Justin Morris
Brendon Chen
Minh Tran Quoc

Pet++

Remote Automated Pet Monitoring and Training

Final Report

Abstract

Professional pet trainers teach owners that consistency is key to successfully training their pets. However, most pet owners need to leave for work for most of the day leaving their in-training pets to act on their impulses and often degrading their training. There exist products on the market that allow owners to monitor their pets remotely to check on them, but no product has the ability to automatically react to pets and reinforce training. Pet++ is the first automatic pet monitoring and training framework. Pet++ utilizes several sensors, like cameras, to monitor the pet's activities. The central system then processes the sensor information and triggers other devices, like speakers, to react to the pet to reinforce training. We tested Pet++ in a typical scenario where an owner wants to keep their pet off of a couch. The system was able to successfully get the pet off the couch 90% of the time with automated commands.

Introduction

Motivation

Most people that have raised pets know that the key to successfully training them is consistency. That is, using the same clear voice commands every time your pet does the action associated with the command. However, most pet owners do not have the luxury of being with their pets 24/7. Therefore, there are times when the pet is on their own, acting on their impulses, often slowing down the training process. For instance, say you are training your dog to stay off the couch. When you are there with them, you can easily tell them “off” when they go on the couch. You can also easily reward them for not going on the couch, reinforcing the training. However, when you go off to work, your pet is on their own and they might find the couch comfier than their bed. Therefore, there is no one there to continue the consistent training and the pet gets mixed signals about the couch. This significantly slows down or outright prevents the pet from learning that they are not allowed on the couch. We are proposing a solution to this problem, remote automated pet training and monitoring.

Related Work

Existing solutions to monitoring pets remotely simply offer a way to check in on your pets [1]. Some of these products offer additional features like dispensing a treat, however, none of them can automatically react to your pet and assist with training or even log activity for the user to review. Furthermore, these products are often closed source, and the user is unable to add any additional features. There are also a plethora of patents regarding monitoring pets and devices to assist with training [2]–[4]. However, from our research there does not seem to be any product on the market that can automatically monitor and assist with training your pet. The exception being glorified torture devices such as shock colars [5]. In contrast, our product provides a remote monitoring system where users can add any type of sensor they desire to monitor their pets. In addition, they can even add network devices that can react to their pets such as a speaker to play commands. The system then monitors the deployed sensors and triggers automations to assist in training the pets autonomously. Lasty, the system has a web UI where users can view logs of activity, live sensor readings, as well as interact with their pets live if they have a speaker setup.

Studies have shown that the key to successful habit formation is repetition and consistency [6]. This sentiment spans all kinds of applications including training dogs as most professional dog trainers share the idea that consistency is key to successful training [7]. Therefore, pets owners around the world often run into an issue in that they are not home 24/7 to monitor their pets and they often perform undesirable behaviors during this time. Our system attempts to combat this with the best practice of providing consistent training by monitoring the pet with a system of sensors and triggering automations created by the user to continue training even when the user is not home.

Project Overview

Our remote automated pet training and monitoring is a system of sensors connected to a central computer that will both monitor your pet as well as automatically reinforce their training. Our system accomplishes these two tasks in the following ways: 1) The sensors collect information about the environment and how your pet is interacting with it and sends that information to the central computer. 2) The central computer then processes the sensor information and performs automated training if the

sensors detect your pet's behavior needs to be corrected. With our remote automated pet training framework, the user has the ability to add any sensors they want to the system to help them automate training their pets. They also have the ability to determine when and how the system should react to their pet by setting up automations based on the sensor information.
Going back to the couch example, a user could add a camera sensor to the system and link the camera feed to the framework. They would then be able to indicate where the couch is in the camera feed. Then, if they also add a speaker to the system they would be able to set up an automation to be consistent with the "off" command they are trying to teach their dog. For instance, when the dog appears on the couch, the speaker can play a pre-recorded message of the user saying the "off" command. The dog would then hop off the couch like when the user is normally home! This helps train pets by giving owners the ability to stay consistent with commands they are trying to teach their pets when they are not even home. We plan on implementing support for a variety of sensors. We also plan to test the ability of our system to be easily user programmable by creating and testing multiple use cases.

