Closed-loop Predator-Prey System for Investigating Mice Pursuit Strategies

Leo Li

Abstract

The purpose of this project is to investigate the pursuit strategies of mice when hunting for prey. A system is developed where a mouse freely behaves in a square arena with a gantry-controlled chocolate chip. A camera is used to determine the position of the mouse and chip and then sends commands to a gantry that allows the chip to escape from the mouse. The latency of the system is determined to be 132.6±14.4ms and can produce robust pursuit behaviors when interacting with mice. Mouse exhibited mostly classical pursuit strategy when attempting to catch the prey, as well as some preference towards the arena wall and some anticipation for chip movement. Observations from the experiments provide insights into the mice's engagement, behavior, and strategies during pursuit.

Introduction

How animals hunt prey in the natural world is a complex task requiring sensorimotor control, planning and navigation, and sequences of real-time decision-making. While investigating the pursuit strategies of animals can be challenging in the natural environment due to the lack of tractable methods, the same investigation couldn't be conducted under head-fixed conditions as constrained animals do not hunt. We find an intermediate where the animal – the mouse in this case – is freely behaving in an empty arena while a gantry-controlled chocolate chip escapes from the mouse when it comes close. Such an experimental setup not only gives the mouse ample space to exhibit natural behavior but also allows the experimenter to choose specific escape policies of the prey based on hypotheses of the mouse's pursuit strategy, something that isn't possible with natural prey. In setting up such a system, the main objectives would be to achieve low latency, robustness to animal behavior, flexibility in implementing escape policy, and logging experiment state for downstream analysis.

Results

System Schematic

Figure 1a shows the overall schematic of the system. The mouse freely behaves in a rectangular arena of size 483mm by 454mm (Figure 1b). The "prey" is constructed by gluing magnets on top of a flat piece of plastic, which will be referred to as "puck" in the rest of the report. The puck is

colored red with nail polish for easier tracking with the camera, and a piece of chocolate chip is glued on top of the puck to motivate the mouse to pursue the puck (Figure 1c).

