Lego Robot and Red Mouse

Michelle Dong & Daniela Martinez Partida

Computer Science, Macalester College

COMP 480-01: Bodies and Minds: AI Robotics

Professor Fox

January 23, 2025

1 Introduction

Rodents are a part of everyday life, especially in crowded cities. We have all had experiences with mice in homes, train stations, and campuses. They are hard to get rid of and exterminators are expensive. We thought that it would be cool if we could use robots as a way of catching mice and other random moving objects. Having worked on various assignments in class with similar functionality, we knew that it was a project that was rather approachable. Modifying the Sturdybot to to better designed for mobility and object detection, we sought out to build a "cat-like" robot. We say "cat-like" because we want the robot to be very reactive to random movements of the mouse, follow the mouse, and search for the mouse using other "senses" if the mouse is not in view.

2 Background

Getting a sense of the project we wanted to accomplished, we looked at existing literature related to object-following robots, object detection and tracking, and robot localization.

Model-Free Detection and Following of Moving Objects by an Omnidirectional Mobile Robot using 2D Range Data, Ahmed, et al.

In this literature, the researchers worked on implementing a robot that serves to follow moving objects around. This was motivated by the possibility of robots to take care of the elderly, patients, and just serving people in general. The researchers used a model-free approach, meaning that moving objects are detected without the need for the target to be specified in advanced, with Lidar sensor to first find the moving object and then follow it. Without using vision, the researchers used Laser Range Finders to calculate differences in

movement and rather complex math involving matrices and even arc tangents to determine angles and distances. For us, while our topics are similar, their approach is way more complicated than what we really needed it to be. A takeaway from this literature that we thought seemed useful was the mentioning of extended Kalman filter for target position predictions.

We thought that a Kalman filter could be useful for more efficient following of our mouse so we looked into Kalman filters.

The Kalman Filter: A Tool to Help Robots See Through the Noise, Estay

In this blog post, we get a sense of how Kalman filters are useful for robot localization when there is uncertainty in measurements. In particular, the Kalman filter recursively predicts and measures the environment to eventually get close to the true values. From class, this is similar to the idea with Markov localization, where we predict and check, and with more information you are able to gain more accuracy. Looking at our project, the Kalman filter could be useful for determining the the speed and angle our robot should be going at to more efficiently follow the mouse.

Robotics Vision Processing: Object Detection and Tracking, Singh

In this piece, the author provides a good overview for how object detection and object tracking would work and the potential processes that you could take. This is a good introduction to some of the ideas we could encounter as we work on the project. For example, with object detection, given that objects can look very different at different angles, a solution could be using convolutional neural network (CNN), to train the robot into detecting the object from different viewpoints. In terms of object tracking, the author introduces several algorithms: mean shift, centroid tracking, and also Kalman filters to track location of the

object. Besides detecting the location of the object, the author mentions that we also need to determine the location of the robot itself. One method was using Simultaneous Localization and Mapping (SLAM), where the robot uses multiple sensors and filters to map it's environment. Using this method, our robot could have a better sense of where there might be obstacles or if there are quicker paths.

Robots track moving objects with unprecedented precision, Matheson

In this work, motivated by the ability to move objects with incredible accuracy for manufacturing and also search-and-rescue missions, researchers utilize RFID technology to locate objects. In essence, an RFID tag is put on the object that is being tracked, when a signal is sent from a computer or robot, the RFID tag will bounce back a special signal that can be used for localization. RFID tags work very well because radio frequency signals can work in cluttered environments and also through walls, which is even better than computer vision. While this is a great method for object tracking and the tasks that they are trying to accomplish, for our project this is pretty much cheating: seems pretty unrealistic for a mouse to have a RFID chip.

The Robat—A Robot That Senses the World and Maps It Using Sound, Like a Bat, Yovel and Eliakim

In this project, the authors sought out to build a robot that mimics the echolocation properties of a bat. Their robot is built with a speaker that emits sounds of ultrasonic frequencies and two outward facing microphones that listen for echos. Essentially, as their robot moves around in an environment, it emits sound and listens for echos as it goes in order to map distance of obstacles and turn to avoid obstacles. To determine direction of the obstacle, the robot compares sound value between the left and right microphone based on

time difference: "if the object is on the right the sound will arrive in the right ear first and then in the left ear". The authors went further into the project by additionally including object recognition. They utilized machine learning to train the robot to identify the echos of different objects and was able to achieve an accuracy of about 70%.

