Smart Squirrel Proof Bird Feeder

By

Christine Li

Linfei Jing

Yitian Xue

Final Report for ECE 445, Senior Design, Spring 2021

TA: Shaoyu Meng

1 May 2021

Project No. 10

Abstract

The project centers on the design and make of a bird feeder. The bird feeder aims to increase the ability to prevent squirrels from stealing bird food. Compare with the regular squirrel proof bird feeder, we design the smart squirrel proof bird feeder to actively repel squirrels and load food only to birds. The design also increase the interaction between the bird feeder and users. This report describes the design, components, and building process. This also is a guide for someone who wants to make a smart squirrel proof bird feeder by themselves.

Contents

1 Introduction

Bird feeders are indispensable for the bird lovers to attract birds to visit their backyard, but they are also one of the main food sources that attract squirrels. Squirrels are the "enemy" of backyard birders. Even though squirrels are cute, bird lovers detest them to appear in their gardens. They do not just want to share the bird food, but they can take all of it. Their amazing athletic ability, voracious appetite and ability to chew through almost everything makes them unstoppable in the backyard. According to the National Pest Management Association, squirrels are considered pests. They are threats to the electrical wires and telephone lines outdoors. They can also invade houses for food and warm shelters [1]. The main goal of this project is to keep squirrels away from bird feeders and provide a peaceful and enjoyable environment in the backyards for the birders.

To solve this problem, we plan to build a smart bird feeder with a camera compatible with a web application. The camera will first distinguish bird by object detection model. After the identification, the feeder machine will either automatically load a reasonable amount of bird food or start the repelling system to decrease squirrels' interests to steal food. Researcher Jackson and his fellows from University of Toledo, Ohio, claimed in their study that their study objects, two fox squirrels have a hearing range between 113 Hz to 49 kHz at a level of 60 dB sound-pressure level [2]. Ultrasonic also works as an alarm call among squirrels by another study in University of Manitoba, Canada [3]. By comparison, human and birds have a similar hearing range. Both are most sensitive from 1 kHz to 4 kHz. Therefore, ultrasonic does not have any effect on birds.

The smart squirrel proof bird feeder contains six subsystems (see figure 1). The microcontroller determines to start the repel module or to load food to birds by the outputs of the object detection module. The object detection module must perform real time (less than $500ms$) detection to distinguish squirrels and birds. The feeder machine needs to load approximately $10g$ bird food each time, and repel system should be able to produce ultrasonic sound to repel squirrels. In addition, when the food is almost eaten up, the feeder will notify people to refill the food on the web application. To please the backyard birders, we also create an additional feature of taking birds' pictures when birds are eating in front of the bird feeder.

