International Journal of Advanced Engineering Research and Science (IJAERS)
Peer-Reviewed Journal
ISSN: 2349-6495(P) | 2456-1908(O)
Vol-12, Issue-6; Jun, 2025
Journal Home Page Available: https://ijaers.com/
Article DOI: https://dx.doi.org/10.22161/ijaers.126.3

Intelligent Cat Recognition and Feeding System Based on Raspberry Pi and OpenCV Vision Technology

Dong-Xu Pan, Hao-Lin Ye, Chih-Ying Chuang*

Guangdong-Taiwan College of Industrial Science & Technology, Dongguan University of Technology (DGUT), Dongguan 523000, China
*Corresponding author

Received: 18 Apr 2025,

Receive in revised form: 02 Jun 2025,

Accepted: 07 Jun 2025,

Available online: 12 Jun 2025

***Keywords*— Raspberry Pi 4B; cat face recognition; smart feeder; OpenCV**

***Abstract*— *To address the issue of feeding outdoor cats, this study designed an intelligent cat recognition and feeding system. In terms of hardware, it integrates a Raspberry Pi 4B with OpenCV to combine ultrasonic sensor cameras, servos, and pressure sensors, forming a dynamic monitoring, characteristic identification, and precise feeding operation process. In software, OpenCV is used for cat face recognition, and Python scripts are employed to coordinate the work of sensors and actuators. The response time of the system can be controlled to be below 4 seconds. In the accurate recognition and feeding management of stray cats, the effect is quite significant. The adoption of low-cost hardware and lightweight control strategies has endowed an operational approach that is both engineering innovative and socially meaningful, enhancing the efficiency of outdoor stray cat management and facilitating the search for lost pets.*

I. INTRODUCTION

China has over 50 million stray cats, with an annual increase of 12%, making their management a challenging issue for urban governance. Current traditional feeding methods are inefficient, have limited coverage, and lack individual identification and health monitoring capabilities. Most smart feeders on the market are designed for home use and suffer from inadequate waterproofing and accidental triggering issues [1][2], making them unsuitable for long-term deployment in outdoor complex environments. Edge computing devices, such as Raspberry Pi, combined with OpenCV's lightweight visual system offer a new solution for cat face recognition, already applied in agricultural monitoring and robot navigation [3][4]. However, existing technologies still face bottlenecks: traditional Haar cascade classifiers achieve less than 65% success rates at night [5], and ultrasonic sensors have distance measurement errors of ±15mm in rainy or foggy conditions [6], leading to increased false positives and affecting recognition accuracy and stability. Maintaining robustness in complex environments is the main challenge for cat face recognition technology, with early research mostly relying on Haar-like feature extraction techniques combined with cascaded classifiers for initial detection [7]. In facial recognition, Haar cascade classifiers are widely adopted due to their small size, but when used for cat face detection, the false alarm rate can reach as high as 18% [8]. Intelligent monitoring systems, leveraging Raspberry Pi's multi-threaded processing capabilities, can promptly perform image analysis, enhancing the level of intelligence in monitoring [9].

This study developed an outdoor intelligent feeding system with multi-sensor collaboration and adaptive algorithms. By improving the dynamic parameter adjustment strategy of Haar cascade classifiers [10], combined with multi-frame temporal verification and PID self-tuning control [11][12], it can enhance night recognition rates and reduce false triggering rates. The system interfaces with animal protection organization databases to establish electronic records for stray cats [13], and its single-machine cost is less than 400 RMB, which is a 60% reduction compared to traditional methods; moreover, it has both precise feeding and lost pet matching functions. With technological innovation and social welfare as dual advantages, it provides a replicable technical solution for ecological management in smart cities.

II. THE ALGORITHM PRINCIPLE AND SELECTION BASIS OF HAAR CASCADE CLASSIFIER

Haar Feature-based Cascade Classifier typically describes the rectangular differences of local image regions. By calculating the difference in the total gray-scale sum of pixels within a rectangular area, edges, lines, and changes in light and dark dynamics can be accurately detected [8]. In the task of detecting the bright and dark changes under the cat's eye, the Haar-like feature detection method, with the help of an integral graph, can obtain the total sum of pixel points from three small rectangular areas. This processing significantly reduces the process of reading each pixel individually, greatly enhancing the detection speed. The improved Haar feature detection method is able to more accurately identify specific image features and determine their locations.