Even though we are using vision for our project, this project is extremely relevant to us because we want to be able to incorporate sound into the "senses" that our robot can use for object location. Interestingly and slightly foreshadowing, the authors emphasized the importance of how "[Bat] ears are not simple cylinders, like the microphones. They have a complex structure and many bats can also move them".

Robotic Systems Developed to Find Lost Items, Scarlett Evans

Massachusetts Institute of Technology (MIT) has many examples of larger scaled project that has a similar idea to our project. An example being RFusion, a robotic arm that can retrieve hidden items. The way RFusion works is by emitting radio frequencies via antenna that get reflected off a requested object, and the camera attached verifies to see if this is the correct object the user is looking for. The purpose of the antenna and the PixyCam are loosely similar, in which the PixyCam reflects light onto what is in front of it to measure distance. With enough training, we would be able to see if what we are seeing is the mouse or another object.

Deep Learning for Computer Vision: A Brief Review, Voulodimos, et al.

In this literature, the authors goes into the ins and outs of computer vision, focusing on the layers computers have to go into to transfer pictures to code. They highlight the training many programs need to do to be able to acquire accurate information, in doing this we figured out what important things we needed to focus on to get the robot to be able to

track the robotic mouse. There was a large focus on the need for image recognition as well as the layering and functionality that image processing holds. In this, the paper broke down on the different forms of image detection models and how they might be able to succeed more in one environment, like face detection. Although we didn't follow this paper in the form of testing our image detection, it was good to understand the patterns that get developed in image processing.

2.1 Other works

In doing our background research, we were quite ambitious, looking at CNNs for object detection, Kalman filters for target positioning predictions, and SLAM for robot localization. These concepts may work well but many of them are quite complex; requiring a good foundation to build off of. As such, we began with using what we have learned in class as a starting point

From our assignments, we have worked with Lego robots that used the PixyCam to detect objects of a predetermined color, robot that finds the opening of a box with the use of light, and Braitenberg reactive robots.

In the activity with the PixyCam we program the robot to detect an object of a certain color, turn so that the object is centered, and move towards it so that objects takes up most of the frame. This is essentially how we want to implement the chasing aspect of our project: once the mouse is detected should turn and move towards the mouse so that the mouse takes up the majority of the frame.

Similarly, in the homework assignment where we have the robot trying to get out a box;

we have a robot that has to detect light and move toward areas with more light until the opening is reached. This assignment is relevant to the approach of our project as it not only parallels our ideas for vision but also for sound.

Braitenberg reactive robots respond fast to changing environment by constantly updating the robot's behavior based on input values. In our activities, our robot would move forward when no obstacle is detected and stop if obstacles are detected. Using Braitenberg methods, we also looked into the scaling of speed based on how close or far the obstacle is. This aspect is particularly useful for our project because it allows our robot to have more variation in actions depending on where and how the mouse move. For example, if the mouse is moving really fast, our robot should be fast, and if the mouse is slow, our robot should also be slow.

3 Our Project

In this project, we had main focuses on object detection and following which included looking at other existing robotic methods related to moving objects, object detection and tracking, as well as sound-based localization. Due to time constraints we weren't able to make a project as extensive and big as other organizations, but tried to work within what we knew.

3.1 Robot design and functions

From the beginning we knew had to account for several features in order for the project to go smoothly. First, the design of the robot. When we received the toy mouse, we immediately noticed how fast the mouse was able to go compared to the bulky SturdyBot.

Taking inspiration from other ev3 robot designs, specifically EV3 Discovery designed by

Sanjay and Arvind Seshan, we modified our robot to be more lightweight. To implement vision for our robot, we attached a Lego Pixycam to the front of our robot and tilted it downwards, in order to best capture the mouse that is closer to the floor. To incorporate sound, we utilized NXT Sound Sensors. We used two sound sensors, one pointed outwards to the left and one outwards to the right. We also included the Lego Ultrasonic sensor in our design for the possibility of obstacle detection. Figure 1, shows the final design of our robot.

