AN INTERNET-OF-THINGS (IOT) SYSTEM TO AUTOMATE THE PET DOOR CONTROLLING USING ARTIFICIAL INTELLIGENCE AND COMPUTER VISION

Alex Lin[1] and Yu Sun[2]

[1]Arnold O. Beckman, 3588 Bryan Ave, Irvine CA 92602
[2]California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

Pets are cute and fun, but sometimes it gets annoying when you are trying to get some sleep in [14]. Also the population of people who would like to be accompanied by pets is rising rapidly. However, people these days lack time to meet all requirements of pets, which causes plenty of tragedies [3]. In this paper, we designed an application/tool called Automate the Pet Door Controlling which is using artificial intelligence and Computer vision to create a better living condition for pets, and a easier life for people who love their pets, especially for those who cannot spend a lot of time or physically hard to open doors for pets. My design builds an integratable system which is available for most smart home systems, also this design supports extra datasets imported to enlarge the types of pets.

KEYWORDS

IoT, Mobile, Machine Learning.

1. INTRODUCTION

Having a pet is fun and painful at the same time, because it bothers you sometimes when you do not want them to be there [15]. Yet they also get lonely when you lock them in one place. Yelling and breaking things is what they often do when they want your attention. Aside from that, keeping them away from danger is being responsible to them.

Most of the pet doors on the market are designed for dogs and large animals/pets. The issue with all the existing products is that using a pet collar is the key to exiting or entering the pet poor. If the battery of the pet collar ran out, pets wouldn't be able to enter or exit the door. There are also safety issues with this kind of automatic pet door; An intruder can easily access or break into your house if they gain access to the pet collar. They also take up a lot of space because it is using the method of sliding up and down. Most important issue of existing pet doors on the market is the price. Price points vary from $250 to $600, which makes a lot of pet owners not be able to own one.

The automatic pet door I designed is through artificial intelligence and computer vision [4]. It uses a camera as a detection tool to locate whether pets are staying in an assigned area. The door only opens when your pet is detected. Therefore, the problem of intruders wouldn't happen. Second, this design of pet door uses a flipper door, it goes both ways so pets can both enter or exit. Flipper door design takes up less space compared to the sliding door method. Lastly, the cost of this design is economically friendly, all the parts needed for this design are easily accessible to everyone.

In two applications, we demonstrate how the combination of Internet of Things and Artificial Intelligence techniques would increase the quality of living for both pets and the person who keeps them. First, the pet door is fully controlled by an intelligent system with Raspberry Pi and AWS server [5]. In this way, after the camera captures pictures, these pictures are sent to the AWS server and get recognized by trained AI, thus, the whole system will not require any physical assistance as long as it gets well installed. Second, this system is economically friendly, since we have analyzed the evaluation of any part to create this embedded system, we try to approach the most efficient cost for each component by comparing the popular productions in the market, and have the highest ratio of performance and price [6]. Additionally, those parts we use are easy to obtain and replace, so even if some exceptional situations happen, our automatic pet door could be maintained with much less effort and cost compared to other existing smart pet door productions. Therefore, we believe that our automatic pet door's functionality would be proved by a fully self-implementation system and an easily accessible components structure design.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the conclusion remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the tracking system, a few challenges have been identified as follows.

2.1. Building environment for Raspberry Pi

One challenge I face during the process of demo design is having trouble installing the environment on Raspberry Pi [7]. Without everything installed and up to date, our code would not be working. For example, torch is a library for Python that facilitates building deep learning projects [8]. Torch cannot be installed correctly because Raspberry Pi uses Raspbian, which is a special version of Debian, as its operating system [9]. Normal packages that are made for Linux or MacOs cannot be used for Raspbian.

2.2. Training AI

Using AI training was the second challenge during the design process. Finding good datasets was not easy. Even though cats are a popular pet around the world, there is a lot of data set online. Most of the data sets either have less amount of training because a normal computer doesn't have power to compute and process large datasets. When an AI model is used to train our AI, it will return a lot of outputs, and not all of them are useful output. Sorting all the data and determining whether it is useful or not will help with the accuracy of detection.