(1) AdaBoost algorithm

The discriminative power of individual Haar features is weak, so a set of the most distinctive features are selected through the AdaBoost algorithm and combined into a strong classifier:

(a) The weak classifier, in which each Haar feature corresponds to a simple decision tree, such as threshold judgment.

(b) Feature selection, AdaBoost iteratively selects the Haar features that have the highest degree of distinction between positive and negative samples, that is, the features with the smallest classification error.

(c) Weight allocation, where in each iteration, misclassified samples are assigned higher weights, forcing subsequent weak classifiers to pay more attention to difficult samples.

(d) The strong classifier, which ultimately combines multiple weak classifiers into a single strong classifier by weighting them.

(2) Cascade structure

In the Viola-Jones object detection system, the multi-layer structure of the cascade classifier has been designed and improved to significantly reduce the computational load in the target detection process [8]. The system includes a series of consecutive execution stages. The AdaBoost algorithm at each stage fuses massive Haar-like features to generate strong classifiers, gradually increasing the detection rate requirements and lowering the acceptable threshold for false positives. In the lower stages, 1 to 10 highly distinctive features are used, resulting in a significant filtering rate for background areas; in the higher stages, the number of features increases to several hundred, allowing for detailed differentiation of complex candidate regions. In classic face detection scenarios, the cascade classifier can achieve a detection efficiency of up to 30 frames per second with the aid of 2000 to 6000 features. This remarkable detection performance makes the cascade classifier one of the classical models in the field of computer vision. After considering the design ideas and effects of the cascade classifier, it can be determined that the cascade classifier in the Viola-Jones system plays a key supporting role and influence on the development of modern object detection technology. The description of the multi-level classification, layer-by-layer filtering and early stopping mechanisms is as follows:

(a) A multi-level classifier, which consists of multiple strong classifiers cascaded together, usually with 10 to 20 layers.

(b) Layer-by-layer filtering, where each level of the classifier is responsible for quickly eliminating areas that are obviously not targets. Only areas that pass through the previous level will proceed to the next stage of detection.

(c) Early stopping mechanism, which means that if a region is rejected at any level, subsequent classifiers no longer process that area.

The selection criteria for Haar cascade classifiers can be explained from the following three aspects.

(1) Computational resource limitations

The memory usage of the pre-trained model haarcascade_frontalcatface_extended.xml is 12 MB, significantly lower than deep learning models. Tested on a Raspberry Pi 4B, its single-frame processing time is less than 0.8 seconds, fully meeting outdoor real-time requirements. Compared to MobileNet's 1.2-second processing time under the same conditions, the model's applicability in low computational power environments is particularly important.

(2)Balancing real-time performance and power consumption

Cascaded acceleration significantly reduces computational load by discarding non-target areas early on. In Raspberry Pi usage, cascaded acceleration keeps CPU load between 30% and 40%. Dynamic power management, with its low computational intensity, supports the Raspberry Pi's sleep-wake mechanism, thereby extending outdoor battery life.

(3) Data and training costs

Using OpenCV's pre-trained cat face classifier reduces reliance on large annotated datasets and eliminates the need for self-training models. With parameter adjustments, the Haar classifier achieves a 93% recognition rate during the day using a self-created dataset of just 500 samples. In comparison, deep learning models require thousands of labeled datasets for training.

(4) Limitations and mitigation strategies

The Haar cascade classifier significantly enhances recognition performance in this system from an academic perspective, but it has limitations in describing features. In cases of heavy occlusion and profile views, the recognition rate drops significantly. To reduce false positives, multi-frame consistency verification can be used, such as continuous detection over three frames. In dense vegetation and complex texture environments, misjudgments are likely; these can be filtered out using aspect ratio and area filtering to remove non-cat faces. In backlit and highly reflective scenarios, features lose their effectiveness. Dynamic adjustment of camera exposure settings and activation of infrared auxiliary lighting can enhance contrast.