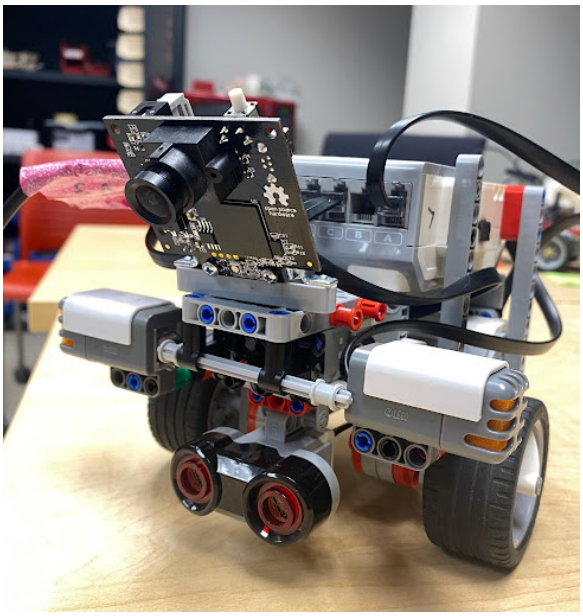

Figure 1

Our robot builds off of our SturdyBot class, meaning that it utilizes the same methods of: *turnLeft*, *turnRight*, *forward*, etcetera. With our robot, since we added the PixyCam, we have the additional functionality of getting the number of objects detected, X and Y-coordinates of the centerpoint of the detected object, and the height and width of the rectangle around detected object. From the inclusion of the sound sensor we can get the sound pressure. Interestingly, sound pressure ranges from 0 to 100, where the lower the number, the louder

the sound is.

3.2 Our method

We began by implementing our robot to recognize and move in the direction of the mouse. This process involved first having the PixyCam to detect our mouse. We taped our mouse using red tape and created a signature for red for the PixyCam using PixyMon. We chose the color red since it had the most contrast compared to our environment. Once this step was complete, we were able to detect the mouse, where the PixyCam will return the number of objects detected, X and Y-coordinates of the centerpoint of the detected object, and the height and width of the rectangle around detected object.

Having done background research on object detection, we knew that a better method for detecting the mouse was using machine learning and CNNs. By using CNNS our robot would be able to detect just the mouse and not everything else that was red. This is an limitation of our project, but considering the complexity of implementing machine learning into our project, and the relative gain to our simple PixyCam approach, it wasn't a path we pursued further.

Following the implementation from our in class activity, moving towards the mouse, we came up with the following pseudocode:

```
Def run:
    while true:
        get pixy cam input; count, x-cor, y-corr, width, height
        if mouse detected:
            Move so that mouse in centered in camera frame
            Go straightforward
        else:
            Stop
```

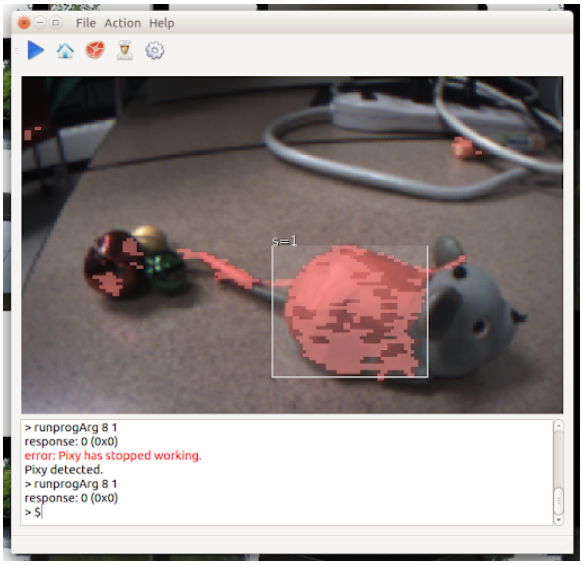

Figure 2: Sample view of mouse from PixyCam with bounding rectangle

This allowed the robot to turn and center the mouse within the frame and move towards the mouse. This worked great as long as the mouse was within the frame. When the mouse turned, the robot was able to turn in the same direction as well. Problems occurred when the mouse speeds up and is no longer in frame.