2.3. Testing cases

Test cases were the last challenge during the process of demo design. Manual testing takes up a lot of time and some mistakes are hard to find. First problem with the test case is not being able to recognize the cat when it appears in the camera frame. What I figure is that not having enough light will hardly affect the outcome. The second problem is that the program cannot recognize the cat's face is not facing the camera. The chance of the program recognizing when the cat is facing towards the camera is high. When the cat is facing the camera diagonally, the accuracy is lower. The program has a really low chance of recognizing the cat when it is completely facing backward.

3. Solution

This system is an object detection system that uses a camera to capture the environment and receive messages back to Raspberry Pi. The camera records video as the pet approaches a certain location [1]. All the information is sent to a server through Raspberry Pi. The server has a database which processes the video and sends its result to the application. In this application, you can set whether it is automatic, or manually open the pet door. Multiple settings can be controlled from the application: the detection time range in a day, how long it takes to close the door after each opening, and record how many times a door is open each day. From the application, the signal is sent back to the Raspberry Pi, and it would be able to control the door through its motor which is connected to a 3D-printed Door. The pet door is designed to be a secondary door of your room door [2].

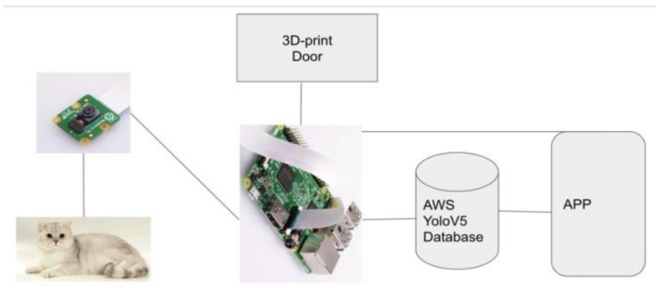
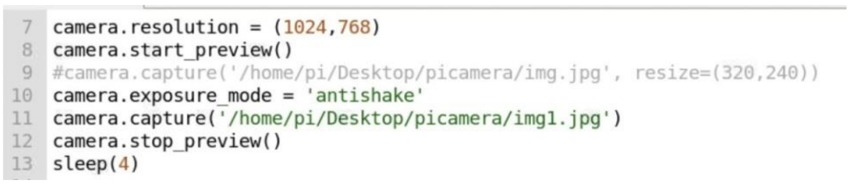


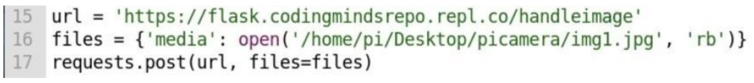
Figure 1. Overview of the whole system



```
camera.resolution = (1024,768)
camera.start_preview()
#camera.capture('/home/pi/Desktop/picamera/img.jpg', resize=(320,240))
camera.exposure_mode = 'antishake'
camera.capture('/home/pi/Desktop/picamera/img1.jpg')
camera.stop_preview()
sleep(4)
```

Figure 2. Code of camera

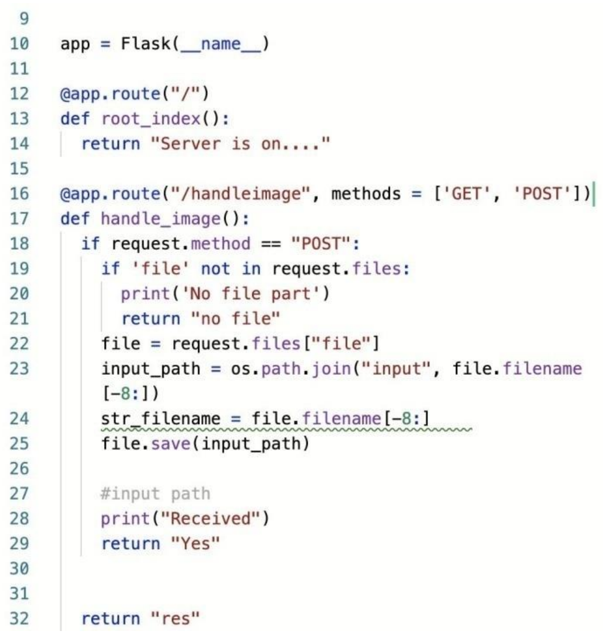
This section of the code is the setting for the camera. The resolution is set to the highest for better detecting results. Camera supposedly captures an image every four seconds.



```
url = 'https://flask.codingmindsrepo.repl.co/handleimage'
files = {'media': open('/home/pi/Desktop/picamera/img1.jpg', 'rb')}
requests.post(url, files=files)
```