2 Design

2.1 Block Diagram

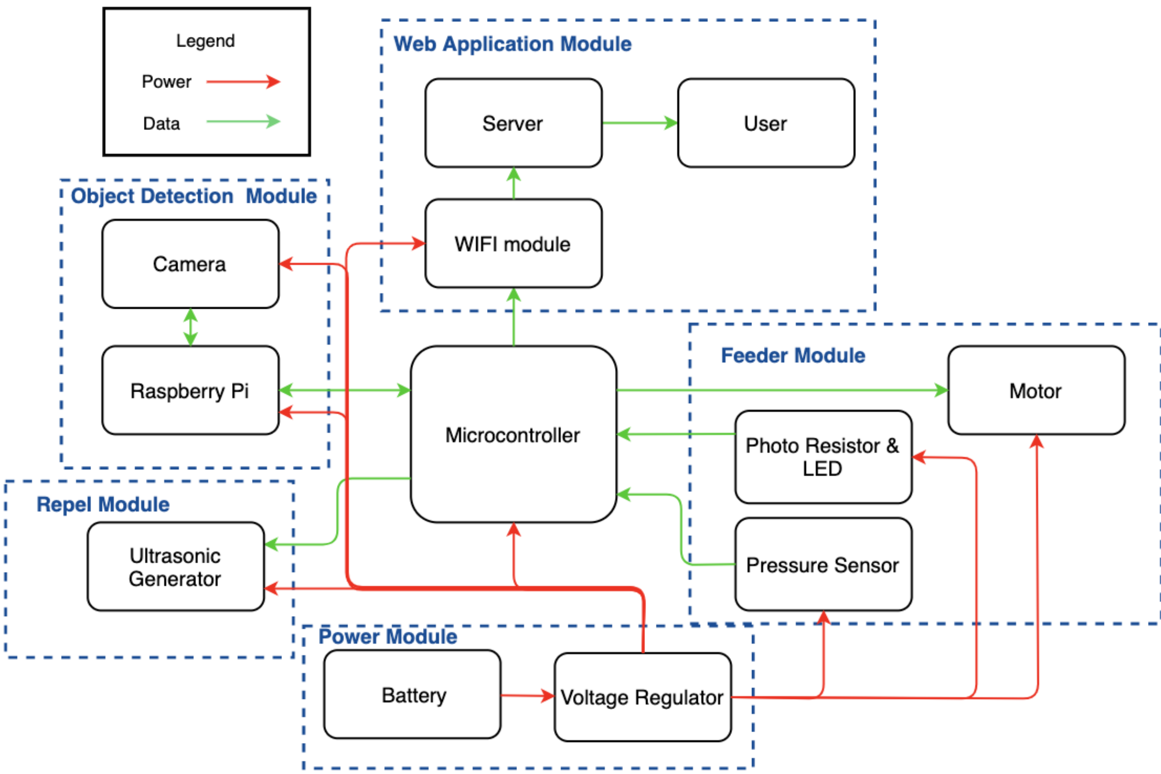


Figure 1: Bird Feeder Block Diagram

2.2 Circuit

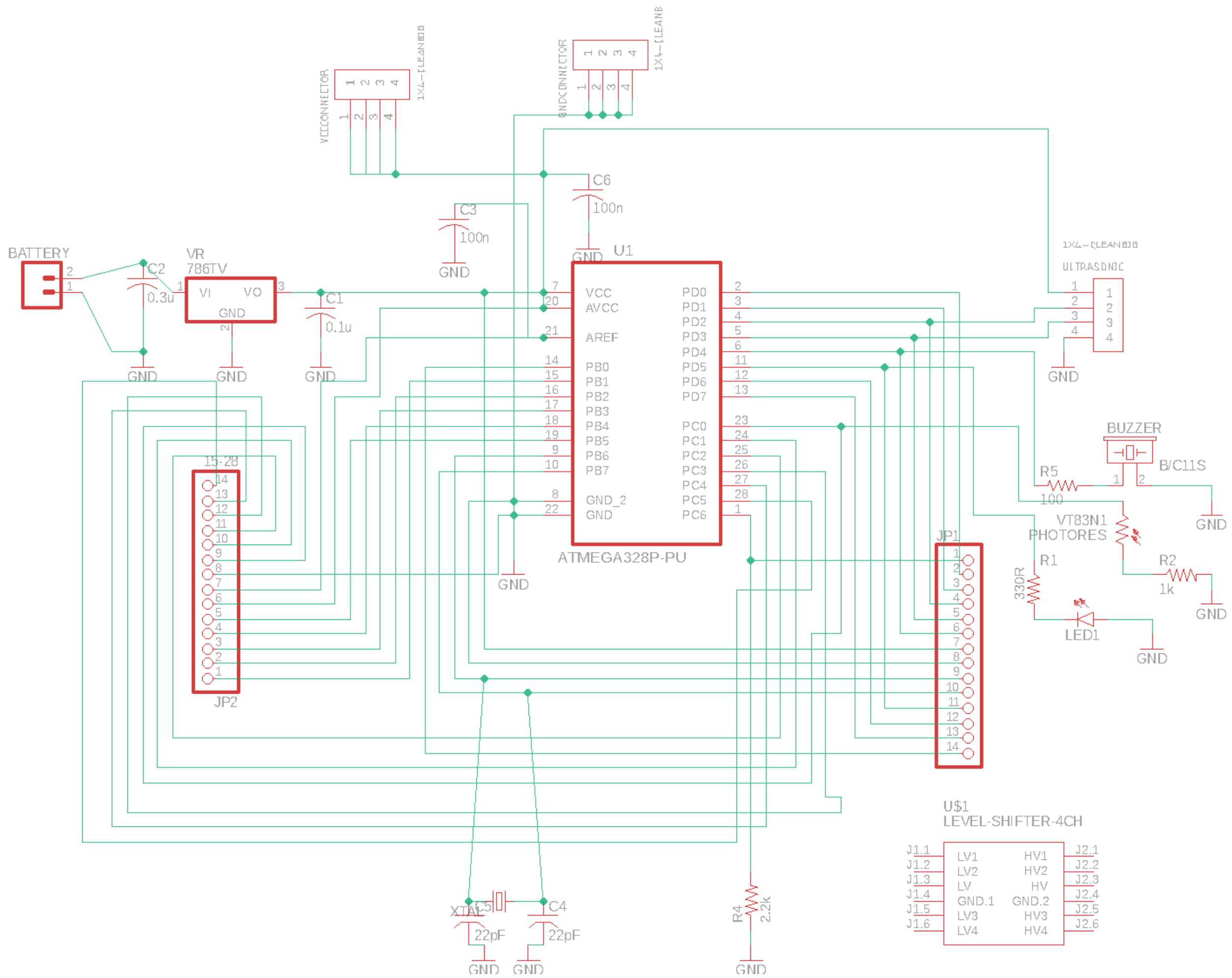


Figure 2: Circuit Schematic of Bird Feeder

2.3 Flowchart

As shown in the Figure 3, there are two threads running at the same time, food detection and object detection. On the first thread, the ultrasonic sensor runs all the time to detect if there is an object in front of the feeder. If the ultrasonic sensor detect an object, the object detection algorithm on Raspberry Pi will start. The algorithm only runs once for each object to reduce the power consumption, which also means the bird feeder loads once bird food for each bird. After processing the image, the model will identify the object. A squirrel will lead the repel system to start. If the object is a bird, the picture will be uploaded to the image host and store the image's URL to the database by Raspberry Pi. In the mean time, microcontorller will control the motor to load food for the bird.

On the other thread, LED lights up for every one hour. The photoresistor checks if there is enough food in the container. If the bird food in the feeder is below the photoresistor, the food shortage signal will send to the web application.