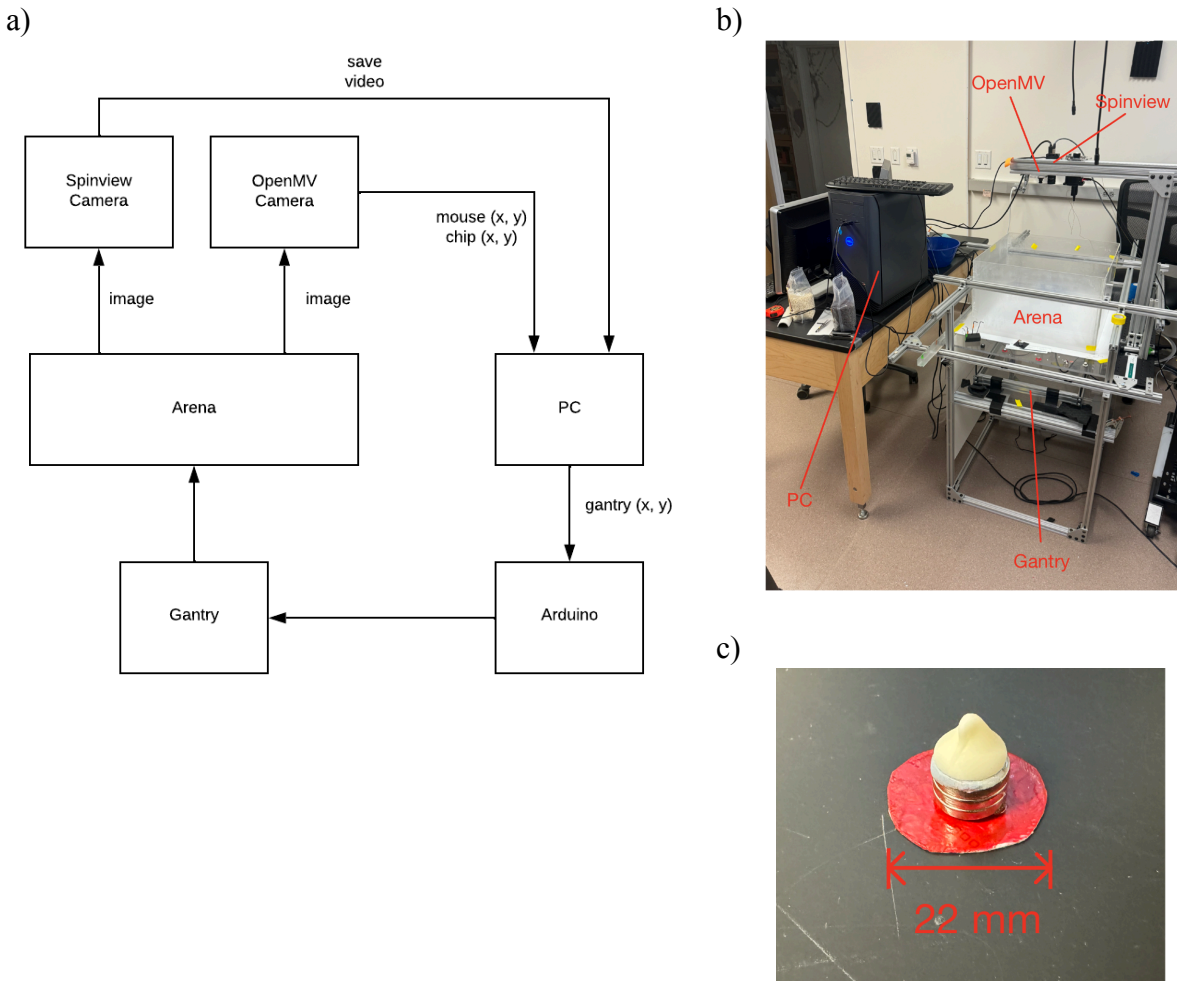


Figure 1: Experiment Setup.
(a): System schematic. **(b)**: Picture of the experiment setup. **(c)**: Picture of the puck with a chocolate chip on top.

Latency Test

The first step is to ensure the system is fast enough for the task. A separate latency test was performed to determine the time delay between an event in the arena and a command to the gantry being executed. To test for the closed-loop latency, an iPhone is placed in the arena that plays a video that oscillates red and green light at a frequency of 1Hz. The OpenMV camera captures the image and tells the PC whether the phone is currently displaying red or green. The PC then commands the gantry up or down depending on the color, resulting in the gantry actuating periodic motion that is lagged behind the phone's color (Figure 2a). The phase difference between the gantry's up/down motion and the phone's red/green switching is the

closed-loop latency of the system, which can be calculated offline using the video data recorded by the Spinview camera.

