2C protocol for the communication, but we finally fail for this method. We think it may be related to the EMI inteference. This inteference will make I2C wire can't work well during transmitting. We guess that we can use pull-up resistor and capacitor to decrease the impact for it.(Because Serial port works really well, we didn't finish dealing with this problem.) In the end, we use Serial port, which is UART protocol. Serial port is dumber but more stable. The big difference between them is the generation of stop-bit which will prevent the receiver will keep listening when the transfer data is over. In addition, I2C will use callback functions for receiving, but Serial port use *loop()* as operation.

(3)Auto login & auto executing python code in Raspberry pi

For auto login, we degrade the init system to older version, and modify the content of */etc/inittab*. For auto executing Python code, we move our Python code to */usr/local/sbin/* , and add a script into */etc/init.d*.(Auto_execute in code file)

E. Open Source

a. RPi (code_on_rpi.py in code file)

   i. **picamera** Library to obtain frame in RGB format via picamera
      User can install this library via command, *sudo apt-get install python-picamera*.

   ii. **cv2** Library for image processing
      User can install this library via command, *sudo apt-get install libopencv-dev python-opencv.*

   iii. **numpy** Library to transform type of variables
      User can install this library via command, *sudo apt-get install prthon-numpy.*

   iv. **time** Library to delay

   v. **serial** Library to communicate with Arduino

   vi. **GPIO** Library to control LEDs

b. Arduino (code_on_arduino in code file)

   i. **IRremote** Library to control the IR communication
      Users can download it on https://github.com/z3t0/Arduino-IRremote
      Note: Arduino will have the default RobotIRremote library which will collide this IRremote one because they have same function name. The only thing you can do is removing the RobotIRremote library and change the path to the new one.

F. Result

We almost achieve the goal we setted up before . Our machine can filter out most of noise so that it doesn't interfered by some noise at the background. The correct rate of gesture controlling fan is more than 60%. Most of errors are caused from inaccurate gesture identification. The rate we can control the fan is about 1 instruction/sec, it's acceptable.

G. Difficuilities we faced

(1)For Raspberry pi:

→ We can't accurately identify gestures.

We set a threshold and filter out the noise by verifying whether their peremeter more than the threshold or not. There are some large noise we can’t filter out. Somtimes, they cause errors on the gesture identification.

We have tried some methods to eliminate the errors, like substrating two frames for diminishing backgound. However, they cause other problems. Thus, we didn’t apply these methods on our design.

(2)For Arduino:

→ We can't find the direct protocol for us to send the IR signal.

The IR remoter the fan use doesn’t belong any protocol exishted in the IRremote library. Luckily, it provide sendRaw() function to complement this difficulty.(but the efficiency will affect)

→ We can't listen and work for arduino at the same time.

In the beginning, we try to strengthen the ability for Arduino. However, Arduino is limited for its multiple working system. We can't listen the message for serial port and control the fan at the same time. Moreover, we can't use callback function for dealing with this one because Serial interrupt and callback interrupt will crash the Arduino.

At the end, we make Arduino simplier that Arduino only transmits the IR signal and the order it use are all from Raspberry pi.

(3)For the connection between Raspberry pi and Arduino:

→ We fail to use I2C as our protocol

Because of the EMI inteference the IR light produces, I2C fail at the end.(The reason we mentioned at the post design)We use Serial port as our communication gate.

H. Future Work

(1)Optimizing the program of gesture identification

As the difficuilty mentioned above, we want to look for other method to identify more accurately the gesture and improve its sensitivity.

(2)Humidity module on Arduino for auto-mode

In this project, we only use temperature sensor to help the fan automatically adjust its wind. However, we want to add more useful tools to increase the data for the fan to refer such as humidity. Moreover, if we change the fan to the air conditioner, we can try to sense the number of people in the room to change its operation.

(3)Creating a better way for demodulation

For IRremote Library, it works well for TV remoter and some fan remoter. It won’t have a good performance for air conditioner to operate. We think that this library doesn’t have an advanced way to demulate which we may finish it in our future work.