Figure 3. Code of url, files, and request

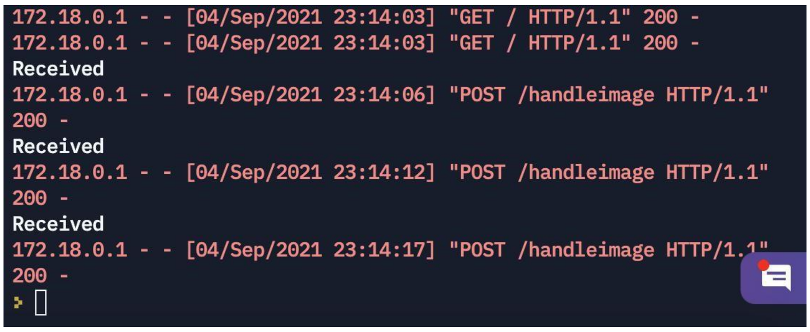
In figure 2, url is the link to the server; files is used to make the captured image into a file; I use a request to send the picture to the server.



```python

app = Flask(__name__)

@app.route("/")
def root_index():
    return "Server is on...."

@app.route("/handleimage", methods = ['GET', 'POST'])
def handle_image():
    if request.method == "POST":
        if 'file' not in request.files:
            print('No file part')
            return "no file"
        file = request.files["file"]
        input_path = os.path.join("input", file.filename
        [-8:])
        str_filename = file.filename[-8:]
        file.save(input_path)

        #input path
        print("Received")
        return "Yes"


    return "res"
```

Figure 4. Code of APP flask

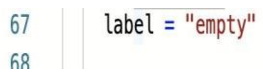
Use Flask to set up a server in order to receive the image sent from figure 3, and check if the image is available for the recognition process.



```
172.18.0.1 - - [04/Sep/2021 23:14:03] "GET / HTTP/1.1" 200 -
172.18.0.1 - - [04/Sep/2021 23:14:03] "GET / HTTP/1.1" 200 -
Received
172.18.0.1 - - [04/Sep/2021 23:14:06] "POST /handleimage HTTP/1.1"
200 -
Received
172.18.0.1 - - [04/Sep/2021 23:14:12] "POST /handleimage HTTP/1.1"
200 -
Received
172.18.0.1 - - [04/Sep/2021 23:14:17] "POST /handleimage HTTP/1.1"
200 -
>
```

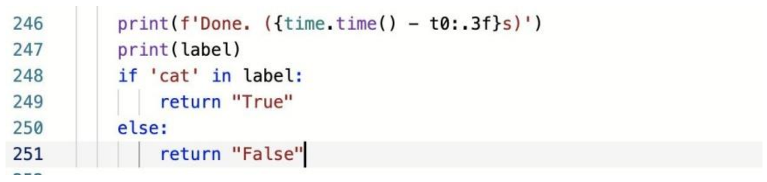
Figure 5. Screenshot of the server

This shows the server has successfully received the post images every 5-6 seconds.



```python
label = "empty"
```

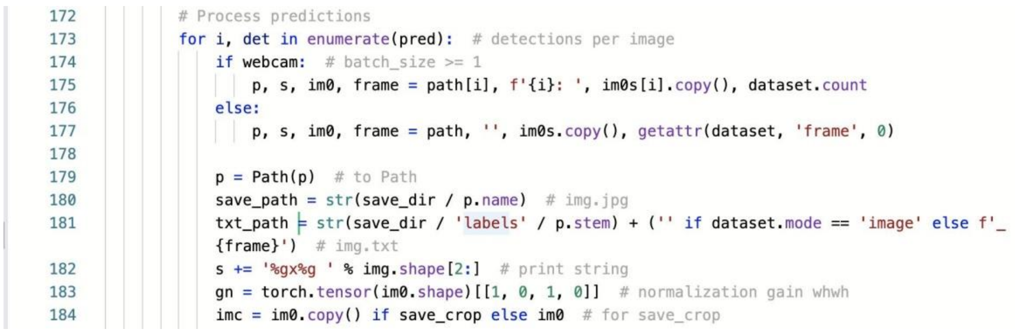
Figure 6. Code of label



```python
print(f'Done. ({time.time() - t0:.3f}s)')
print(label)
if 'cat' in label:
    return "True"
else:
    return "False"
```

