International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 7 ǁ July 2022 ǁ PP. 868-875

The Design of Collecting Shuttlecocks Robot Based on Object Detection

Changjiang Mo[*1], Zan Huang[2]
[*1]*Office of Scientific Research, Lingnan Normal University, Guangdong, China*
[2]*School of Department of Mechanical and Electrical Engineering, Lingnan Normal University, China*

Abstract
The aim of this paper is to illustrate the implementation of the system for collecting shuttlecocks by the automatic robot. This system, based on Jetson Nano and YoloV5, is consisted of five modules as follows: collection shuttlecocks mechanical module, motor controlling module, image recognition module, speech recognition module and Android App controlling module. The equipped software recognizes the shuttlecocks by image algorithm, then sends the calculates the coordination to micro-chip-controlled robot to execute the collection shuttlecocks action.

Keywords:*Image Process for irregular object, Delphi App, Jetson Nano embedded system, TCPIP connection.*

Date of Submission: 15-07-2022 Date of acceptance: 29-07-2022

I. INTRODUCTION

As an international-wide sport, playing badminton can not only exercise and enhance physical quality but also cultivate the spirit of tenacious struggle and excellent will quality. In the process of badminton training, especially in service practice, if no one picks up the shuttlecocks for a long time, there will be a large number of them scattered and stacked on one side of the court. These shuttlecocks need to be collected manually afterward. To speed up the collecting process, people often use the way with certain damage to the shuttlecocks, when picking up them, people make bending and upright movements quickly, these excessive movements may lead to lumbar muscle injury, especially for the elderly and people with lumbar injuries. In addition, the long-time collective work also makes badminton practitioners mentally tired, which would increase the visual movement reaction time of badminton players, resulting in low accuracy hitting performance[1]. Therefore, the manual shuttlecocks picking method is not only inefficient but also less flexible in our daily life or professional training scenes.

Nowadays, scholars have studied ball-picking robots, among which a team in Boston is famous for designing the world's first tennis-picking robot called “Tennibot”. The machine is divided into a visual module and a ball-picking module part. The visual module includes a fixed camera, which is used for positioning tennis balls; The ball-picking module is used for the robot movement and the pick-up ball movement.

The design[2] of collecting shuttlecocks devices mainly has two approaches: one is a manual operation and the other is an automatic machine using machine vision. There are two main ways to pick balls: manipulator grasping and vacuum suction[3]. There is low efficiency by the manipulator grasping and high-power consumption, causing huge dust and noise in the field by vacuum suction, which makes it difficult to massive production. Some studies use OpenCV[4] to locate the shuttlecocks, which are strongly dependent on brightness, background, etc.

In view of the above research status, this paper designs a collecting shuttlecocks robot, which uses a sweeping device to improve the efficiency of picking up the ball and avoid high costs with fewer conditions for pictures recognition. It has a certain cost-performance ratio.

II. PROPOSED SYSTEM

2.1 SystemArchitectureDesign

In order to utilize the visual tool to process shuttlecock image recognition, the embedded Linux-supported chip known as Jetson Nano[5] was adapted as core module. Jetson Nano with GPU(Graphics Processing Unit) module can deploy AI systems for terminal devices in many industries, including smart cities, smart factories, agriculture and robots. It has ultra-high computing performance, which can provide 472 GFLOPS computing performance for running modern AI algorithms. With powerful and efficient AI, computer vision and high-performance computing, it can innovate at the terminal, and the power consumption is only 5 to 10 watts. In this design, the image processing program based on the computer vision can be designed on its Linux operating system by connecting the CMOS camera IMX219 to it. At the same time, it has a built-in

wireless module, which can interact with mobile app through wireless connection. The diagram of the system design is shown in Figure 1, and Figure 2 illustrates the prototype for the application.