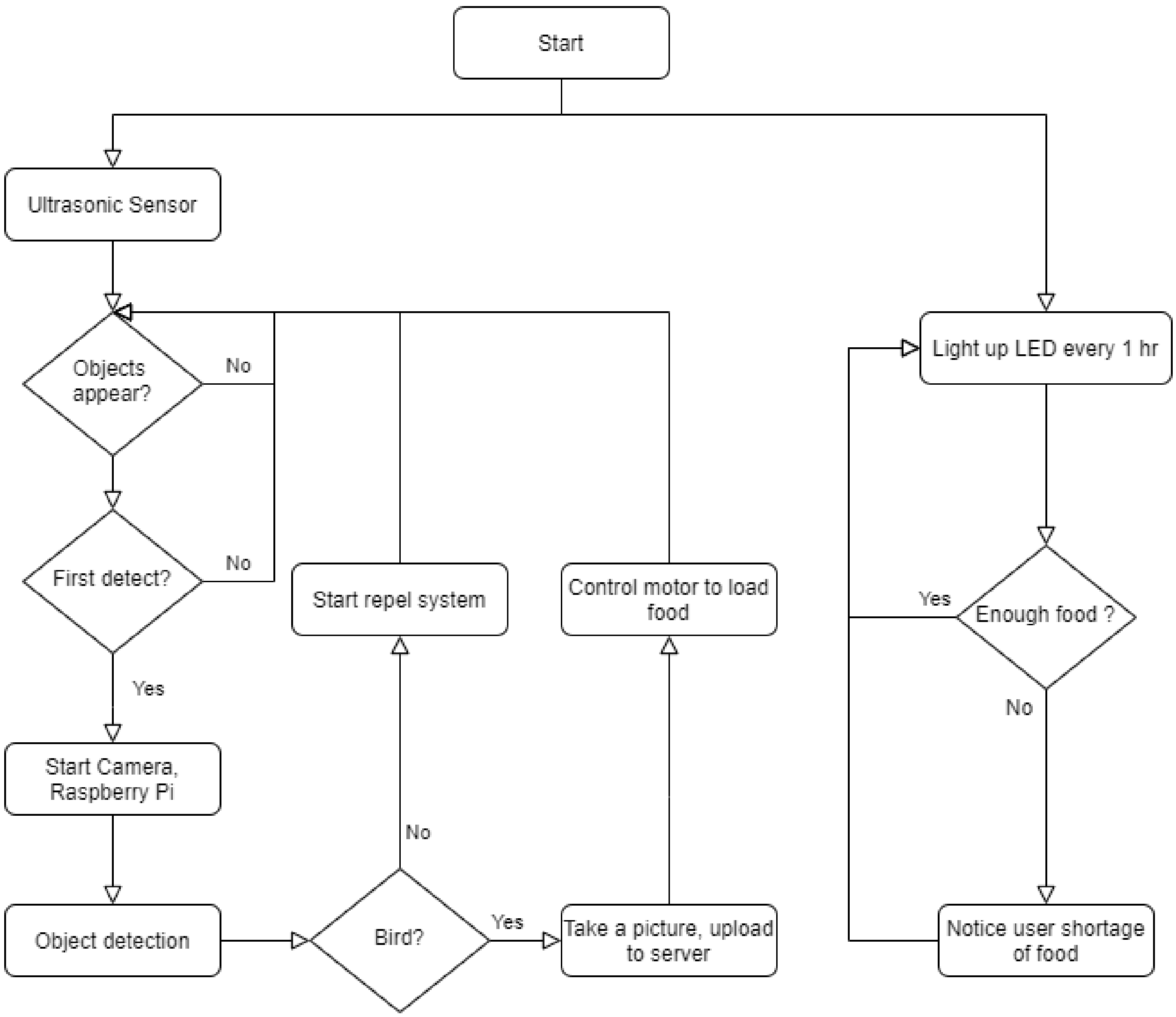


Figure 3: Bird Feeder Flowchart

2.4 Physical Design

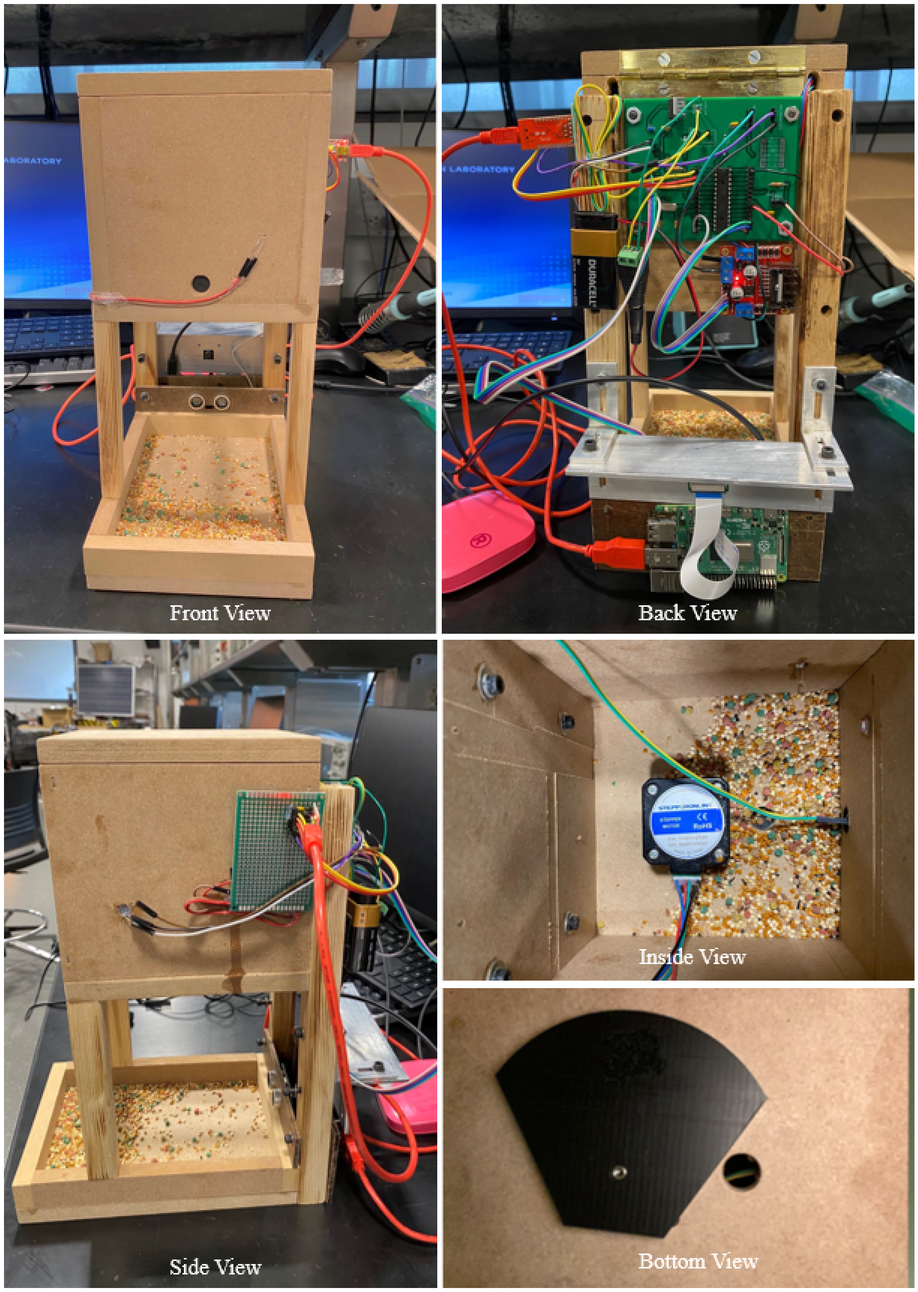


Figure 4: Physical Design of Bird Feeder

Figure 4 shows the thorough view of our real product. The device is around 10 inches tall and 4 inches wide. The upper box is 4 inches depth and bottom platform is 6 inches depth. The box above is the place to store bird food, and the lid could be opened for food refilling. The ultrasonic sensor and Pi camera are placed in the back facing forward in order to provide enough distance to sense and take pictures when animals come. The metal that holds the

Pi camera could be slide manually up and down, back and forth to adjust the position of the camera. PCB broad, motor driver, Raspberry Pi and batteries are mounted in the back. The stepper motor is placed inside the feeder box and fixed with the black plate underneath. The LED and photoresistor system is implemented at a corner inside the box because it would be easier for photoresistor to sense the light from LED. All components are connected together with jumping wires.