Justin Morris
Brendon Chen
Minh Tran Quoc

Technical Material

Overall Design

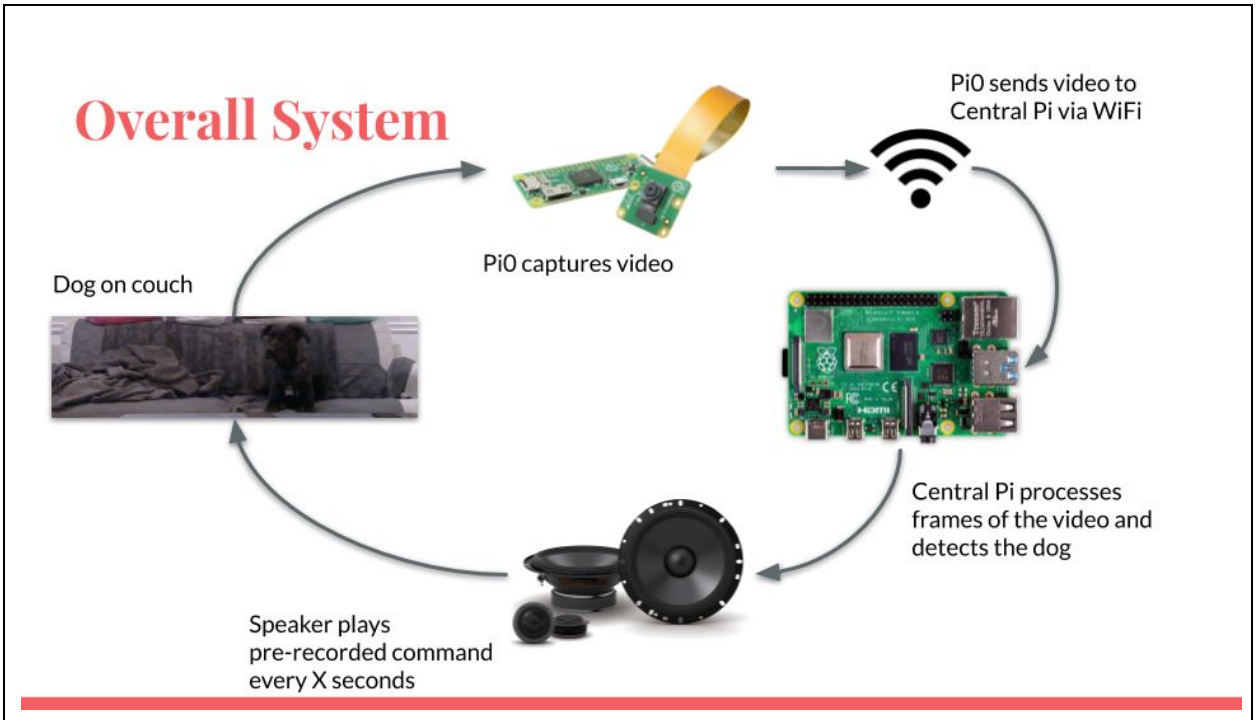

Figure 1. Overview of how the automated pet monitoring and training system communicates data.

Figure 1 shows an overview of the automated pet monitoring system. The passive sensors, such as the camera, communicate their sensor information to the central system, in this case a Raspberry Pi. The Raspberry Pi then processes that data and if any of the automations that were set up by the user are triggered, the Raspberry Pi would then send the programmed commands to the appropriate active sensors, such as the speakers. For the active and passive sensors and communication interfaces, we used existing libraries such as pycamera, pyaudio, and sockets. The bulk of our contribution will be the processing of the sensor information on the Raspberry Pi as well as the user interface.

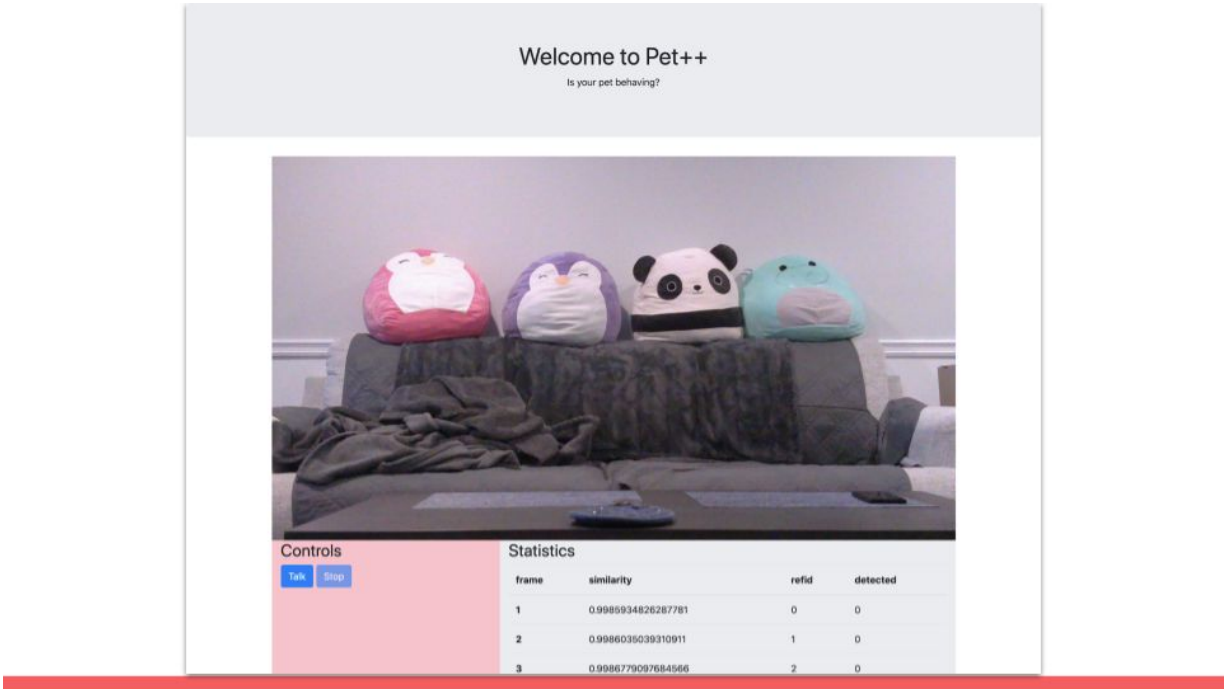

Figure 2. The Web UI.