Figure 7. Code of print

Variable label is used to determine whether the cat showed up in the assigned area. Variable label contains two perimeters: one is cat or empty and the other one is accuracy. If a cat is detected, it will return “True” and proceed to the next step which is sending a signal to the application.



```python
        # Process predictions
        for i, det in enumerate(pred):  # detections per image
            if webcam:  # batch_size >= 1
                p, s, im0, frame = path[i], f'{i}: ', im0s[i].copy(), dataset.count
            else:
                p, s, im0, frame = path, '', im0s.copy(), getattr(dataset, 'frame', 0)

            p = Path(p)  # to Path
            save_path = str(save_dir / p.name)  # img.jpg
            txt_path = str(save_dir / 'labels' / p.stem) + ('' if dataset.mode == 'image' else f'_
            {frame}')  # img.txt
            s += '%gx%g ' % img.shape[2:]  # print string
            gn = torch.tensor(im0.shape)[[1, 0, 1, 0]]  # normalization gain whwh
            imc = im0.copy() if save_crop else im0  # for save_crop
```

Figure 8. Code of process predictions

Through AI training, it processes the input image to determine whether the image captured contains a cat or not, by using the certain data-set machine learning model.

4. EXPERIMENT

4.1. Experiment 1

After completing the demo, I used my cat to test out the accuracy. The experiment was set up in two different locations, one with black wood floor and one with white wood floor. Each scenario was tested 50 times. The test participant was a white British shorthair. This experiment is designed to solve the problem with how lighting and background color affect the result.

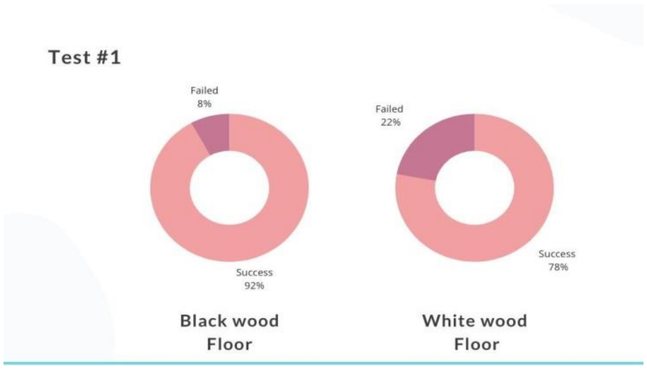


Figure 9. Test #1 result

From the graph above, we can see that a white British shorthair has a higher passing rate on the black wood floor and a lower rate on the white wood floor. The average is 85%. The result shows that the color of the background does affect the accuracy. The color back has a good contrast with the color white, which makes the result more accurate. Same color cat and floor would affect the accuracy because the cat blends into the color of the floor, which makes it harder to detect.

4.2. Experiment 2

In the second experiment, two more different kinds of cat become the participants. One of the participants is a gray/white persian cat. The second participant is a mixed Persian-American gray shorthair cat. Both participants are tested on both black and white wood floors. This experiment is designed to show how a different kind and color of cat will affect the accuracy of the result. Gray hair color cat can contrast with both black and white floor, it won’t blend into either of them.