III. IDENTIFICATION ALGORITHM IMPLEMENTATION AND ARCHITECTURE DESIGN

The implementation of the system's recognition algorithm is divided into the following five steps:

(1) Image preprocessing

After obtaining the original image, it first undergoes grayscale conversion, transforming RGB images into single-channel grayscale images to reduce computational dimensions. Subsequently, histogram equalization is performed to enhance image contrast and improve the clarity of low-light areas using global histogram equalization. In nighttime environments, Gaussian filtering is applied with a 5x5 kernel for smoothing to reduce noise. To prevent local overexposure, constrained contrast adaptive histogram equalization is also implemented.

(2)Multi-scale cat face detection

It is one of the crucial steps in the system's recognition algorithm, with the following detection sequence:

(a) Haar Classifier Initialization

Load the OpenCV pre-trained model haarcascade_frontalcatface_extend.xml, which is optimized for frontal cat face features and includes 20-level cascaded classifiers, with a minimum detection window of 64×64 pixels.

(b) Multi-scale Sliding Window Detection

The multi-scale target detection mechanism processes images using image pyramids and sliding windows. It reduces the image layer by layer with a scaling factor of scale Factor = 1.1, generating a 12-layer pyramid structure. For each pyramid level, it uses an 8-pixel step sliding window for detection, with the window size gradually increasing from 64×64 to match the image width. During the detection process, feature calculation and classification operations are performed sequentially. Each window calculates its Haar features and is evaluated through the cascaded classifier level by level. Only when a window passes all levels of the classifier can it be marked as a candidate box.

(c) Non-Maximum Suppression (NMS)

Merge candidate boxes with an Intersection over Union (IoU) greater than 60%, retaining only the highest confidence detection result and suppressing redundant boxes.

(3)Post-processing verification

It effectively reduces the false touch rate of identification. The order of processing and verification is as follows:

(a) Geometric constraint filtering

Remove candidate boxes with an area smaller than the threshold pixels, where this threshold can be set based on the size of a cat's face, and use aspect ratio validation to exclude non-feline animals such as birds and squirrels.

(b) Multi-frame temporal verification:

Use an interactive window mechanism to verify the consistency of detection results over three consecutive frames. The sliding window cache is designed to maintain a queue of length 3 to store the coordinates of the detection boxes from the most recent frames. If the target position offset is less than 15 pixels across three consecutive frames, it is considered a valid detection, and the feeding command is activated.

(4)Dynamic parameter optimization strategy

This strategy can be implemented in two ways:

(a) Night mode parameter adaptation

In night mode, the detection window is expanded, and the minimum detection size is adjusted from 64×64 pixels to 96×96 pixels to accommodate distant targets in low-resolution environments. Additionally, the classifier sensitivity is increased, and the scaling factor and merging threshold are adjusted to compensate for the loss of features in infrared images.

(b) Measures against rain and wind interference

The first challenge in rainy and windy conditions is focusing. Motion area focusing should be linked to the ultrasonic trigger position. Initially, scan an area of interest (ROI) with a radius of 50 pixels centered on the detection point. Then, perform a secondary enhancement on the ROI to optimize the distinguishability of features in rainy and foggy conditions, and execute local histogram equalization.

(5) Performance optimization design

By reusing integral images, computational speed can be effectively improved, with only one calculation required per frame for all Haar features to share. Additionally, utilizing the Raspberry Pi's quad-core CPU, the image can be divided into four sub-regions for parallel detection, significantly reducing processing time and achieving performance optimization.

Based on the aforementioned recognition algorithm, this study proposes the design of an intelligent cat identification and feeding system, detailing both hardware and software architectures.

(1) Hardware architecture design

The hardware architecture is divided into three parts: the perception layer, the execution layer, and the core controller. The perception layer includes ultrasonic sensors, camera modules, water level sensors, and gravity sensors. The execution layer comprises servo motor driver modules and warning modules. The core controller is a Raspberry Pi 4B, and Figure 1 illustrates the control relationships among all these hardware components.