2.5 Control Module

The control module contains the communication between Raspberry Pi and ATMega328, the microcontroller. Based on the flowchart shown in figure 3, the microcontroller will send a signal indicating the existence of the object to Raspberry Pi. After the Raspberry Pi has done image processing, it will send the result to the microcontroller. The microcontroller receives the signal and decides to either load food for the birds or repel the squirrels. Meanwhile, the LED will be lighted up every one hour, which is controlled by the microcontroller. When the food level is less than 10%, the microcontroller will send the signal to Raspberry Pi based on the change of current going through the photoresistor, and the Raspberry Pi will upload the food shortage notification to the website via the server. In our project, serial communication is used for its stability and speed.

2.6 Object Detection Module

The object detection module contains three components, Raspberry Pi 4 with 2GB RAM works as the single board computer to run object detection model, a Pi camera and an ultrasonic sensor. The Raspberry Pi has WiFi module build in, so it also work as the connection between the bird feeder and web application module.

As Raspberry Pi is a computer, the power consumption is huge. To reduce the power consumption, we don't want Raspberry Pi to run all the time and perform object detection on live streaming. To achieve this goal, we implement object detection module with an ultrasonic sensor, whose power consumption is relatively low. The ultrasonic sensor will run all the time to detect if there is any object in front of the bird feeder or not. With the output of ultrasonic sensor, the control unit will conduct a signal to start the camera and to run object detection on Raspberry Pi.

The object detection model is trained by TensorFlow 2 with pre-trained model, SSD mobilenet V2, from TensorFlow model zoo [4]. We used 400 images of birds and squirrels as the train/test data, and set batch size to 4. The whole process takes around 5 hours with GTX 1050Ti and converges at 12,000 steps (see figure 5). To run better on Raspberry Pi, We convert the TensorFlow model to TFLite model with TFLite converter [5]. Compare with TensorFlow model, TFLite model is much faster, less size and less computationally expensive. It is more suitable for small device like Raspberry Pi. To further reduce the latency time, quantization is introduced during the conversion. We converted a model with dynamic range quantization, but full integer quantization conversion caused some unknown error, which might relate to support issue of TensorFlow 2. Theoretically, full integer quantization reduces the most latency time. Fortunately, We found a trained model which uses TensorFlow 1 and successfully quantizes by full integer.

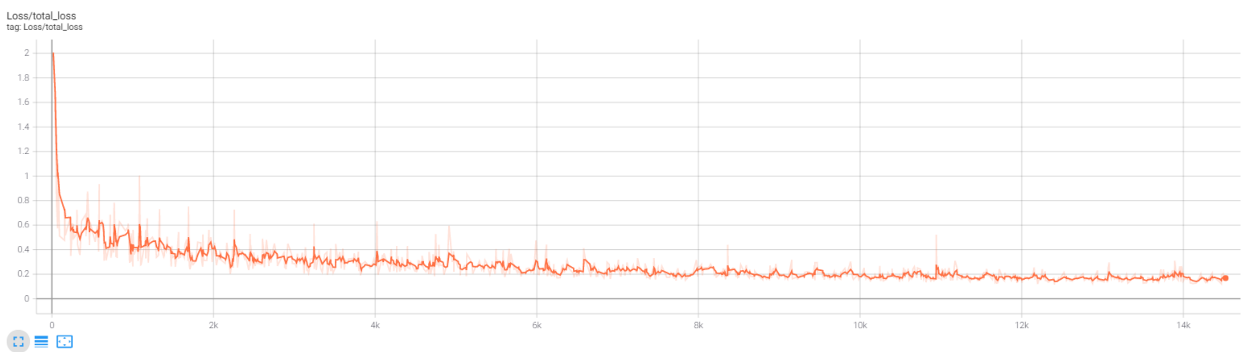


Figure 5: Diagram of Total Loss

Below algorithm 1 is the script running all the time on the Raspberry Pi to communicate with control unit and call different functions to perform object detection and insertion to the database.

Algorithm 1 *pi_script*()

```
ser = serial connection with Atmega328p
camera = connect to Pi camera
while True do
  try do
    wait for Atmega328p
    if ser.in_waiting> 0 then
      if food shortage notification then
          SSH to cPanel to insert notification
      else
          pic = camera.capture()
          species = object_detection(pic)
          if species == bird then
              url = imgur.upload(pic)
              SSH to cPanel to insert url
          end if
          write species to Atmega328p
      end if
    end if
  except
    pass
end while
```

2.7 Repel Module

The purpose of the repel module is to chase away squirrels that come to the bird feeder. According to the research, squirrels are able to hear sounds up to 49 kHz whereas birds and humans are only sensitive to sounds with frequency around 1 kHz to 4 kHz [2]. Since ultrasound works as alarm call of danger among squirrels, playing ultrasound should be efficient for repelling squirrels and this alarm will not disturb birds or humans. We used a passive electromagnetic buzzer of which the frequency of the sound it generates could be controlled between 50 Hz to 14 kHz. The buzzer is connected directly with microcontroller which provide audio drive signal to the buzzer in order to make it sound. When a squirrel is detected, signal will be send to Atmega328 and it will control the buzzer to output ultrasound of the frequency we set.

2.8 Feeder Module

The feeder module aims to load food for the bird detected and to detect the food shortage in the same time as mentioned in the flowchart of the previous section. To fulfill the function of loading the bird food, we design a mechanical system that could release a small amount of food each time. First, we designed to drill a hole at the bottom of the feeder box and to mount a rotatable plate with a same size hole below. To drive the plate, we used the L298N motor driver and a stepper motor since the angle of its rotation could be controlled precisely. Thus, when the microcontroller send signals to make the driver rotate, the motor will turn clockwise to let two drilled holes match with each other and rotate back to the original position counterclockwise. To control the amount of food loaded each time, we could set the wait time for the motor before rotate back to give more time for releasing food. In real testing, we found the torque for rotating the entire plate is too much for the stepper motor due to friction between surfaces and not enough power supply, so we decided to cut the plate into a fan-shape piece just to cover the hole at the bottom of the box. This way, the covering piece still rotate to cover and uncover the hole of the box but with smaller weight and friction.