As pictured in *Figure 2*, the current web UI takes on a simple form, which provides three main features to the user: a live video feed of an area of interest that the system monitors, statistics pertaining to each frame capture and analyzed by the system, and controls for the user to take over in order to initiate direct voice communication with the pet. The first two features, live video feed and statistics feed, work as intended. The third feature is not fully connected. To be specific, an end-to-end Python-based solution was implemented and demonstrated to work. Using that system, users are able to initiate real-time voice communication through the system. Unfortunately, integrating that system with the web UI proved non-trivial.

Justin Morris
Brendon Chen
Minh Tran Quoc

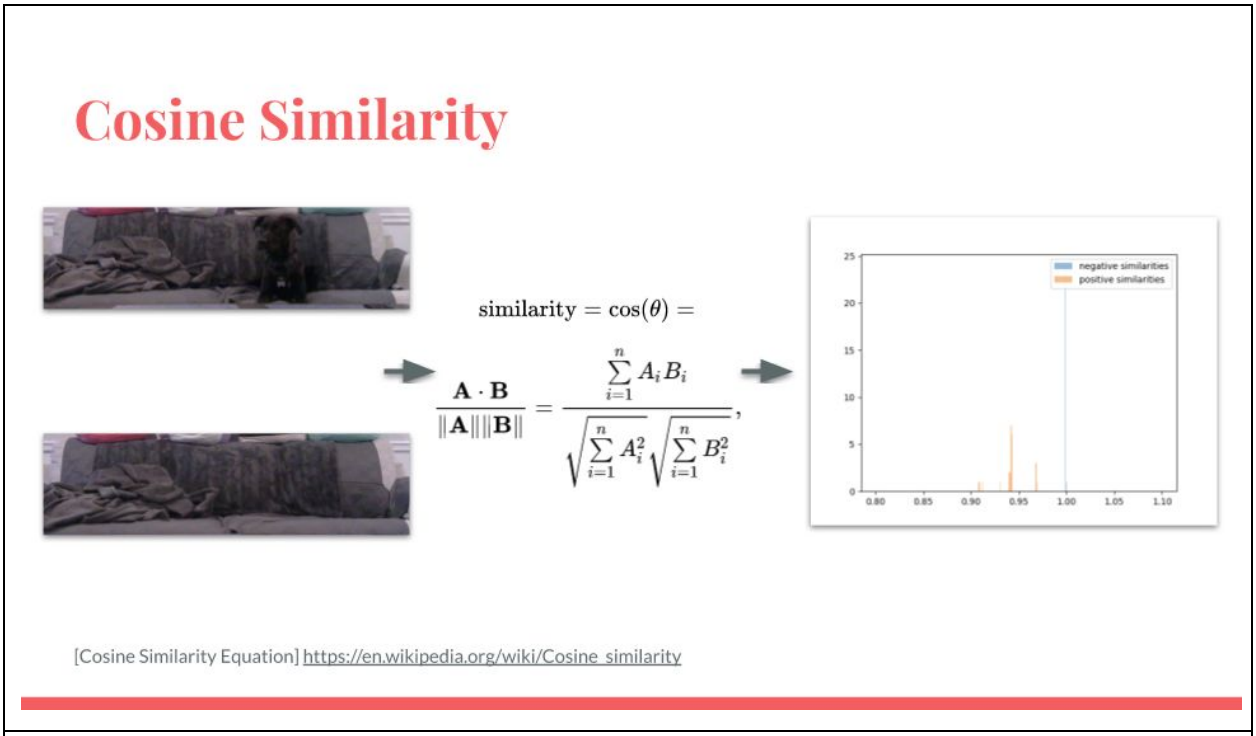

Figure 3. An illustration of cosine similarities calculated on two instances of an area of interest.