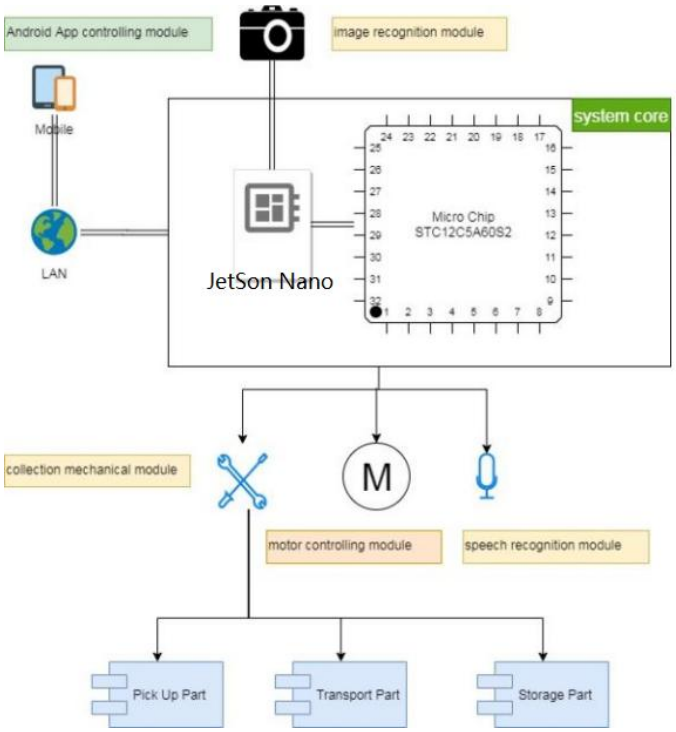

Figure 1: Diagram of System Design

The single chip micro-computer[6] or MCU(Micro Control Unit)(STC12C5A60S2), controls the collection mechanical module which is consisted of three parts mainly, there are pick-up part, transport-part, and storage-part. The pick-up part is made of a rollable plate with soft burr edges, the transport-part is composed of a conveyor belt and a motor, which is responsible for conveying the shuttlecocks to storage-part. Four cylinders load shuttlecocks one by one when one is full.

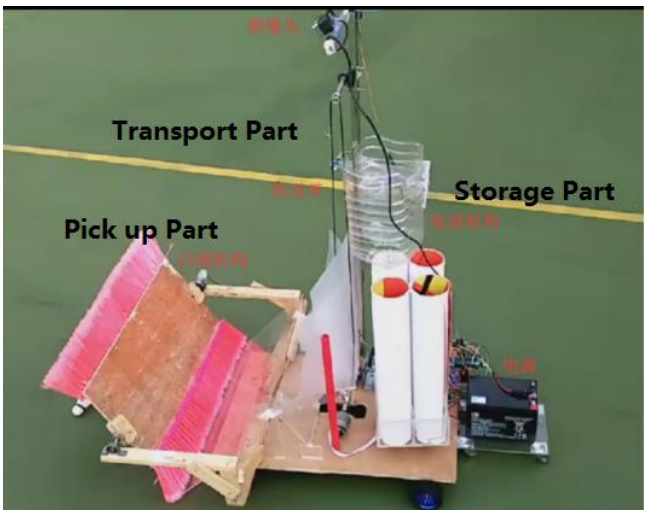

Figure 2: System Hardware Prototype

2.2 System Hardware Design

Since the motors need 12V voltage to drive, and Jetson Nano needs 5v2a for normal operation, 12v12ah battery is used for power supply. The voltage is regulated to about 8V through LM2596 step-down module and then regulated to 5V through the Jetson Nano extension board for itself and the MCU.

There are 4 motors driven by 12V, so two L298N motor drive modules are used to connect the MCU with the motors. The MCU can control the motor by changing the level state of IO port. The overall wiring diagram of the system is shown in Figure 3.

As is shown in the figure, the MCU is responsible for connecting the hardware control modules, such as motor driving, voice recognition chip input, infrared input, steering motor control for collecting by cylinder rotating, and so on.