To detect the food shortage inside the box, we originally designed to use a pressure sensor at the bottom to sense the weight of the food. However, the hole and motor the at bottom of the box for food releasing made the set up of pressure sensor much harder. So, we designed a LED & photoresistor system that could be implemented on the inside wall. The photoresistor will be connect to Atmega328 all the time and send real time detection result to the microcontroller. The inside of the box is in dark all time. The LED is set to be light up, if the food level is higher than the position of photoresistor, it will not able to sense the light from LED. Otherwise, if the photoresistor could sense the light from LED, it means the food is below the level of photoresistor. Then, a signal should be send to web module to notify the user. We set up the LED to be light up every one hour, so the system would automatically check if the food needs to be refilled.

2.9 Web Application Module

Web application module works as the interface of the smart squirrel proof bird feeder. The website is "birdfeeder.web.illinois.edu". We use cPanel as the host to the website and mySQL on the cPanel as database. cPanel is capable to run python application, so we choose to use django as our web framework and build the website with python and HTML. Since cPanel with Illinois account doesn't support remote database, to access the database we have to use SSH access feature of cPanel. The SSH insertion calls a shell script which contains SQL command stored in cPanel. This makes store pictures in database impossible and also storing images directly in database has several disadvantages. Thus, we use Imgur API to load image to Imgur which is a image host, and store the url to the images in the database.

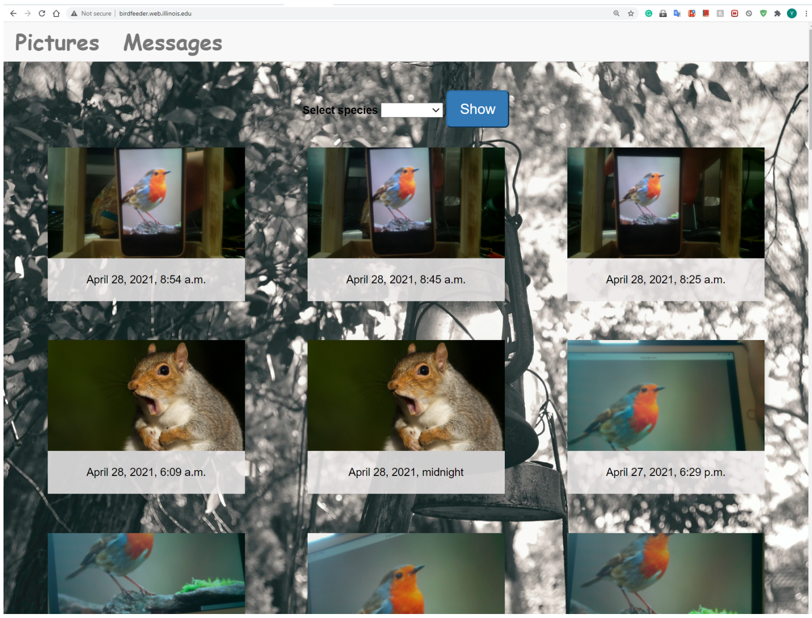


Figure 6: Screenshot of Picture Page

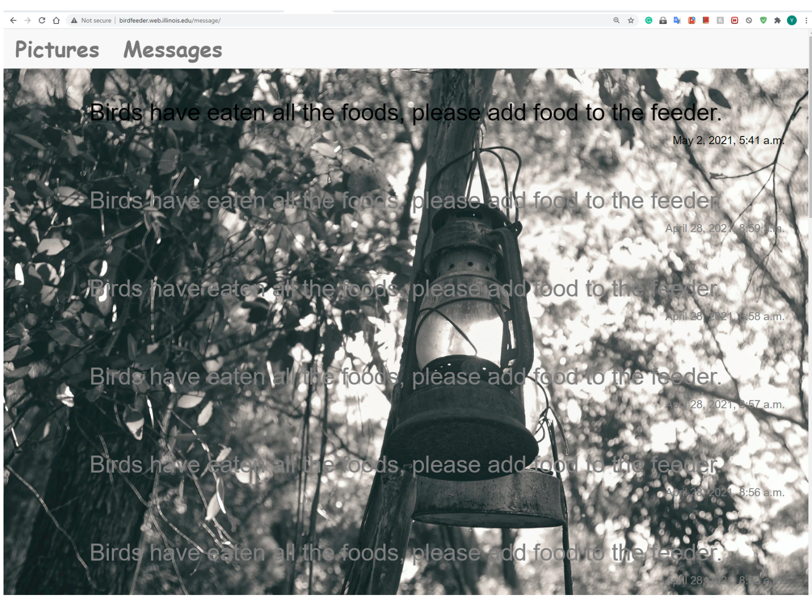


Figure 7: Screenshot of Message Page

The website has two pages. One is picture page and the other one is message page. Figure 6 is the picture page that shows all the birds' images with the time. Currently the pictures of squirrels would not insert to the database in our design, but it can be easily changed. Therefore, the website also has a filter feature to show only images of birds or squirrels. Figure 7 is the message page which shows the food shortage notification. The font color of the notification would change from black to grey after user views it. Whenever there is a black notification, user should refill bird food to the bird feeder.

2.10 Power Module

The power module will have two separate power supplies, one for the microcontroller, the other for Raspberry Pi. The alkaline 9v battery will be used to power the microcontroller chip and hardware components (ultrasonic sensor, buzzer, LED, and motor driver). The voltage needed in the circuit is 5v, therefore, a voltage regulator is used in the circuit. To power the Raspberry Pi, a large power bank is used since it can provide a long time running the program and is convenient to recharge.

3 Design Verification

3.1 Control Module

We have created three tests on serial communication between the microcontroller and the Raspberry Pi to ensure the overall functionality of our project. The module can pass all the tests and complete the goals of the project.

1. The LED will light up every 1 hour. When the photoresistor detects the light, the Raspberry Pi can receive the signal from the microcontroller and upload the food shortage notification to the website.
2. The Raspberry Pi is able to receive the signal from the microcontroller when the ultrasonic sensor detects an object.
3. The microcontroller can receive the result of image processing from Raspberry Pi immediately.