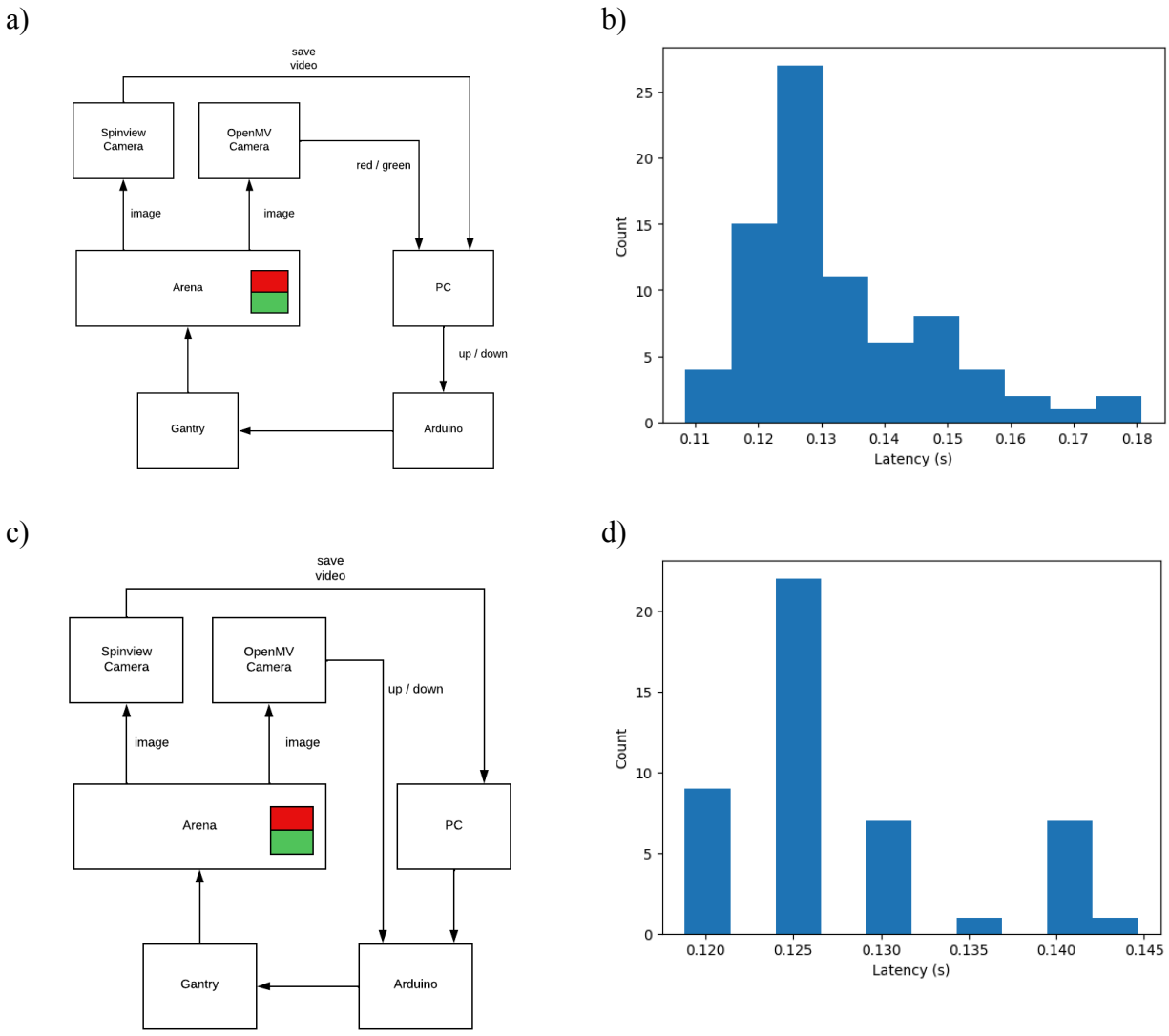


Figure 2: Latency Tests.
(a) and (c): System schematic for two latency tests, where the main difference is in (a) the OpenMV camera is connected to the PC, while in (c) it's directly connected to the gantry.
(b) and (d): Latency histograms corresponding to the tests under (a) and (c) respectively, with (b) having a latency of 132.6±14.4ms (mean±std) and (c) having a latency of 126.9±7.0ms.

The resulting latency is 132.6±14.4ms (mean±std) (Figure 2b), which should be sufficient as mice's reaction time ranges from 150-400ms (Goldstein, 2022). Several sources cause the variation of latency, the first of which is a direct result of the bin size. In this test, the OpenMV camera takes a picture every 20ms. As a result, any event occurring within the gap in which a picture is taken would need to wait for the next picture to be captured. Thus, the latency caused

by bin size would be a minimum of 0 ms and a maximum of 20 ms, corresponding to the width of the histogram (Figure 2b and 2d). Other sources that cause additional latency variability might arise from the Windows scheduling system, where a command sent to a serial port might not occupy the highest priority and thus additional time is consumed.

Another possible configuration of the system is shown in Figure 2c, where the OpenMV camera directly commands the gantry through RX-TX pins. In this configuration, the latency is 126.9±7.0ms (Figure 2d), which is slightly improved compared to Figure 2a, but with much less variability. The small improvement in average latency compared to Figure 2a suggests that the majority of latency comes from the actuation of the gantry. The much lower latency variability suggests that the additional step of going through the PC is the likely cause of the wider distribution in Figure 2b. However, such a configuration in Figure 2c makes the system more difficult to log relevant information and the difficulty doesn't justify the marginal improvement in latency, so we settled on the system configuration of Figure 2a for the remaining experiments.

Computer Vision and State Estimation

To determine the mouse and puck position from the camera, a computer vision pipeline of 4 steps is implemented (Supplementary Materials). First, lens correction is applied to remove the curved distortion of the image due to lens effects. Then, a homography transform is applied to the image such that the four corners of the arena are located at the four corners of the image. After these two steps, the pixel x-y coordinates of the picture conveniently correspond to the millimeter x-y coordinate of the arena by a constant multiplication factor. After applying separate color filters for the red puck and the black-ish mouse, a blob detection algorithm with area thresholding was run on the image to determine the x-y coordinates of the mouse and the chip respectively. Lastly, the determined positions of the mouse and puck are then sent to the main PC through a microUSB to USB cable (Figure 1a). The entire image processing pipeline takes roughly 45ms to complete.