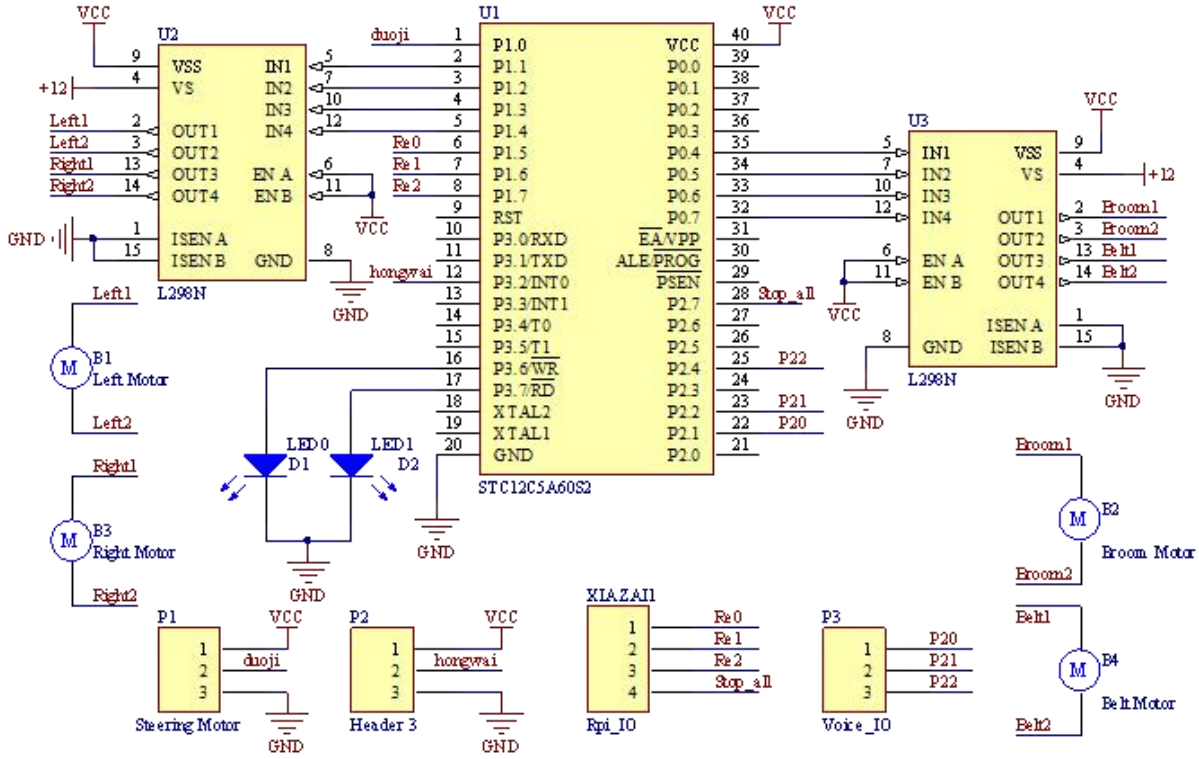

Figure 3: Circuit Design for Core Control Module

This design uses the LD3320[7] voice chip to match the input voice with the keywords in the keyword list after spectrum analysis and feature extraction and find the word with the highest recognition as the output result. The output results are sent to the MCU in the form of digital voltage level. The commands and outputs are shown in Table 1 below

Table 1: Voice Commands Output Protocol.

ID	Voice command	Coding Output (P20,P21,P22)
1	drive	001
2	forward	010
3	backward	011
4	turn left	100
5	turn right	101
6	open broom and conveyor belt	110
7	stop	111

2.3 System Software Design

The control system is mainly composed of motor drive control, speech recognition processing, image recognition shuttlecock, and mobile app control. The system control flow chart is shown in Figure 4.