3.2 Object Detection Module

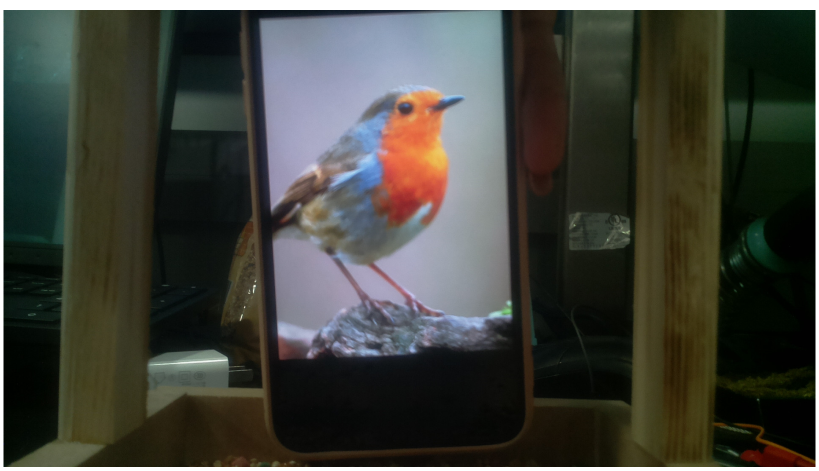
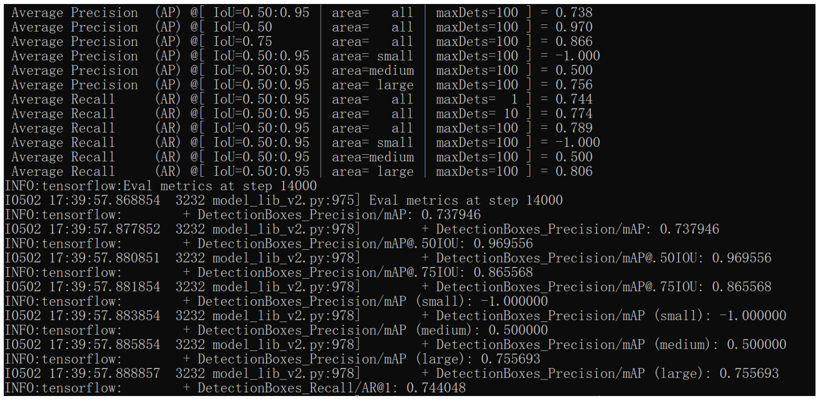


Figure 8: Image Captured by the Pi Camera

We used a picture of bird showing on the iPhone to test the system. As shown in figure 8, the Pi camera is able to capture the whole image and also include extra area besides the iPhone. This satisfied our requirement for the area that Pi camera need to capture. Furthermore, the position of Pi camera is not fixed. The machine shop made a steel frame that can move camera in all directions.



```
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.738
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.970
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.866
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.500
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.756
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.744
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.774
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.789
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.500
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.806
INFO:tensorflow:Eval metrics at step 14000
I0502 17:39:57.868854  3232 model_lib_v2.py:975] Eval metrics at step 14000
INFO:tensorflow:        + DetectionBoxes_Precision/mAP: 0.737946
I0502 17:39:57.877852  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP: 0.737946
INFO:tensorflow:        + DetectionBoxes_Precision/mAP@.50IOU: 0.969556
I0502 17:39:57.880851  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP@.50IOU: 0.969556
INFO:tensorflow:        + DetectionBoxes_Precision/mAP@.75IOU: 0.865568
I0502 17:39:57.881854  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP@.75IOU: 0.865568
INFO:tensorflow:        + DetectionBoxes_Precision/mAP (small): -1.000000
I0502 17:39:57.883854  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP (small): -1.000000
INFO:tensorflow:        + DetectionBoxes_Precision/mAP (medium): 0.500000
I0502 17:39:57.885854  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP (medium): 0.500000
INFO:tensorflow:        + DetectionBoxes_Precision/mAP (large): 0.755693
I0502 17:39:57.888857  3232 model_lib_v2.py:978]        + DetectionBoxes_Precision/mAP (large): 0.755693
INFO:tensorflow:        + DetectionBoxes_Recall/AR@1: 0.744048
```

Figure 9: COCO Evaluation Metrics

During the training process of the object detection model, the COCO evaluation can be set to true to get all COCO evaluation metrics. Figure 9 is the screenshot of the COCO evaluation Metrics to our own model. As localization is useless for the bird feeder, we can use 0.5 IOU as the threshold for calculating the mean average precision. The mAP of the model is 0.969 which is higher than the requirement 0.9.

Table 1: TFLite model comparison

Model	**Quantization**	**Time to process one image (s)**	**Accuracy**
Bird_Squirrel	None	1.9	0.8958
Bird_Squirrel	Dynamic Range Quantization	1.4	0.9
Bird_Squirrel_Raccoon	Full Integer Quantization	0.26	0.167

The most important requirement of the object detection module is real time. We want the object detection happens in $500ms$, which is satisfied by the model, Bird_Squirrel_Raccoon. It only takes 0.26 second to output a label of one image (see table 1). We also used 240 images to test the accuracy of all three models. Bird_Squirrel_Raccoon has the highest accuracy 0.9167.

3.3 Repel Module

To test the output frequency from the passive electromagnetic buzzer, we connected the buzzer directly with an Arduino on the breadboard. Since we do not have microphone and spectrum analyzer to measure the sound frequency, we decided to connect the buzzer with oscilloscope to measure out the alternating current frequency in the buzzer. First, we set up the output frequency to 20 kHz and display AC signal on the oscilloscope, the peak to peak frequency for the output is around 18 kHZ which is a little below 20 kHz. Then, we set up the frequency to 45 kHz, as shown in Figure 10, the peak to peak AC frequency is about 40 kHZ which theoretically should be able to repel squirrels. However, due to limited time and space, we are not able to test of the feedback and reaction of this ultrasound from real squirrels.