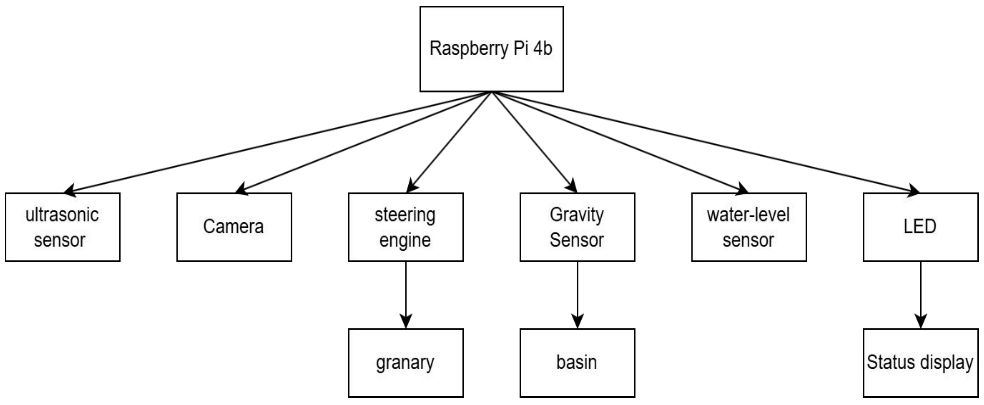


Fig.1 Hardware Structure Control Chart

The Raspberry Pi 4B serves as the core controller for the smart cat feeding system, leveraging its multiple interfaces, low power consumption, and efficient multi-threaded processing capabilities to achieve sensor data fusion and real-time image processing. Its open-source ecosystem and hardware expandability provide a technical foundation for intelligent stray cat management. The USB high-definition camera, as a key component of the cat face detection system, captures cat facial details with 1080P resolution (30 frames per second), is meeting the high-quality image requirements of the Haar cascade classifier. By connecting via the USB 3.0 interface to the Raspberry Pi 4B, data transfer speed is enhanced and latency is reduced, significantly optimizing detection efficiency.

The servo drive module acts as the core actuator of the smart cat feeding system, precisely controlling the opening angle of the food storage bin valve to achieve accurate feeding. The MG996R digital servo is selected for its adjustable torque design, which accommodates different weights of cat food, ensuring feeding precision and operational reliability. The gravity sensor serves as the core feedback unit of the smart cat feeding system, detecting the weight of the cat food in the storage bin in real time and employing closed-loop control methods to ensure precise feeding. The HC-SR04 ultrasonic module, as a key component of the perception layer of the smart cat feeding system, triggers recognition and initiates the feeding process upon detecting the proximity of a living being in real time. The water level sensor is a crucial component of the water supply system, used to monitor the water level in the tank in real time.

(2) Software architecture design

This design employs a layered architecture, consisting of a data acquisition layer, an algorithm processing layer, and a control logic layer. In the data acquisition layer, the Raspberry Pi system uses the RPi.GPIO library in Python to read real-time data from ultrasonic modules, camera modules, gravity sensors, and water level sensors. Image data capture is handled using the cv2 library functions of the camera module. The algorithm processing layer in the open-source OpenCV code repository includes callable Haar classifiers, which contain an XML file for detecting cat faces. This classifier has been pre-trained and can be used directly.

In the Raspberry Pi system, the control logic layer uses a dual-thread control method. One thread is responsible for monitoring the feeder and controlling the ultrasonic sensor, camera module, servo motor, and gravity sensor. It employs a finite state machine to manage the process of 'sleep-detect-identify-feed.' In most cases, the feeder remains dormant, with only the ultrasonic sensor active. Once the ultrasonic sensor detects a living being within 15 cm, the feeder transitions to the next phase, activating the camera and starting a 60-second timer. During this period, it captures images and determines if the subject is a cat. The servo motor and gravity sensor work together to precisely control the feeding process. If the identification result is a cat, the feeding task is completed; if not, the capture and identification process continues until the 60-second countdown ends, at which point the camera turns off and the feeder returns to sleep mode. The second thread monitors water levels, with an LED light status linked to the water level. When the water level is below the set threshold, the LED lights up red; when it's above or equal to the threshold, the LED lights up green. The flowchart is shown in Figure 2.