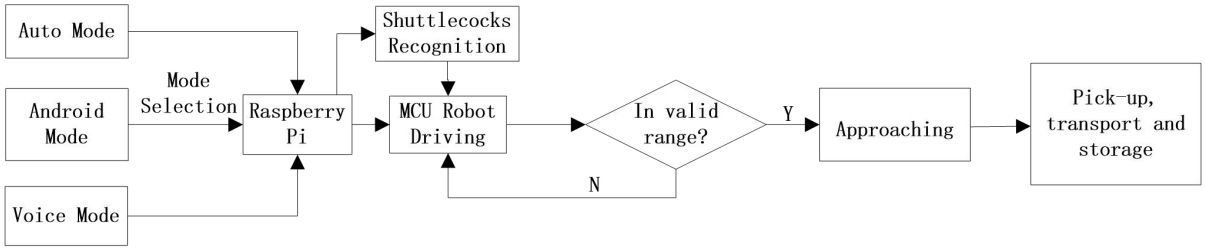

Figure 4: System Control Flow Chart

When the MCU detects the level signal sent by the Jetson Nano and voice chip, it drives the LM298 module to drive the motors by changing the p2.1, p2.2, and P2.4 pin states. After the infrared detection module detects the arrival of the set number of shuttlecocks, the MCU directly controls the motors through the corresponding pin signal. The flow chart of the MCU control is shown in Figure 5.

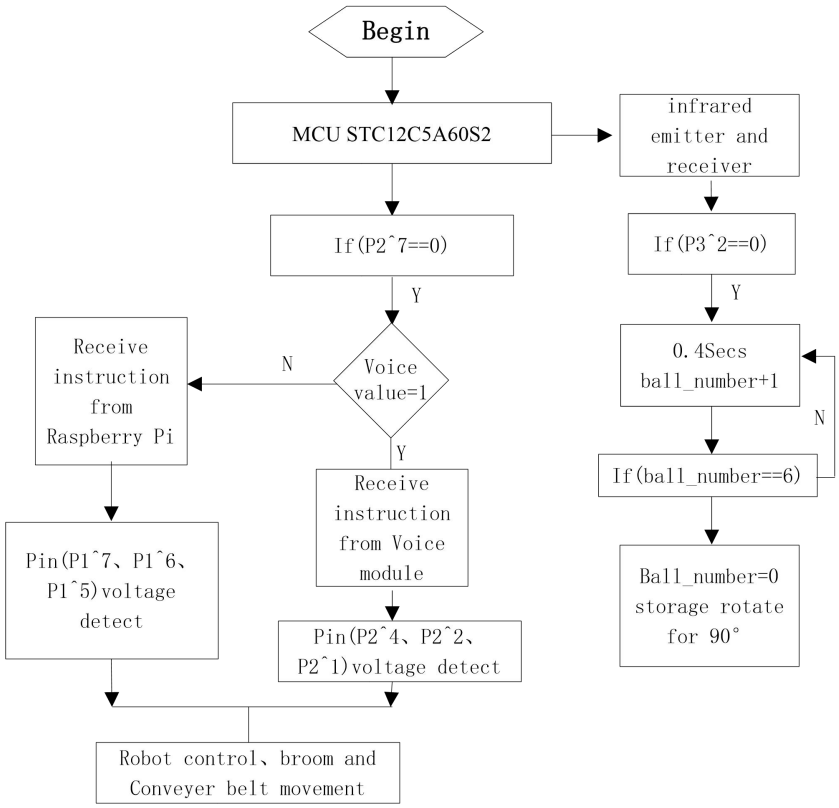

Figure 5: Flow Chart of MCU Control

2.4 Shuttlecocks Recognition Algorithm

Different from regular round balls, shuttlecock recognition should use another approach. To overcome the disadvantages of locating common round shape balls in normal ball-picking systems, an object detection[8] algorithm based on computer vision and machine learning is proposed.