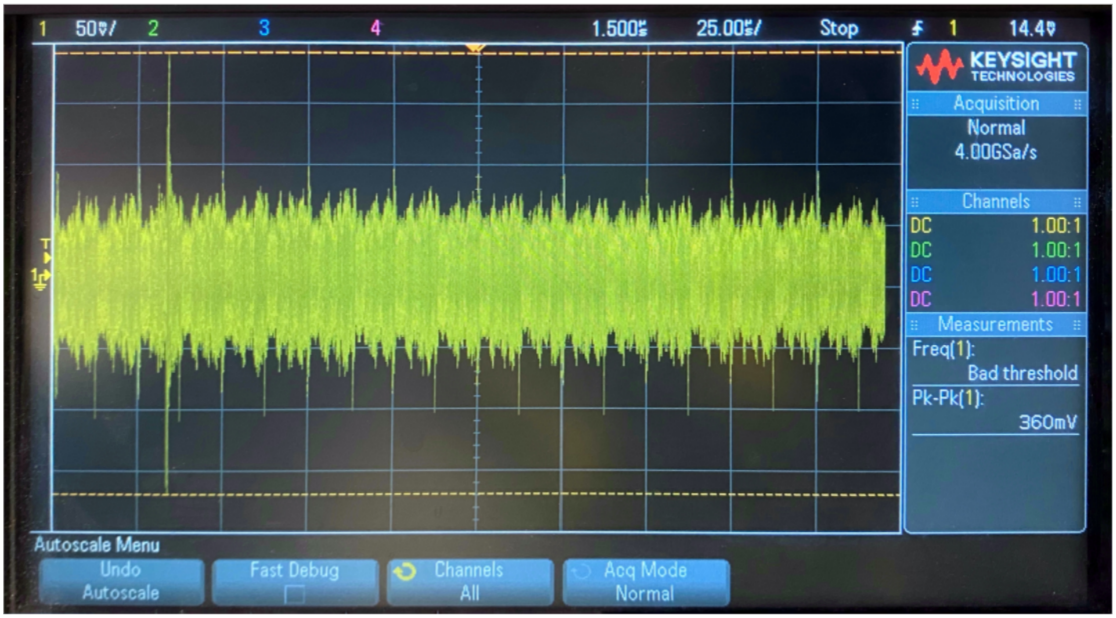


Figure 10: Oscilloscope view of AC frequency for buzzer

3.4 Feeder Module

To test the food dispensing system, we firstly tested the stepper motor with motor driver and Arduino to make sure the motors are able to rotate with the angle and speed as desired. After machine shop helped us installed the motor in the feeder box and fixed it with the rotating plate, we created another test to check if the motor could drive the plate to cover and uncover the hole properly. After this step, we added bird food inside the box and eventually put it into a thorough test. We found that the amount of the food loaded is appropriate during each round when the plate got moved away and return to its position immediately. 15 trials were made and we record each outcome and calculated that the average weight of the loaded food is around 10 grams which is reasonable amount of food for each bird.

For the LED and photoresistor food shortage detection, we first set LED to light up every minute and microcontroller to read analog value from photoresistor. The resistance of the photoresistor will decrease when there is more light. So when reading voltage value from the resistor, we could determine the brightness inside the box. After testing, we have set in the microcontroller that if the value is larger than 4, it will send signal to Raspberry Pi which will upload notification on the website.

3.5 Web Application Module

We created three unit tests for the web application module to test each feature. The system can pass all the tests.

1. Run Imgur API to upload a image and get the URL of the image. The URL links to the image.
2. Insert image to database. We can see the image on the picture page of the website.
3. Insert notification to database. We can see the notification message on the message page of the website.

3.6 Power Module

The operating voltage and current of the main components is shown in table 2. The 9v alkaline battery our project uses has around 500mAh capacity, which meaning it can deliver a current of 500mA for one hour. After rough calculation, the feeder machine will work around 30 hours before it runs out of power when detecting 10 birds per hour. For Raspberry Pi, since it is powered by a power bank, a larger power bank of the customer's choice will provide a longer time. However, our project does not need to work 24 hours per day. Thus, adding extra features, such as a switch or solar panel, can save more power.

Table 2: Operating voltage and current of components

Component	**Operating Voltage**	**Operating Current**
Ultrasonic Sensor	DC 5V	15 mA
LED	DC 5V	15 mA
Motor	DC 5V	~700 mA
Buzzer	DC 5V	~10 mA
Raspberry Pi	5V	500mA~1.2A
Total		1235A~1935A

4 Cost

4.1 Labor

The total estimated time for the project is approximately 360 man hours. Also, assuming the development salary is around \$40/*hour*. The total labor cost for the project will be:

Table 3: Labor Cost

Name	Hourly Rate	Total Hours	Total
Linfei Jing	\$40	120	\$12,000
Christine Li	\$40	120	\$12,000
Yitian Xue	\$40	120	\$12,000
Total			\$36,000

4.2 Components

Table 4: Part Cost

Part	Quantity	Cost
ARDUINO UNO R3 [A000066]	1	\$23
Buzzer [W-08A]	1	\$2.32
ULTRASONIC SENSOR [3942]	1	\$3.95
STANDARD MOTOR 9100 RPM	1	\$1.95
PHOTO RESISTOR GM5539	1	\$0.95
LIGHT EMITTING DIODE [5mm]	1	\$0.5
Resistors [A-0003-C07]	10	\$5.95
RASPBERRY PI 4 MODEL B 4GB SDRAM	1	\$55
Total		\$93.32

4.3 Total

Sum of the cost for this project is around **\$36,093.32**.

5 Conclusion

5.1 Accomplishments

In the end, the squirrel proof bird feeder we built satisfied all the high level requirements. The bird feeder is exactly what we imagined and designed at the beginning of the semester. The object detection module can distinguish squirrels and birds in real time, and the feeder module is able to execute corresponding actions, loading bird food or repelling squirrels. The web application module has all the functions for users to view images of birds and get food shortage notification. Overall, We consider our project to be successful.

5.2 Uncertainties

The material of the bird feeder we are currently using is not water-proof. The material may deform when it gets wet. Also, our PCB board, battery, and Raspberry Pi are all exposed to air without covering. We may need to add protection to cover the entire machine from humidity and other animals or change the material to other waterproofed ones. Also, since we are using alkaline 9v battery as power source, the voltage output from the battery may decay as it runs out. Lower voltage may not be able to run the process properly.

5.3 Ethics and Safety

There are several concerns regarding the safety and ethics in the project which need further consideration. For the safety concerns, improper ways of using and storing batteries can cause fire or explosion hazard. The damage may can occur when the temperatures are too high (e.g., above 130°F) or too low (below freezing 32°F [6]).

The bird feeder, as an outdoor electrical device, could be damaged by the moisture in extreme weather which can lead to short-circuits. We are responsible for notifying the users of the potential consequences and send out the instruction and correct implementation for this product, in compliance with IEEE Code of Ethics [7].