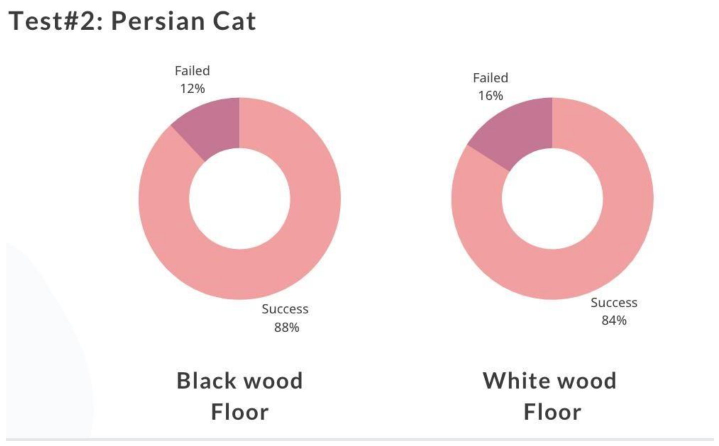


Figure 10. Test #2 result (1)

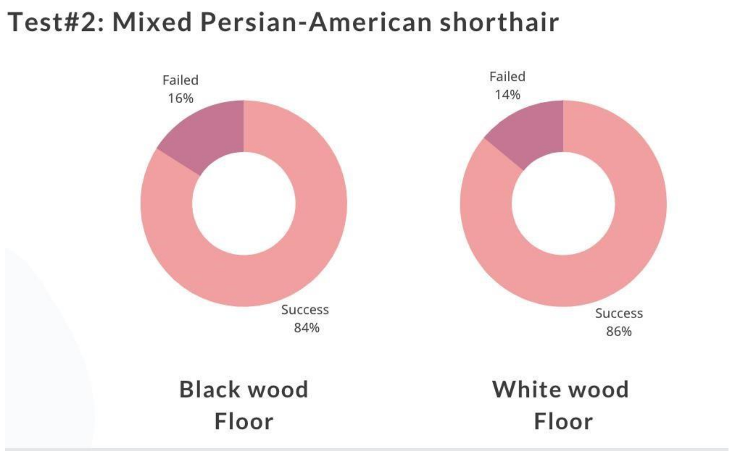


Figure 11. Test #2 result (2)

The graph above shows an average of 86% passing rate and 85% passing rate for Persian cat and the mixed cat. The bread of a cat does not affect the accuracy of the result. The main reason that affects the accuracy is the color of background and the color of the cat. If the background and cat have a similar color, the cat will blend into the environment. Therefore, the result will be affected.

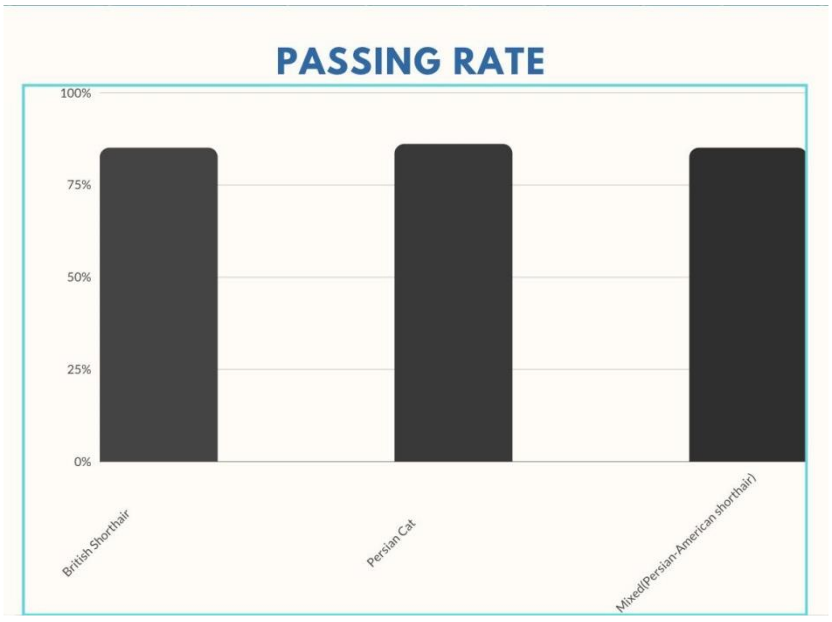


Figure 12. Compare two experiment

The result of two experiments shows an average passing rate of 85.3%. The rate of passing of British shorthair and the mixed is 85%. The rate of passing is 86% for the Persian cat. The final result is consistent and the passing rate between different breads/colors of the cat does not affect the accuracy. The error is in between 1%.