YOLO[9] an acronym for 'You only look once', is the name of an object detection algorithm based on ML. It redefines object detection as a regression problem, applies a single convolutional neural network (CNN) to the whole image, divides the image into grids, and predicts the class probability and boundary box of each grid. Yolo algorithm is very fast for the detection problem is a regression problem, there is no need for complex pipelines. It is 1000 times faster than "r-cnn" and 100 times faster than "fast r-cnn". Yolov5 is the latest version of Yolo. YOLOv5 also has been packaged with all dependencies for the environments including CUDA/CUDNN, Python[10] and PyTorch.

Step 1: Using LabelImg software as shown in Figure 6 to train custom data sets. The training pictures should include shuttlecocks with different angles, brightness and backgrounds to meet the recognition requirements under various conditions in real-time working status. In LabelImg, it defined the "shuttlecock" category as its label, used the mouse box to select the shuttlecocks' area in pictures, and completed the label

marking in a batch of pictures. In this step, an XML custom label file is generated for single picture, which records the label name and four coordinates of xmin, ymin, xmax and ymax for its selection box.

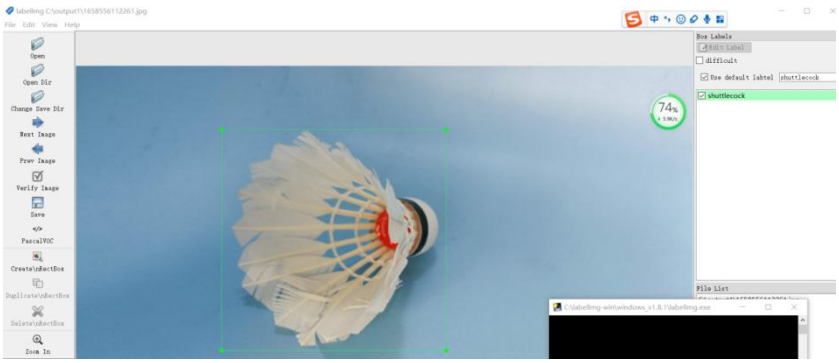

Figure 6: Custom Data-set annotation

Step 2: Organized the directory, and put the image-set and the label file generated in the first step into the specified directory of Yolov5 source code. Modified the 'coco128.yaml' file, and specified the number of classes=1, names: ['shuttlecock'] to correspond to the training data. Selected a pre-trained model to start training from. Here Yolov5s was selected, and a model with appropriate cost performance is available.

Step 3: Generated the training image-set, testing image-set files, verification image-set, and so on, generated annotation files containing label coordination by original images and corresponding XML files.

Partial source code for generating image sets:

```
trainval_percent = 0.2
train_percent = 0.8
xmlfilepath = './data/Annotations'
txtsavepath = './data/ImageSets'
......
for i in list:
    name = total_xml[i][:-4] + '\n'
    if i in trainval:
ftrainval.write(name)
        if i in train:
ftest.write(name)
        else:
fval.write(name)
    else:
ftrain.write(name)
```

Partial source code for generating annotation files:

```
sets = ['train', 'test','val']
classes = ["shuttlecock"]
......
for image_set in sets:
    if not os.path.exists('data/labels/'):
os.makedirs('data/labels/')
image_ids = open('data/ImageSets/%s.txt' % (image_set)).read().strip().split()
list_file = open('data/%s.txt' % (image_set), 'w')
    for image_id in image_ids:
list_file.write('data/images/%s.jpg\n' % (image_id))
convert_annotation(image_id)
list_file.close()
```

Step 4: Trained a YOLOv5s model on COCO128 by specifying dataset, confident threshold, epochs, batch-size, image size, and either pre-trained --weights yolov5s.pt. Pre-trained weights are auto-downloaded from the latest YOLOv5 release. After this step, the 'best.pt' weight file that can distinguish shuttlecocks was generated for real-time detection. The training result are illustrating in Figure7, the vital parameters, such as recall rate, precision, mAP@0.5, are in ideal performance.