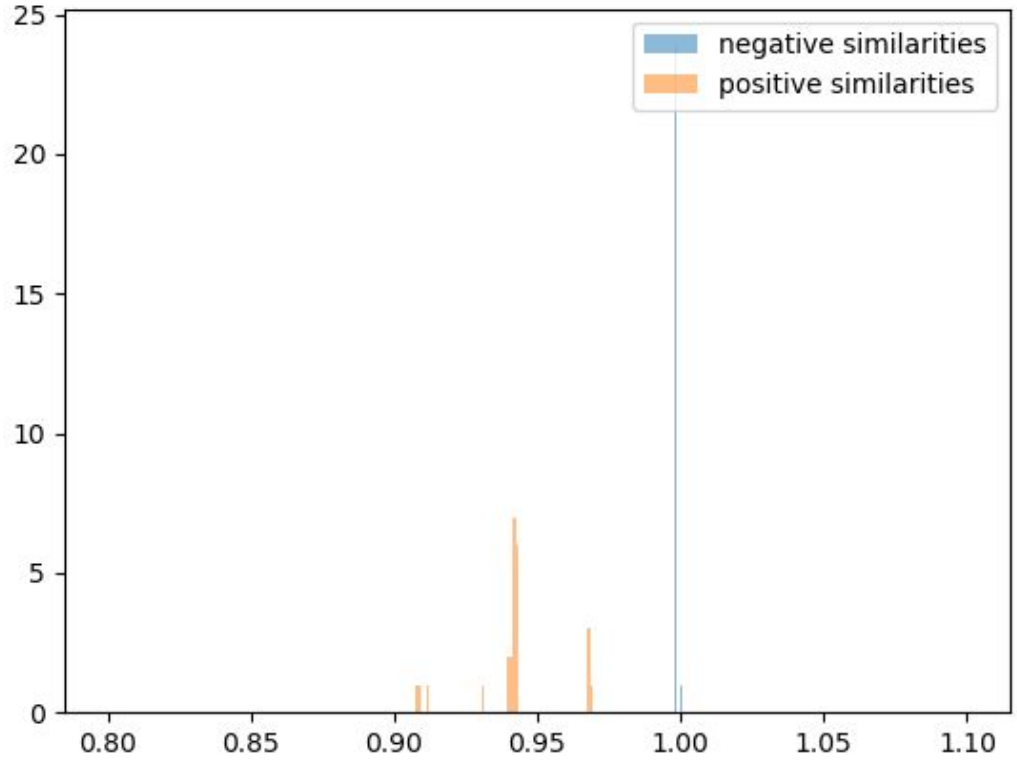

Figure 4. Cosine similarity experiments on empty (negative) scenes and non-empty (positive) scenes.

Figure 3 shows the initial approach to processing input images, toward the goal of detecting a pet entering an area of interest. First, raw images captured by the Pi Zero are sent over to the central Raspberry Pi over WiFi. Then, the images are essentially cropped so that only the area of interest is analyzed. An incoming image, following cropping, gets compared with a reference image. The way the system handled this comparison was by calculating a cosine similarity between an incoming image and a reference image. Finally, that similarity value gets compared to a predetermined threshold value in order to determine whether there is a pet. As seen in *Figure 4*, there is a clear difference between the similarity value calculated from an empty scene versus from a non-empty scene, when compared to an empty scene reference. From that fact, we were able to draw a threshold between the two.

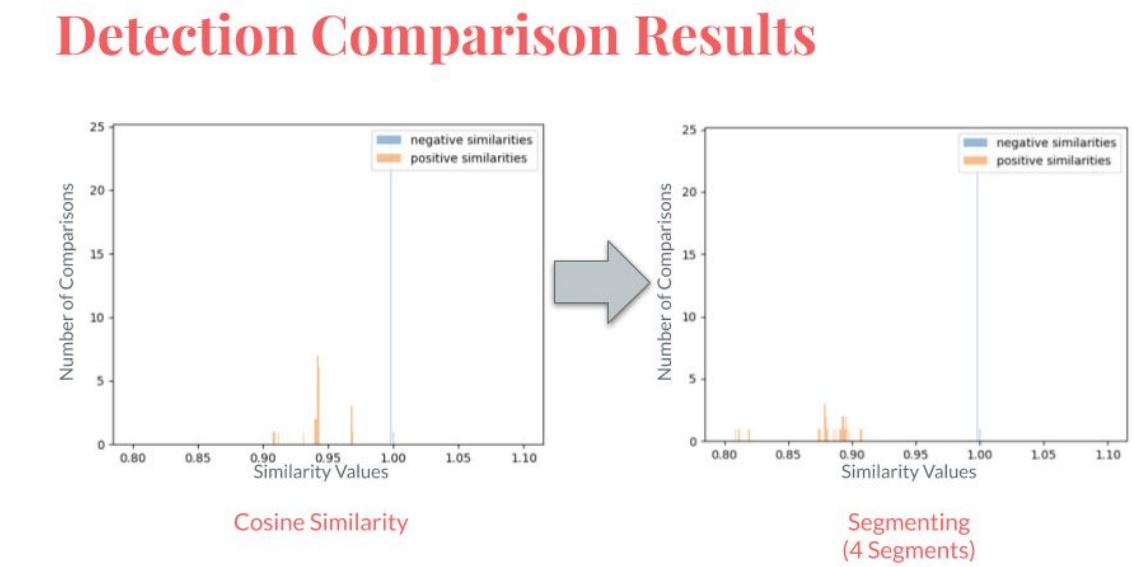

Figure 5. Improvements over cosine similarity by employing segmenting (4 segments).

Post Milestone Report, we iterated on the cosine similarity approach by employing segmenting. Taking area of interests, cropped from images captured by the Pi Zero's camera, we further divided the area of interests into segments in hopes of improving the threshold range between empty scenes and non-empty scenes. (Specifically, the distribution of similarity values are illustrated for four segments in *Figure 5*.) After dividing into segments, we applied cosine similarity to each segment and then took the minimum similarity value across all segments. As desired, the distribution of similarity values for non-empty scenes were pushed down, thereby widening the threshold range.