For ethical issues, we will focus on implementing the security for the web application module to ensure the privacy of users. We will keep the data confidential and reliable by avoiding illegal transmission of the data and clarifying the retention and disposal periods for the information [8].

5.4 Future work

To further improve our project, we will perform different tests for it. We will test the bird feeder with real squirrels to verify if the repel module work efficiently. The feeder machine will also be tested under different weather conditions to ensure its endurance.

To reduce the complexity for users, we want to combine the power supply for PCB and Raspberry Pi. To reduce the overall power consumption, we can add solar panels to efficiently take the advantage of the sunlight. Adding extra features such as an on-off switch, which lets users to decide when to turn the feeder on and off, can also save power consumption.

References

[1] N. P. M. Association, "Squirrels 101," Feb 2021, accessed February 2020. [Online]. Available: https://www.pestworld.org/news-hub/pest-articles/squirrels-101/

[2] J. L. H. HE and H. RS, "Audiogram of the fox squirrel (sciurus niger)," *J Comp Psychol*, Mar 1997, accessed February 2020.

[3] D. R. Wilson and J. F. Hare, "Ground squirrel uses ultrasonic alarms," *Nature*, Jul 2004, accessed February 2020.

[4] L. Vladimirov, "Tensorflow 2 object detection api tutorial," 2020. [Online]. Available: https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/install.html

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[6] Osha.gov, "Preventing fire and/or explosion injury from small and wearable lithium battery powered devices," Jun 2019, accessed February 2020. [Online]. Available: https://www.osha.gov/dts/shib/shib011819.html

[7] ieee.org, "Ieee code of ethics," 2016, accessed February 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[8] acm.org, "Acm code of ethics and professional conduct," 2018, accessed February 2020. [Online]. Available: https://www.acm.org/code-of-ethics

Appendix A Requirement and Verification Table

Table 5: R&V Table for Repel Module

Requirement	Verification	Result
1. The ultrasound generator should generate ultrasound via a microcontroller at a frequency level of $40kHz$ $\pm 2kHz$ and a sound-pressure level of $100dB$ $\pm 5dB$ to repel squirrels.	1. Connect the output signal from the microcontroller to the ultrasonic annunciator to an oscilloscope. The signal should oscillate at a frequency of $40kHz$ $\pm 500Hz$.	Satisfied
2. The ultrasonic generator can generate ultrasound with a frequency range of $20kHz$ - $50kHz$.	2. To test if the sound frequency of the ultrasonic annunciator has reached the desired range, we need utilize the spectrum analyzer to measure its frequency range.	Satisfied under lab setting: AC frequency satisfied

Table 6: R&V Table for Web Application Module

Requirement	Verification	Result
1. Raspberry Pi can insert pictures and notification to the database.	1. Apply insertion by ssh from raspberry pi and check if the data is in the database.	Satisfied
2. User should be able to view the history of bird feeder.	2. All the images should show up when click the image tab and all the notifications should show up when click the message tab.	Satisfied
3. The module can upload pictures to image host and get the url of pictures.	3. Run the insertion command and see the picture on the website.	Satisfied
4. User can select to only show images of squirrel or bird.	4. Use the filter on the web app to get images of only one species.	Satisfied

Table 7: R&V Table for Object Detection Module

Requirement	**Verification**	**Result**
1. The camera placed within 15cm to the objects should output image or video that covers the full front side of the feeder (approximately $250cm^2$).	1. Take pictures by the camera with a $16cm$ by $16cm$ referenced object placed $10 - 15cm$ away from the camera, ensure the camera can capture the whole object.	Satisfied
2. The object detection should happen in real time, at least in $500ms$.	2&3a. Test the algorithm by importing bunch of test images, time average processing time and compute the mAP. The average testing time should less than $500ms$ and the mAP should higher than 90%.	Satisfied
3. The object detection model have a mAP (mean average precision) higher than 90%.	2&3b. Put the same test images in front of the camera. The signal for indicating whether this is a squirrel or bird should appear without noticeable delay and the mAP should be higher than 90%.	Satisfied
4. Raspberry pi must be able to communicate with microcontroller via serial communication.	4. Input signals from Raspberry pi and microcontroller, the output should appear immediately and correctly.	Satisfied
5. Raspberry pi must be able to connect WiFi and upload pictures to the web application.	5. Place a picture of the bird, a picture of that picture should appear on the web application with the date and time information.	Satisfied

Table 8: R&V Table for Control Module

Requirement	**Verification**	**Result**
1. The microcontroller must be able to communicate with Raspberry Pi.	1. Input signals from Raspberry pi and microcontroller, the output should appear immediately and correctly.	Satisfied
2. The microcontroller should send out high frequency signal ($40kHz$ $\pm5\%$) to ultrasonic annunciator.	2.Connect the output signal from the microcontroller to the ultrasonic annunciator to an oscilloscope. The signal should oscillate at a frequency of $40kHz$ $\pm5\%$.	Satisfied
3. The microcontroller should control the motor to run for $1s$ to load $10g$ $\pm5\%$ bird food.	3. Input a signal for the microcontroller to load bird food, the loaded bird food should weight $10g$ $\pm5\%$.	Satisfied

Table 9: R&V Table for Feeder Module

Requirement	**Verification**	**Result**
1. The feeder will load food after receiving the signal transmitted back from the image processing result.	1. Implement the identify-feed process with bird picture/model to test the accuracy.	Satisfied
2. The feeder will load approximately 10g bird food each time.	2. Measure the amount of food that the feeder will load each time. Make sure the error is less than 10%.	Satisfied
3. The user will get notification if there is a shortage (less than 10%) in the bird food.	3. Test with different food amount for the accuracy of food shortage detection with photoresistor and LED. If the bird food is below the 10% mark, it should trigger the food-shortage signal.	Satisfied

Table 10: R&V Table for Power Module

Requirement	**Verification**	**Result**
1. The batteries must be able to store enough charge to provide at leaset 50mA at 9V for 1 hour.	1A. Attach a 5 Ω resistor as a load. Measure the current through resistor with an oscilloscope. Discharge the battery for one hour.	Satisfied
2. The voltage regulator must provide 5V±5% at 50mA.	2.Attach 5Ω resistor as a load. Supply the regulator with 9V DC. Measure and ensure the output voltage remains between 4.75V and 5.25V using the oscilloscope.	Satisfied