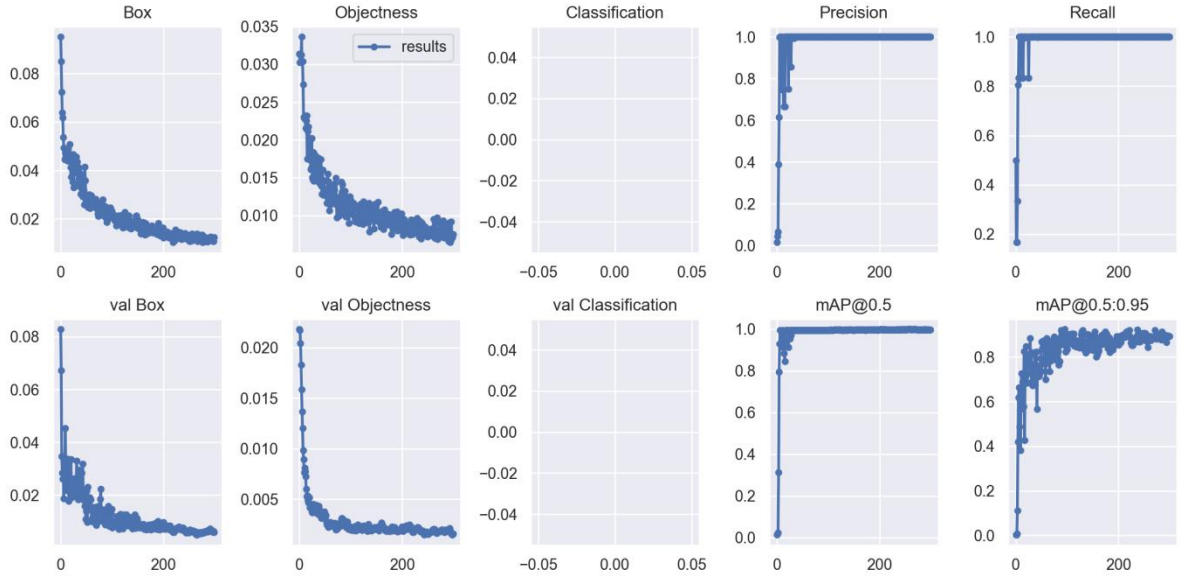

Figure 7: Training Results

Step 5: configured the yolov5 real-time detection program, specified the weight file generated in step 3, and specified that the image source as the real-time camera. Added the operation procedure after successfully identifying shuttlecocks. Once the shuttlecocks were identified in the real-time picture, calculated the size of the shuttlecocks, select the largest one, got the coordinates in the picture, and controlled the hardware to turn left and right and the rotation angle according to the coordinates. As shown in Figure 8, some shuttlecocks were located by blue box with confidence factor which ranging from 0 to 1.

Figure 8: Image Recognition Effect with Confidence Parameter

2.5 TCP Connection with App

Using socket to realize TCP communication, taking Jetson Nano as the server, and mobilizing phone or PC as the client to realize remote control. Some server code of Jetson Nano written by Python are as follows:

```
HOST_IP = "192.168.1.1"
HOST_PORT = 8000
socket_tcp=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host_addr = (HOST_IP, HOST_PORT)
addr:socket.bind(address)
socket_tcp.bind(host_addr)
socket_tcp.listen(1)
socket_con,(client_ip,client_port)=socket_tcp.accept()
```

The app program (TCP client) is written in Delphi[11], and the program flow chart is shown in Figure 9, shuttlecocks picking App was generated to Android 64bits shown in Figure 10.

The TCP Server(Jetson Nano) can be connected by multiple TCP Clients(Android App) at the same time. Specific instructions can be sent to the server through TCP communication. After receiving the corresponding action instructions, the server deploys the commands shown in Table 1 to the MCU for real-time control.