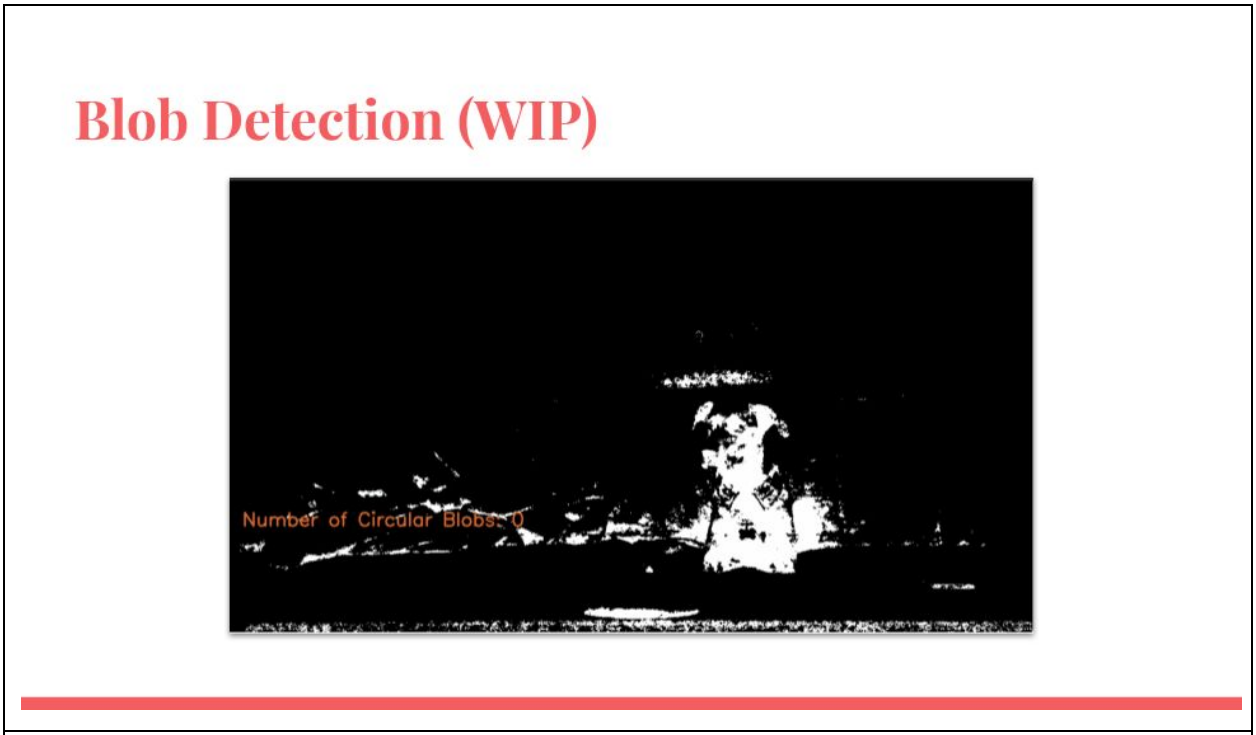

Figure 6. Progress made on blob detection. Note the splatters of white, which introduce undesired noise.

Similar to the motivation behind exploring segmenting, we also explored blob detection as a way of improving the detection algorithm and a way of possibly eliminating the threshold hyperparameter that is required in cosine similarity and segmenting approaches. Unfortunately, as illustrated by the white splatters in *Figure 6*, blob detection proved to be difficult on a black dog with a gray couch in the background. While we are able to visually see the dog (large white blobs), it was not sufficient enough for the algorithm to differentiate it from the other white splatters. With that being said, *Figure 6* seems promising to us. Moving forward, we could try other techniques within blob detection, which would hopefully yield better results. Additionally, there are a plethora of other approaches that could be tried to solve this problem. For instance, we could try to create a HARR cascade that can be used with OpenCV such as their provided cascade for finding human faces. Furthermore, we could attempt to use machine learning if our central system was more powerful and could handle large network sizes as the inputs are images.