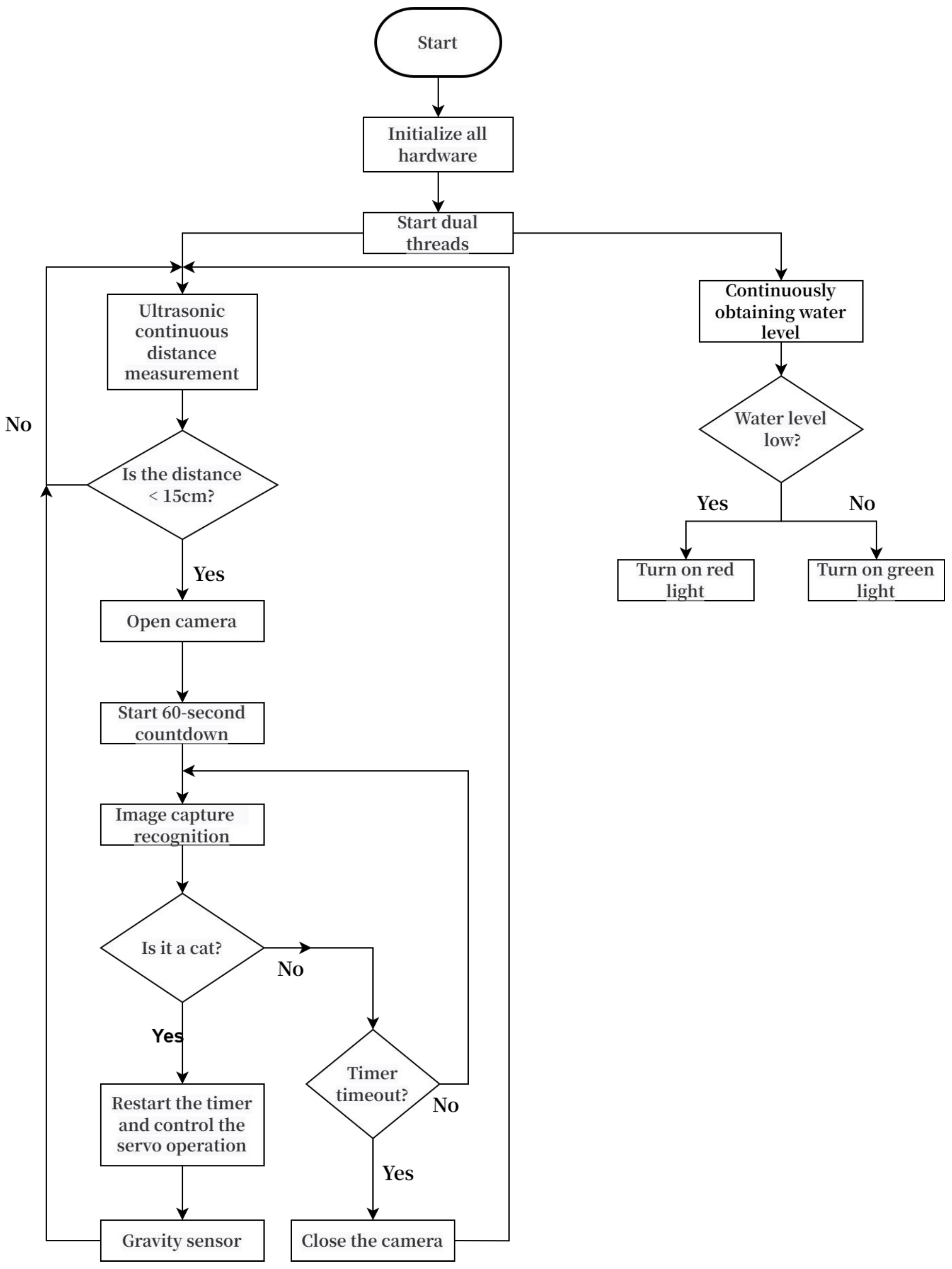


Fig.2 Software Core Control Flowcharts

IV. FEEDING SYSTEM DISPLAY

The demonstration of the feeding system can be illustrated through three aspects: physical integration display, control interface display, and core function implementation.