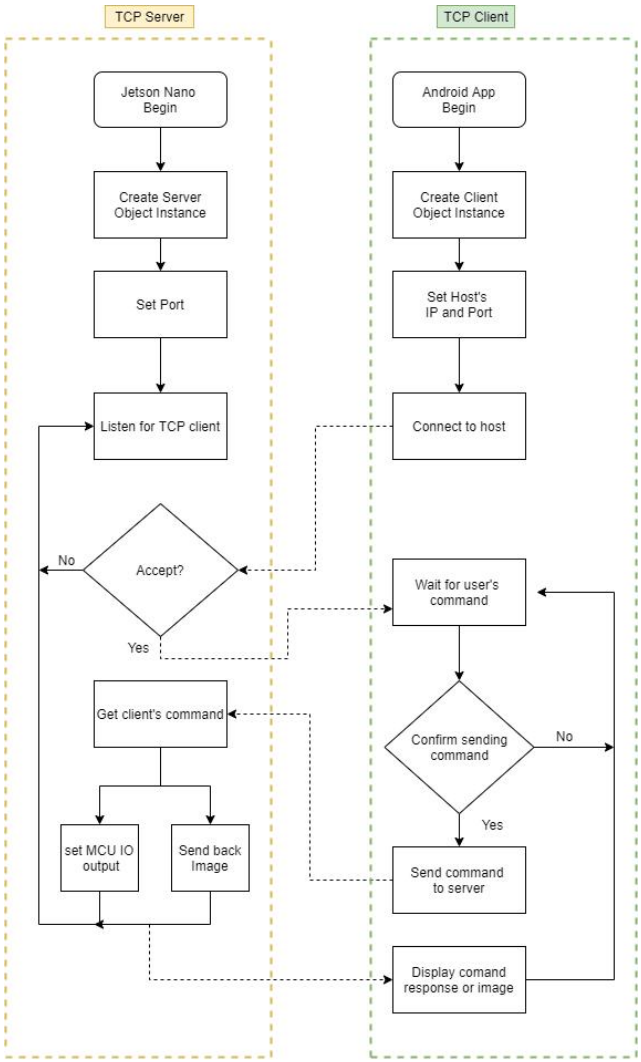

Figure 9: TCP program flow chart

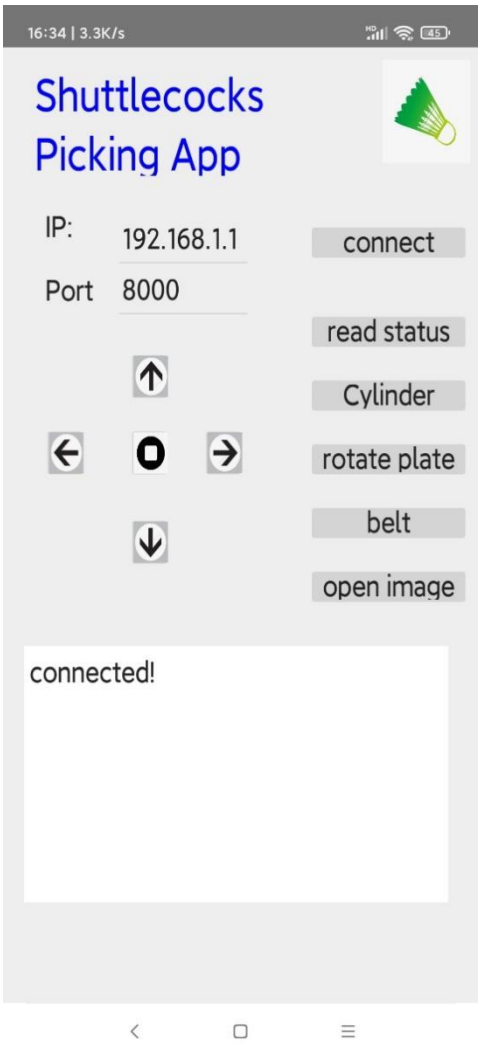

Figure 10: Android App

III. EXPERIMENT RESULT

Table 2 represents the experiment items result of the prototype system, after these experiments, all functions are verified its working performance.