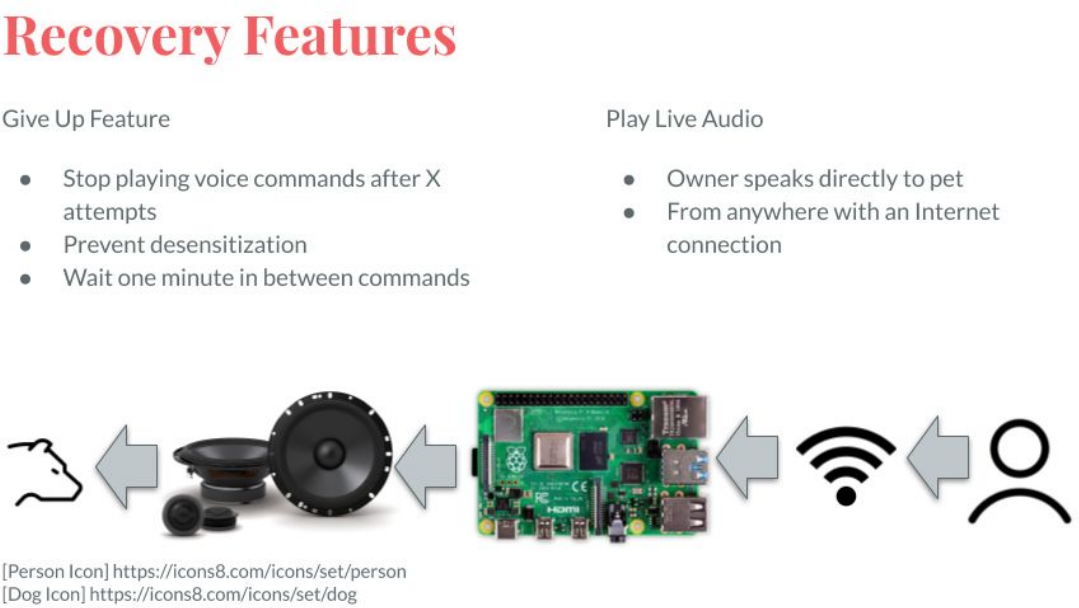

Figure 7. Two main recovery features, mainly intended to avoid command desensitization.

Beyond improving the detection algorithm, we also implemented two simple but effective recovery features. The “Give Up” feature essentially monitors how many times a command is issued to the pet following the system detecting a pet and then “gives up” by not playing the command after X attempts (set by user). The goal is to avoid desensitizing the pet. Additionally, the system also waits a minute between commands, also to avoid desensitizing the pet.

Now, after the system “gives up,” or at any time for that matter, it might be useful for the user to be able to directly speak to the pet from anywhere with an Internet connection. We implemented an end-to-end Python solution to address that potential need, which was intended to be connected to the web UI shown in *Figure 2*. However, in order to use that program on the web UI it seems it needs to be completely reimplemented in a different language.

Results

For our experimental setup, we performed 20 different tests all for 3 hour long durations. We tested our system for the use case of teaching a dog to stay off of the family room couch while the owners are away. The dog started with substantial training already in that when the owners are home, he never goes on the couch. However, when the owners are not around, he believes he’s allowed on the couch. Therefore, the goal is to further train the dog that he is not allowed on the couch all the time by using AutoPet to monitor and automatically send commands to him.

To test this use case, the setup was as follows: First, the pet would be put through his normal morning routine. Then, we would set up the camera to monitor the couch and connect the bluetooth speaker to the central Pi. We would then launch the main thread of our system on the central Pi making sure there are no obstructions in the first key frame. If there is, we would reset the system. Once the system was running, everyone in the house would leave for 3 hours. The 20 separate tests were spread

out over an entire month with 24 hours in between each test to accurately monitor the progress of the training and not capture “in the moment” successes. We define “in the moment” successes as the pet catching on to the training for a brief amount of time, but may not respond accordingly given a sufficient time horizon. For the first 10 tests, Cosine similarity was used for detection. The subsequent next 10 tests were done with segmenting and Cosine similarity for detection.

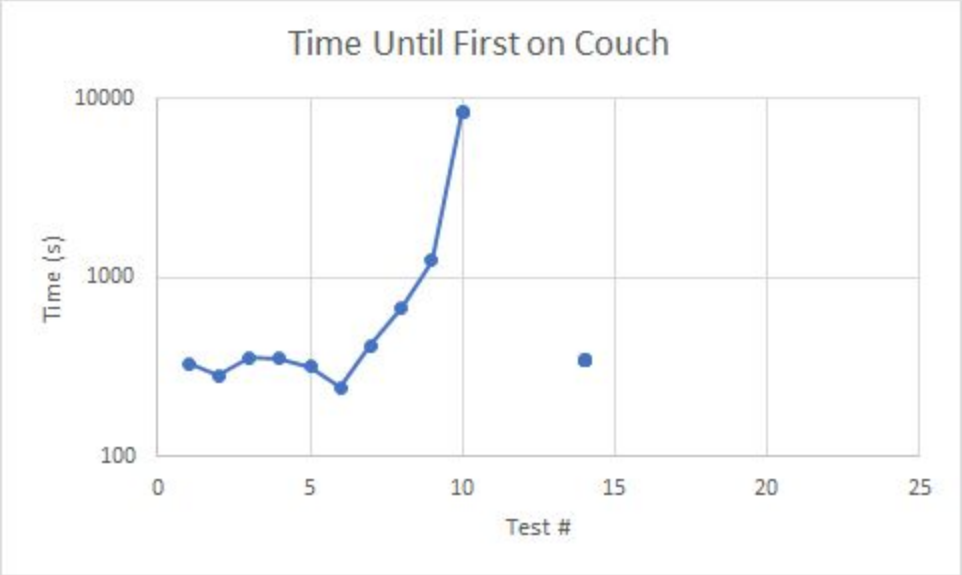

Figure 8: Amount of time in seconds until the pet got on the couch for the first time as testing went on. Missing data points indicate the dog did not go on the couch during the entire 3 hour period.