(1) Physical integration display

Figure 3 shows the finished intelligent cat recognition feeding system based on Raspberry Pi 4B and OpenCV, utilizing a modular design to integrate hardware components. The GPIO expansion board connects sensors and actuators, encased in a waterproof acrylic shell suitable for outdoor environments; the front is equipped with an HC-SR04 ultrasonic sensor, while a tilt-adjustable USB camera at the top enables multi-angle data collection. The food storage and water trough are fitted with gravity and water level sensors to ensure precise feeding and water monitoring. The system combines edge AI models and lightweight algorithms to enhance environmental adaptability and recognition accuracy, promoting the application of smart city technology for stray animal management, achieving the transition from laboratory prototypes to engineering products.

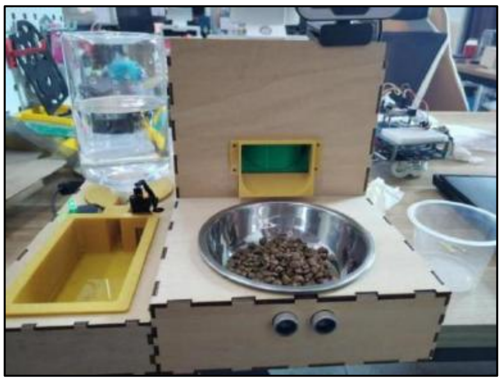
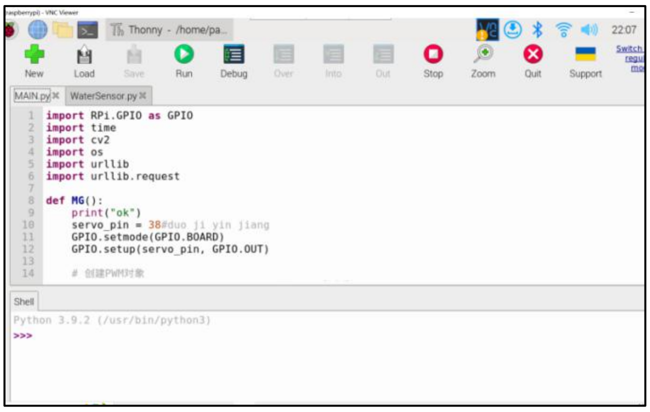


Fig.3 Intelligent Cat Identification and Feeding System Product Image

(2) Control interface display

Figure 4 shows the dual-thread logic control in a Raspberry Pi system with the environment set up. All setup processes are achieved through remote control, and the code is executed solely using the Thonny compiler that comes with the Raspberry Pi. This entire logic control system is implemented in Python, incorporating key code from various sensors and using simple algorithms to enable these sensors to work together.



```
import RPi.GPIO as GPIO
import time
import cv2
import os
import urllib
import urllib.request

def MG():
    print("ok")
    servo_pin = 38#duo ji yin jiang
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(servo_pin, GPIO.OUT)

    # 创建PWM对象
```

```
Python 3.9.2 (/usr/bin/python3)
>>>
```

Fig.4 Dual-thread Logic Control Interface

Thread one is for the cat detection feeder. After activation, the ultrasonic sensor performs distance detection to determine if there is any object near the feeder. The system is set with a threshold of 15 cm. If the detected distance is below this threshold, the camera is activated. Using OpenCV visual technology's recognition capabilities, it detects cat faces in real-world scenes, as shown in Figure 5. In the captured image from the camera, the red rectangular box precisely marks the detected cat face area. Here, the detected cat is a stray campus cat. The system accurately distinguishes and captures the facial features of the cat, excluding environmental factors in the background. This example demonstrates the system's practicality and reliability in complex scenarios, providing factual evidence for the scalable deployment of feeders in society. By improving the classifier with transfer learning, the identification accuracy under side profiles and occlusions is further enhanced. Once the cat is successfully identified, the system sends a signal to the servo module. Upon receiving the signal, the servo module activates and swings back and forth at 0°, 90°, and 180° angles to dispense food until the weight of the cat food reaches the predetermined threshold as measured by the gravity sensor.