Table 2: Experiment Items Performance

ID	Experiment item	Experiment result
1	Voice recognition	Command recognize normally within 50-60 DB and 3 meters away
2	Image recognition	Work normally within the 5-meter field of view of the camera
3	Endurance capability	Work normally for 2 hours under full battery power
4	Sweep and load the shuttlecock	The success rate is about 90%
5	App control	Control normally within a radius of 10 meters

IV. CONCLUSION

This paper designs a shuttlecock picking robot, which is tested in the badminton court environment. When the test field is green, blue and red, the software and hardware of the system can work well, and the recognition effect and shuttlecock-picking rate are high. Both the voice module and the remote communication module can operate normally, and achieve the expected design goal of the system. The system has a high-cost performance and practicality to work in wide-range brightness and complex backgrounds.

REFERENCES

[1] T. Hülsdünker, H. K. Strüder, and A. Mierau, 'Neural Correlates of Expert Visuomotor Performance in Badminton Players', *Medicine & Science in Sports & Exercise*, vol. 48, no. 11, pp. 2125–2134, Nov. 2016, doi: 10.1249/MSS.0000000000001010.

[2] K. Nicolaus, J. Hooper, R. Wood, and C. Ham, 'Development of an Autonomous Ball-Picking Robot', in *2016 International Conference on Collaboration Technologies and Systems (CTS)*, Orlando, FL, USA, Oct. 2016, pp. 373–378. doi: 10.1109/CTS.2016.0073.

[3] N. Pereira, F. Ribeiro, G. Lopes, D. Whitney, and J. Lino, 'Autonomous golf ball picking robot design and development', *Industrial Robot: An International Journal*, vol. 39, no. 6, pp. 541–550, Oct. 2012, doi: 10.1108/01439911211268660.

[4] T. Osiecki and S. Jankowski, 'Real-time detecting and tracking ball with OpenCV and Kinect', Wilga, Poland, Sep. 2016, p. 100315F. doi: 10.1117/12.2249849.

[5] H. M. Mohan, S. Anitha, R. Chai, and S. H. Ling, 'Edge Artificial Intelligence: Real-Time Noninvasive Technique for Vital Signs of Myocardial Infarction Recognition Using Jetson Nano', *Advances in Human-Computer Interaction*, vol. 2021, pp. 1–19, Aug. 2021, doi: 10.1155/2021/6483003.

[6] 'An App Visualization design based on IoT Self-diagnosis Micro Control Unit for car accident prevention', *KSII TIIS*, vol. 11, no. 2, Feb. 2017, doi: 10.3837/tiis.2017.02.020.

[7] L. Wang, 'Design of speech recognition system based on LD3320 chip', Taiyuan, China, 2016. doi: 10.2991/icmemtc-16.2016.33.

[8] K. Bhoomika, 'Development of Agribot System for Weed Detection and Automatic Spraying of Herbicide', vol. 11, no. 7, p. 4, 2022.

[9] C. Ye, Y. Wang, Y. Wang, and M. Tie, 'Steering angle prediction YOLOv5-based end-to-end adaptive neural network control for autonomous vehicles', *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 236, no. 9, pp. 1991–2011, Aug. 2022, doi: 10.1177/09544070211053677.

[10] Z. Jinnuo, S. B. Goyal, M. Tesfayohanis, and Y. Omar, 'Implementation of Artificial Intelligence Image Emotion Detection Mechanism Based on Python Architecture for Industry 4.0', *Journal of Nanomaterials*, vol. 2022, pp. 1–13, Jul. 2022, doi: 10.1155/2022/5293248.

[11] R. da S. Riquena, F. A. da Silva, F. V. Maracci, and M. A. Pazoti, 'FRAMEWORK PARA CONVERSÃO DE APLICATIVOS DELPHI DESKTOP EM APLICATIVOS ANDROID NATIVO', *ColloqExactarum*, vol. 6, no. 2, pp. 120–139, Aug. 2014, doi: 10.5747/ce.2014.v06.n2.e080.