5. Related Work

Raza and Syed mention that camera calibration is important for collecting more precise data [11]. Three dimensional data is harder to collect because most of the cameras are designed to collect two dimensional images. Computer vision system won't function properly if the three dimensional image can't be captured correctly.

In Artificial Intelligence based Camera Calibration, writers stress that database learning is the principle of maximum entropy to produce answers [11]. The expectation is to be more and more accurate than those previous samples. Experiment should be on a very large dataset.

Bergström tests the detector positron camera by integral transformation of projections [10]. It used multiple sets of conditions to get a more accurate result. Control groups would get you a more accurate result and avoid some unexpected errors.

6. Conclusions

The pet door uses artificial intelligence and computer vision to process the image or video into a signal which opens and closes the door [12]. Pets can often be distracting when you are trying to focus on something. The pet door is designed for your pets to enter and exit safely. Most of the pet doors on the market use a pet collar to enter and exit the door. The collar can run out of battery or someone with it can access the pet door easily which will cause safety issues to your pet and you. In this case, a camera operated door would be much safer compared to the regular pet doors they have on the market.

The current limitation of the product is the accuracy when the color of the pet is similar to the background color [13]. If there is a low contrast between the two, the accuracy would be 10% to 20% lower than before. It is hard for the pet door to process smoothly when the internet signal doesn't have a strong connection. Can't control the door manually.

I plan to solve the limitation of accuracy by running more datasets through AI training. Controlling the door manually through an application which allows you to keep track of your pet.

It notifices if your pet has exited the door.

REFERENCES

[1] S. N. Raza, H. Raza urRehman, S. G. Lee and G. Sang Choi, "Artificial Intelligence based Camera Calibration," 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), 2019, pp. 1564-1569, doi: 10.1109/IWCMC.2019.8766666.

[2] Bergström, M.; Eriksson, L.; Bohm, C.; Blomqvist, G.; Litton, J. Correction for Scattered Radiation in a Ring Detector Positron Camera by Integral Transformation of the Projections, Journal of Computer Assisted Tomography: February 1983 - Volume 7 - Issue 1 - p 42-50

[3] Fennell, Lee Anne. "Common interest tragedies." Nw. UL Rev. 98 (2003): 907.

[4] Simon, Herbert A. The sciences of the artificial. MIT press, 2019.

[5] Upton, Eben, and Gareth Halfacree. Raspberry Pi user guide. John Wiley & Sons, 2014.

[6] Lal, Thomas Navin, et al. "Embedded methods." Feature extraction. Springer, Berlin, Heidelberg, 2006. 137-165.

[7] Richardson, Matt, and Shawn Wallace. Getting started with raspberry PI. " O'Reilly Media, Inc.", 2012.

[8] Van Rossum, Guido, and Fred L. Drake Jr. Python tutorial. Vol. 620. Amsterdam: Centrum voor Wiskunde en Informatica, 1995.

[9] Stonebraker, Michael. "Operating system support for database management." Communications of the ACM 24.7 (1981): 412-418.

[10] Bergström, M., et al. "Correction for scattered radiation in a ring detector positron camera by integral transformation of the projections." Journal of computer assisted tomography 7.1 (1983): 42-50.

[11] Raza, Syed Navid, et al. "Artificial intelligence based camera calibration." 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC). IEEE, 2019.

[12] Russell, Stuart, and Peter Norvig. "Artificial intelligence: a modern approach." (2002).

[13] Diebold, Francis X., and Robert S. Mariano. "Comparing predictive accuracy." Journal of Business & economic statistics 20.1 (2002): 134-144.

[14] Koivusilta, Leena K., and Ansa Ojanlatva. "To have or not to have a pet for better health?." PLoS One 1.1 (2006): e109.

[15] Reis, Marta, et al. "Does having a pet make a difference? Highlights from the HBSC Portuguese study." European Journal of Developmental Psychology 15.5 (2018): 548-564.