Our approach to tackle this was using the sound sensor. Essentially, our idea is that even if the mouse cannot be seen, it can still most likely be heard. As such, the robot should move in the direction of the sound. Following this path of reasoning we came up with the following pseudocode:

```
Def run:
    while true:
        get pixy cam input; count, x-cor, y-corr, width, height
        get left and right sound
        if mouse detected:
            Move so that mouse in centered in camera frame
            Go straightforward
        Else if mouse not in view:
            check if there are sounds of mouse around:
                compare left and right sound
                turn accordingly
                go straightforward
```

```
if no sound:
    give up; maybe there's no mouse :D
```

Our idea for how this was going to play out with our robot is shown in Figure 3.

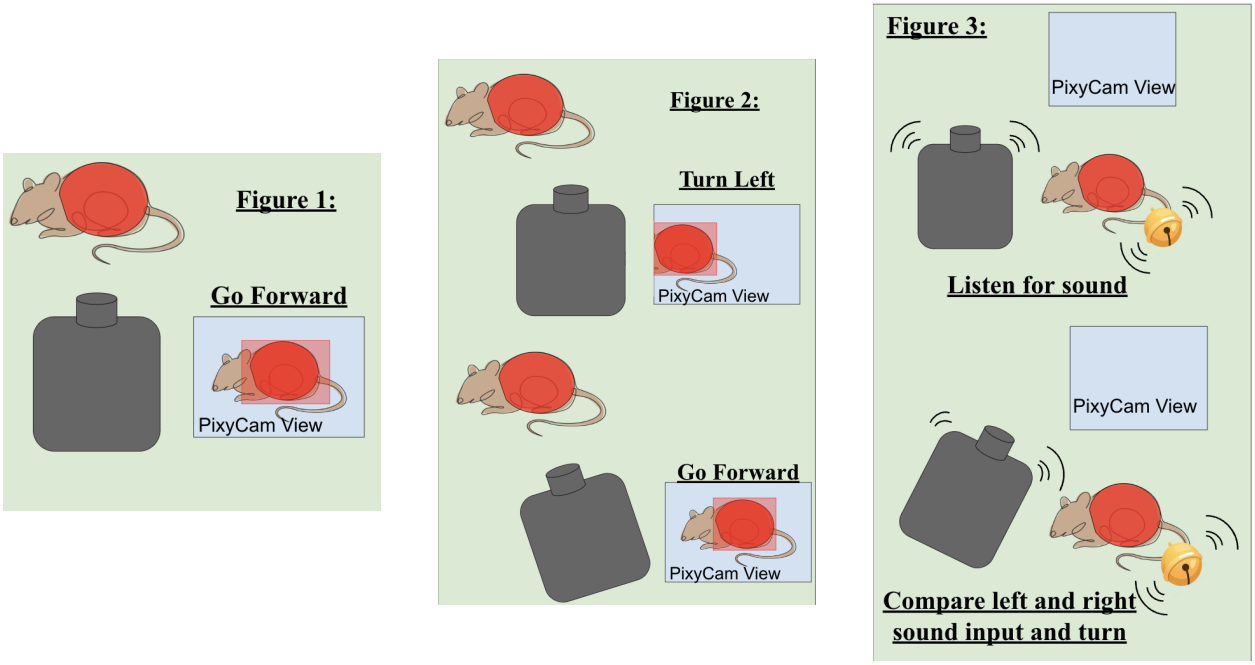

Figure 3

Our our idea was to implement Braitenberg speed scaling so that when the mouse speeds up, our robot speeds up as well. In this way, it is less likely that the mouse completely leaves the field of vision of our robot. We attempted to implement this process by looking at the Y-coordinates of the mouse and based on the differences in Y-coordinate between time t and time $t + 1$, scale the speed of the robot accordingly. This process theoretically should work and work well, but due to time constraints we were not able to successfully implement it. As such, unfortunately, our robot only moves at constantly speed.