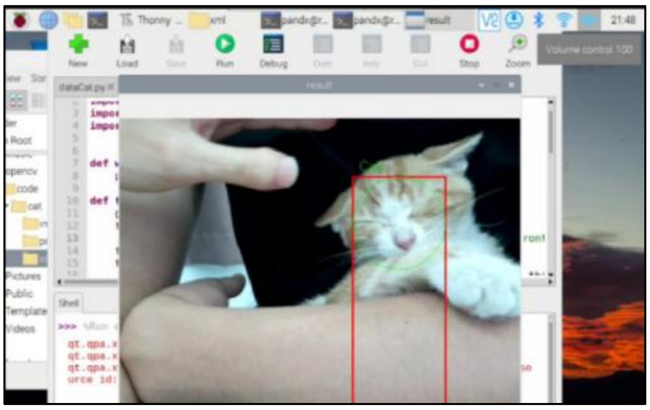
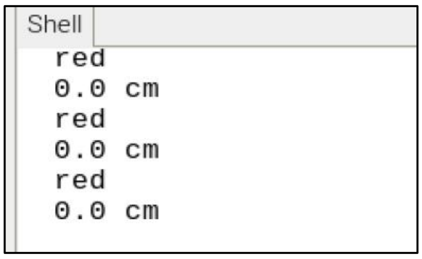
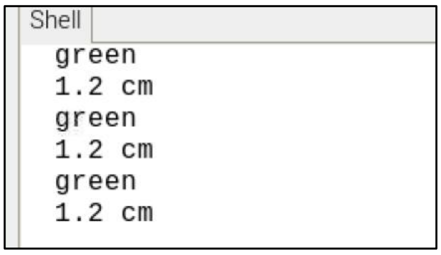


Fig.5 The System Detects Cat Faces in Real-world Scenes

Thread two is the water level monitoring system. When the detected water level falls below the preset safety threshold, as indicated by the monitoring window, the system immediately triggers an alarm mechanism, setting the LED indicator to red to alert the user and remind them to refill. When the water level is detected to be above the preset safety threshold, as shown in the monitoring window, the system has set the LED indicator to green, indicating that the alarm has been cleared. The threshold set for this system is 1 cm. The system's monitoring window is shown in Figure 6, and the tank light number is shown in Figure 7. This thread returns the current liquid level data in real-time through the serial communication protocol during the initialization phase, with a precision of millimeters. Its core algorithm uses sliding window filtering technology to eliminate interference from surface fluctuations, ensuring the stability of the detection results.



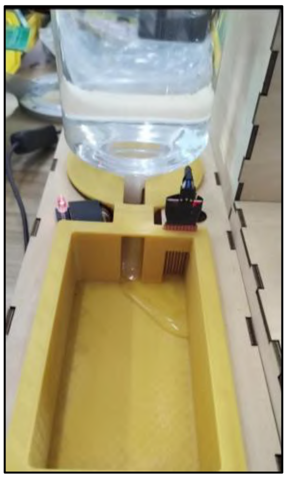
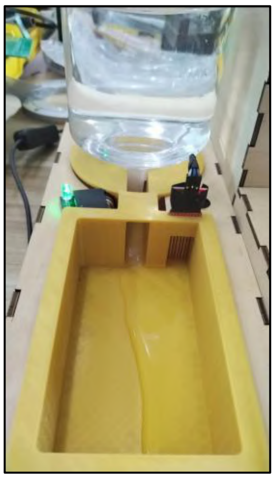
```
red
0.0 cm
red
0.0 cm
red
0.0 cm
```



```
green
1.2 cm
green
1.2 cm
green
1.2 cm
```

(a) Set the red light (b) Set the green light

Fig.6 System Monitoring Window (a) Set Red Light (b) Set Green Light

(a) Red light on (b) Green light on

Fig.7 Feeder Light Signal Display (a) Red light on (b) Green light on

(3) Implementation of core functions

The implementation of core functions is reflected in the following four aspects:

(a) Living being detection and triggering

The ultrasonic sensor continuously monitors the feeding area. When it detects the presence of cats, dogs, or other living beings within 15 cm, the count increases. Once the count reaches the threshold, the camera is activated within no more than 1 second. After the camera starts, the count resets to zero.

(b) Cat visual recognition:

After the camera captures real-time images, it uses OpenCV to detect cat faces, distinguishing them from other animals with an accuracy rate exceeding 90%. Processing time for a single-frame image is less than 2 seconds.

(c) Precise feeding control

Upon confirming the presence of a cat, the system activates the servo motor to open the food storage bin valve, releasing a precise amount of cat food. The error in each feeding session does not exceed 5 grams.