Figure 8 shows the graph of the amount of time before the pet went onto the couch for the first time. Any missing data points indicate that the pet did not go on the couch during the 3 hour test duration. As the results show, as the testing went on, the pet was clearly making progress on learning to stay off of the couch while the owners were gone. During the first 5 tests, it took merely minutes for the pet to decide it was okay to go on the couch. However, as the tests continued, it took longer and longer for the pet to give into their desire to go on the couch. Eventually, by the 11th test, the pet had learned to stay off of the couch while the owners were away. There was only one slip up during the 14th test where the pet went back to its old habit. This data demonstrates how an automated system for monitoring pets and sending them automated commands has potential to help owners change their pet’s behavior while they are not around.

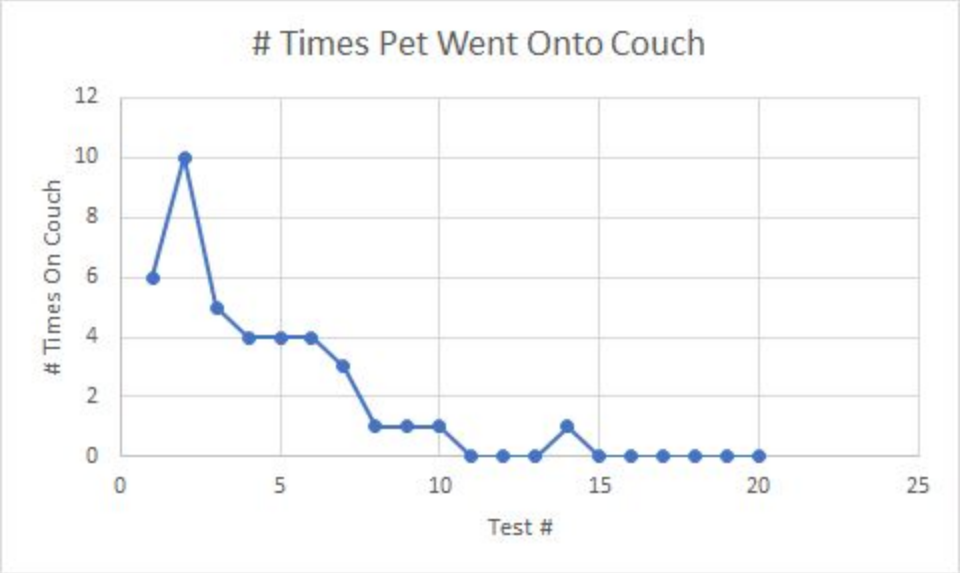

Figure 9: Number of times the pet went on the couch during each 3 hour test as the tests went on.

Figure 9 shows a graph of the number of times the pet went on the couch during each 3 hour test. The graph also shows the trend as testing went on. The data demonstrates that the pet was clearly successfully learning to stay off of the couch. As the testing continued, the pet continually went on the couch less frequently, eventually staying off of the couch entirely.

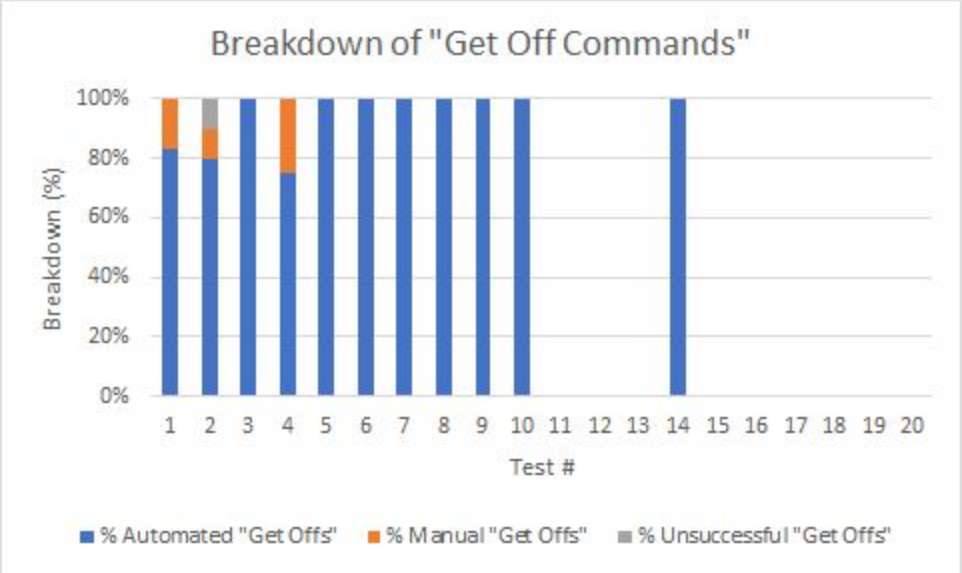

Figure 10: Breakdown of the “Get Off” command. Blue indicates the percentage of the time that the automated command successfully got the dog off the couch. Orange indicates the percentage of the time the user was manually able to get the dog off of the couch when the automated command failed. Grey indicates the percentage of the time that the pet ignored the system and the user was unable to get them off the couch. Any blank bars indicate that the pet did not go on the couch.