4 Results

Overall, in our project we were able to implement object detection and object following, with quite a few caveats. Our robot can detect the mouse using the PixyCam and follow it well, when the mouse is in frame.

One of the main issues we faced was with the sound sensors. When the mouse is outside of the frame, our robot listens for sound from its left and right sound sensors and compare them to determine the next move. This however, was not as simple as we had imagined it to be. In particular, we found the results from the sound sensor to be weirdly inaccurate. When testing the sound sensors, we would play a sound directly next to the left sensor, but our output would show that the right sensor had a louder noise. We found the sound sensors to be consistently-inconsistent and we could not figure out how to get around this obstacle.

Due to time and knowledge constraints, ultimately, our project did not accomplish as much as we had hoped, especially with implementing features from our background research.

5 Limitations and further work

Aside from the processes mentioned in the background research that would have been cool and useful to implement, if we were to have more time we would have liked to implement three extra things: obstacles, multiple mice, and the possibility of changing the quality of our camera and sound sensors. We would have liked to add more obstacles to the robot environment so that it would seem to be more realistic to the real-world. The way that this would have been most likely done would be to avoid colors that aren't red. We would have programmed the SturdyBot to avoid all colors except red at a certain distance by turning,

and having the SturdyBot repeat that action until the SturdyBot could no longer move.

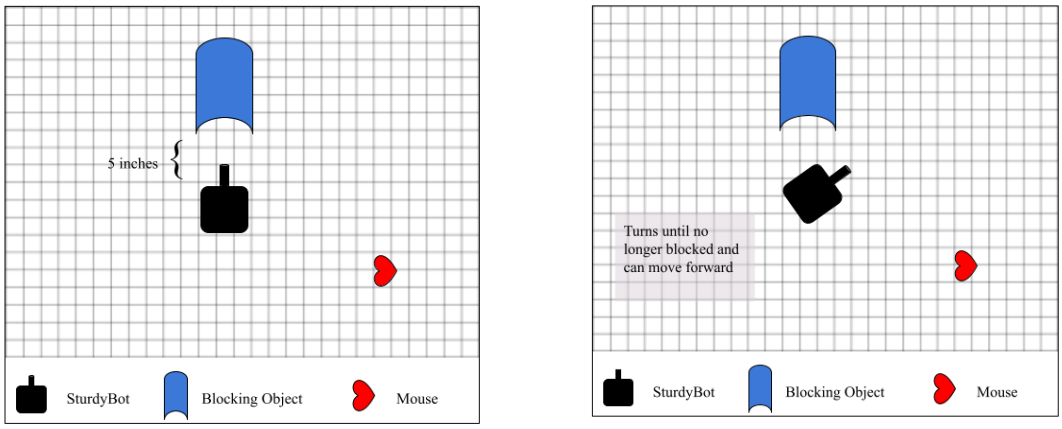

Figure 4

During testing, we noticed that the sound sensors would rarely work. They didn't have a strong difference when it came to hearing the mouse moving since its sound intensity that was getting picked up came out to be very similar. We first attempted to keep the sensors to be relatively close to the SturdyBot's block facing forward and realized they were too close to one another and that resulted in the sound intensity being too close so the SturdyBot didn't know where it was meant to go. We decided to have them facing opposite directions and to be slightly further apart. This helped in some situations, but overall we had learned that the sound sensors weren't sensitive enough to give us significant enough numbers to be able to work with. If we had better quality or even modern sound sensors, we would have been more dependent on using noise to make it easier to track the mouse. This would mean we would be able to track it if it was further away or hiding in a way that would make it more difficult to visually see by the camera. This also leads to wanting a better quality camera that has more realistic coloring to how humans see the colors. The quality of the PixyCam diluted the colors as well as had a grainy quality. This made it difficult to track

the robotic mouse since it at times wouldn't be able to pick up the red we wrapped it in. In cases like these, it would have been nice to have more sensitive sound sensors. We used NXT Sound Sensors to be able to detect the noises that the robotic mouse makes when it moves due to its motors and the bells attached. Our goal was to make the mouse as loud as possible to make it easier for the Sound Sensors to detect the mouse. NXT Sound Sensors are very dated and not as sensitive as we would have liked them to be for this project. They read the noise very similar to one another, causing the audio portion in this project to not be the most accurate or dependable. If we had stronger equipment, we would have had an easier time using audio to figure out where the robotic mouse is when the robot could not visually see it.