(d) Water level monitoring and alerting

The water level sensor continuously monitors the state of the water reservoir. When the water level is sufficient, the LED indicator light turns green. If the water level is low, the LED changes from green to red to remind the manager to refill the water.

V. CONCLUSION

This study designs an intelligent cat recognition and feeding system centered around a Raspberry Pi 4B. It integrates control through HC-SR04 ultrasonic sensors, MG996R digital servos, and HX711 load cells, combined with a dual-threaded cooperative control strategy to achieve fully automated processes of 'dynamic detection-feature recognition-precise feeding,' with a response time under 4 seconds and a cost kept within 400 RMB. To enhance performance, the use of integral image reuse, quad-core parallel computing, and image down sampling strategies has reduced single-frame processing time to 1.8 seconds, decreased memory usage by 40%, while also ensuring low energy consumption, low cost, and high scalability, providing a lightweight yet highly reliable technical solution for protecting stray cats outdoors.

Future improvement directions include:

(1) Algorithm focus: Considering federated learning combined with edge computing to enhance generalization capabilities, and adopting lightweight deep learning models to optimize recognition robustness.

(2) Hardware optimization: Enhancing battery life and performance, such as integrating millimeter-wave radar to reduce environmental interference, upgrading to Raspberry Pi 5 to boost computational power to support real-time detection algorithms like YOLOv5, and adding automatic aperture and moisture-proof designs.

(3) Ecological collaboration: Integrating with smart city platforms to combine GPS positioning with community rescue networks, promoting the social value of technology for good.

REFERENCES

[1] Wang, T. Design and Implementation of Campus Stray Cat Rescue Management System Based on Facial Recognition Technology. Zhejiang Normal University, 2023.

[2] Lee, H. C., Lee, H. W., and Kim. M. S. Development of IoT-Based Real-Time Fire Detection System Using Raspberry Pi and Fisheye Camera. Applied Sciences, 2023, 13(15):

[3] He, H., Jiang, Y., and Xiao, X. Remote Image Data Acquisition Based on OpenCV-Python and Raspberry Pi. Fujian Agricultural Machinery, 2022, (03): 27-30.

[4] Deng, S., Pan, J., Deng, Jun, et al. Automatic Tennis Ball Picking Robot Based on Raspberry Pi and OpenCV Vision Technology. Journal of Mechatronic Engineering Technology, 2024, 53(07): 60-63.

[5] Zhang, X., Liu, Y., and Yao, H. Cat Face Recognition Based on Haar-like Features Using Spatial Pyramid Model. Intelligent Computer and Applications, 2015, 5(04): 104-107.

[6] Shi, X, Wang, Z., and Sun, T. Design of a Rearward Ultrasonic Distance Measurement System Based on Microcontroller. Agricultural Equipment and Vehicle Engineering, 2024, 62(04):107-111.

[7] Zhang, X. Cat Face Detection Based on Candidate Generation. Harbin Institute of Technology, 2015.

[8] Wang, S. A Real-time Face Detection System Based on Skin Color Detection and Haar Cascade Classifier. Software, 2022, 43(07):125-127.

[9] Yu, W., Mo, J., and Huang, J. Design and Implementation of an Intelligent Surveillance Tracking Robot System Based on Raspberry Pi and OpenCV. Modern Computer (Professional Edition), 2018, (17): 80-84.

[10] Xu, X. Research on Cat Face Recognition Based on C++ and OpenCV. Computer and Information Technology, 2021, 29(03): 30-32+52.

[11] Liu, C., and Zhang, W. Design and Implementation of Face Recognition System Based on OpenCV. Modern Information Technology, 2024, 8(14): 20-25.

[12] Ding, Y., Guo, C., Luo, Y., et al. Research on PID self-tuning control based on recursive least squares parameter identification for the fast steering mirror-based optoelectronic tracking system. Sensors and Actuators: A. Physical, 2025, 386116323-116323.

[13] Zhang, M., Tang, H., Zhang, H., et al. Cloud-based Intelligent Classroom Attendance System Using Raspberry Pi and Facial Recognition. Microcontroller & Embedded Systems Applications, 2019, 19(01): 35-37.