Figure 10 shows the breakdown of attempting to get the pet off of the couch. As the results indicate, the system is able to successfully get the pet off of the couch in an automated fashion most of the time. When the automated command through the speaker does fail, the owner is also able to attempt to manually get the pet off by manually speaking live through the bluetooth speaker. This method is also extremely successful. There was only one instance throughout the entirety of testing that the pet completely ignored any attempt to get off of the couch. This can greatly be attributed to the fact that he was playing with a toy on the couch that had his full attention. One thing to note here is that although this particular pet was very responsive to voice command through a speaker. Other research indicates that animals in general do not respond to speakers [8]. This is an interesting study for the future by testing this system on more pets. Another future direction might be to try training pets to respond to speaker commands first.

Justin Morris
Brendon Chen
Minh Tran Quoc

Milestones

Gantt Chart

https://docs.google.com/spreadsheets/d/1wwYvBDhA1BQncHgGeZM8kFhjlqQpZMWWWALKRnebNus/edit?usp=sharing

In the Gantt chart linked above, “Version 1 (Old)” contains our original milestones and deliverables and “Version 2 (Active)” contains our revised milestones following the milestone report. Our original focus involved achieving an MVP and then iterating on the MVP by adding additional functionality by adding on new active and passive sensors, such as microphones and pressure sensors. We achieved our MVP a bit before writing our milestone update, which allowed us to reconsider our main thrust around that time. We determined that there was more value both to the system’s performance and our education by shifting our focus to iterating on our MVP by improving on existing functionalities. For example, we reasoned that by adding more passive sensors, we would effectively be solving the same problem as the problem we solved initially when adding the camera, which was taking in some input and then making a determination.

As a result, we decided to refocus our trajectory toward improving the robustness of our minimum viable product (MVP) and expanding on it rather than adding a fundamentally redundant feature (a pressure sensor). We captured this objective change in two stretch goals: *1) Add recovery features to assist owners when the pets ignore the system* and *2) Create UI to easily interact with pets remotely*. As part of the recovery features, we decided to include robustness improvements and the ability for users to intervene by speaking through the system. The user interface aimed to provide a control panel of sorts to users. At first, focused on getting information from the system to the user. Then, we hoped to create an interactive mode that allows users to speak live with their pets, which tied into the recovery features idea.

The two additional features we did not complete were: connecting the live audio to the UI and blob detection. For live audio with the UI, we implemented two python programs to connect a client to the server and allow the client to speak through the bluetooth speaker. However, we found that there is no easy way to call these programs with a button on an HTML UI. Therefore, to add this feature to the UI involves reimplementing the base code. As we had a working version of the feature without a UI, we put this into the low priority tasks. For blob detection, the issue was that we started with simple blob detection using color filters. Therefore, because the couch is similar in color to the dog as well as the dog being black, so many shadows are also not filtered out, the color filter blob detection did not seem to be as robust as the cosine similarity implementations. However, we could try other more sophisticated blob detection algorithms that don’t rely on simple color filtering.

Conclusion

In conclusion, we have built a system to automatically monitor and react to pets while their owners are away to help assist in training. We tested our system on the use case of keeping a pet off of the couch and found that the system performed well successfully getting the dog off the couch with automated commands 90% of the time. With manual attempts when the automated commands failed boosting that percentage to 97.5%. Additionally, throughout the month of testing, the pet clearly showed improvement in taking significantly longer to first go on the couch and also going on the couch less during the 3 hour tests. This eventually resulted in the pet no longer going on the couch at all except for the rare occasion. However, this was one use case with one test subject. It has yet to be proven that the automated system will work universally with most pets. Additionally, there may be more research needed into proving that pets that don't initially react to the system can be trained to react to it.

References

[1] “Furbo Dog Camera,” Accessed: Jun. 12, 2020. [Online]. Available: https://shopus.furbo.com/.
[2] J. S. Kim, G. H. Lee, and D. J. Lee, “Pet training device,” 6598563, Jul. 29, 2003.
[3] Y.-C. Liao, “Pet training device,” 6799537, Oct. 05, 2004.
[4] C. Kuh, S. I. Lee, A. J. Andreae, S. M. Nonis, M. M. Rice, and A. C. Dubbaneh, “Pet monitoring device,” D812499:S1, Mar. 13, 2018.
[5] *Dog Training Collar - Rechargeable Dog Shock Collar w/3 Training Modes, Beep, Vibration and Shock, 100% Waterproof Training Collar, Up to 1000Ft Remote Range, 0~99 Shock Levels Dog Training Set*. .
[6] P. Lally and B. Gardner, “Promoting habit formation,” *Health Psychol. Rev.*, vol. 7, no. sup1, pp. S137–S158, May 2013.
[7] shibashake, “Cesar Millan’s Positive Dog Training Techniques,” *PetHelpful*, Jul. 23, 2008. https://pethelpful.com/dogs/Cesar-Milan-Dog-Training-the-Dog-Whisperer (accessed Jun. 12, 2020).
[8] “Full Page Reload,” *IEEE Spectrum: Technology, Engineering, and Science News*. https://spectrum.ieee.org/automaton/robotics/robotics-software/dogs-obey-commands-given-by-social-robots (accessed Jun. 12, 2020).