6 Conclusion

Although this project was much more a smaller scale project, we wanted to highlight the amount that we had learned. We had already used the SturdyBots in previous activities and projects in this class, but we were able to play around with these robots more. Our goal was to create a "cat-like" robot that was able to chase mice, and it does that to an extent! Although it might not work as well when the mouse is further away or there's too much of the same color surrounding the robotic mouse, it is somewhat similar to how we as people or cats see rodents. We might hear it, but not be able to pinpoint its exact location. We might catch a glimpse of it, and lose it due to its speed or camouflage. Like in real life, if we want to catch a rodent or a pest, we have to have a perfect series of events to happen to be able to get rid of it. In our results we have acknowledged that this project is far from perfect,

but with the help of better technology we hope that we would be able to keep better track of pests with robots.

7 Appendix

Project

Michelle	Vision Code	Referenced previous activities and homework assignments that had focused on SturdyBots vision abilities.
Michelle	Research	Read papers with a focus on object tracking using sensors and echolocation. Many of the papers were advanced for our type of project, but it provided us with different ideas we could implement if we wanted to expand on this project with differing equipment.
Michelle	Rebuilding SturdyBot	Adjusted some portions of the SturdyBot to remove portions no longer needed, as well as add parts we needed. We removed the gyroscope, but had added the NXT Sound Sensors as well as the PixyCam.
Daniela	Audio Code	Looked at tutorials related to Lego programming to get a grasp of available functionalities. Looked into projects related to Lego sound sensors.
Daniela	Research	Read papers that had focused on vision learning, related projects, as well as tutorials to help difficulties that arose with audio issues.
Daniela	Purchasing Item	Luckily since we didn't have to purchase robot parts for the SturdyBot, we only needed to purchase the automatic mouse.

Paper

Michelle	Background, Our Project, Results
Daniela	Introduction, Limitations and further work, Conclusion

8 Bibliography

Ahmed, S. A., Topalov, A. V., Shakev, N. G., & Popov, V. L. (2018). Model-Free Detection and Following of Moving Objects by an Omnidirectional Mobile Robot using 2D Range Data. IFAC-PapersOnLine, 51(22), 226–231. https://doi.org/10.1016/j.ifacol.2018.11.546

Evans, S. (n.d.). Robotic Systems Developed to Find Lost Items — IoT World Today. Retrieved May 6, 2024, from https://www.iotworldtoday.com/robotics/robotic-systems-developed-to-find-lost-items

Matheson, R. (2019, February 19). Robots track moving objects with unprecedented precision. MIT News — Massachusetts Institute of Technology. https://news.mit.edu/2019/robots-track-moving-objects-unprecedented-precision-0219

PhD, D. S. E. (2023, April 25). The Kalman Filter: A Tool to Help Robots See Through the Noise. Medium. https://danielsepulvedaestay.medium.com/the-kalman-filter-a-tool-to-help-robots-see-through-the-noise-2687bf538cfb

Seshan, S. and A. (n.d.). Robot Designs. Retrieved May 5, 2024, from https://ev3lessons.com/en/RobotDesigns.html

Singh, D. (n.d.). Robotics Vision Processing: Object Detection and Tracking. Embedded Computing Design. Retrieved May 4, 2024, from https://embeddedcomputing.com/application

/industrial/robotics-vision-processing-object-detection-and-tracking

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep Learning for Computer Vision: A Brief Review. Computational Intelligence and Neuroscience, 2018, 1–13. https://doi.org/10.1155/2018/7068349

Yovel, Y., & Eliakim, I. (n.d.). The Robat—A Robot That Senses the World and Maps It Using Sound, Like a Bat. Frontiers for Young Minds. Retrieved May 4, 2024, from https://kids.frontiersin.org/articles/10.3389/frym.2020.00007