For more robust tracking of the puck position, a Kalman Filter (KF) is implemented that takes into account both the gantry commands and the camera measurement to estimate the position of the puck (Simon, 2006). Since the latency of the system is about 130ms and the bin size between camera frames is 45ms, the prediction step of the KF uses the gantry command from 3 frames ago to update the puck position. In the case where the camera successfully detects blobs and has a readout of the chip position, the posterior update step of the KF is then performed to update the chip position. As for the mouse position, the blob detection being done by the camera is sufficiently robust that it doesn't require any further software processing. Lastly, to determine which direction the puck should escape, all commands to the gantry from the past 3 frames are summed to predict where the puck will be when the current frame gantry command is executed. This new predicted position of the puck is then used to calculate the gantry command using the escape policy.

Escape Policy

The escape policy determines what action the puck will take given the state of the arena. Here, we employ the most simple strategy of the puck running in the opposite direction from the mouse under constant velocity (Figure 3a). The escape vector (direction puck is running) will be a normalized difference vector (puck position minus mouse position). Under this escape policy, if the mouse were to pursue the puck under naive pursuit (running directly at the puck), the resulting trajectory should be that both the mouse and the puck run in a straight line.

There are modifications to the escape policy at arena boundaries and corners to ensure the puck stays in the zone where the gantry can reach. When the puck is at an edge, the component of the escape vector that's perpendicular to the edge is truncated so the puck moves parallel to the edge (Figure 3b). If the puck reaches a corner, the escape vector will be whichever edge from the corner is further away from the mouse. Computationally, this corresponds to the sign of the angle between the difference vector and the corner bisection line, which can be determined by projecting the difference vector onto the tangent vector of the corner (Figure 3c).

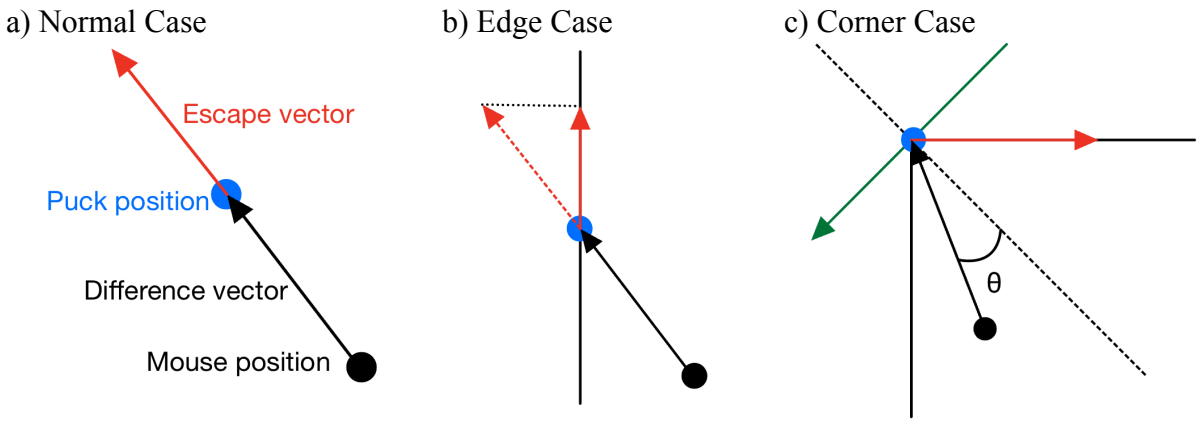


Figure 3: Escape Policy Depiction.

Note that for all three cases described, the magnitude of the escape vector is always normalized so the speed at which the puck is escaping is always constant for all three cases. In addition, the puck will only escape when the distance between the mouse and the puck falls below a threshold, otherwise, the puck doesn't move. As a summary of the system's algorithm, pseudocode can be found in Supplementary Materials.

Behavioral Results

Most of the collected data are in video format, and limited analysis has been done on the recorded videos. Example videos can be found here. This section will discuss qualitatively some of the behaviors observed.

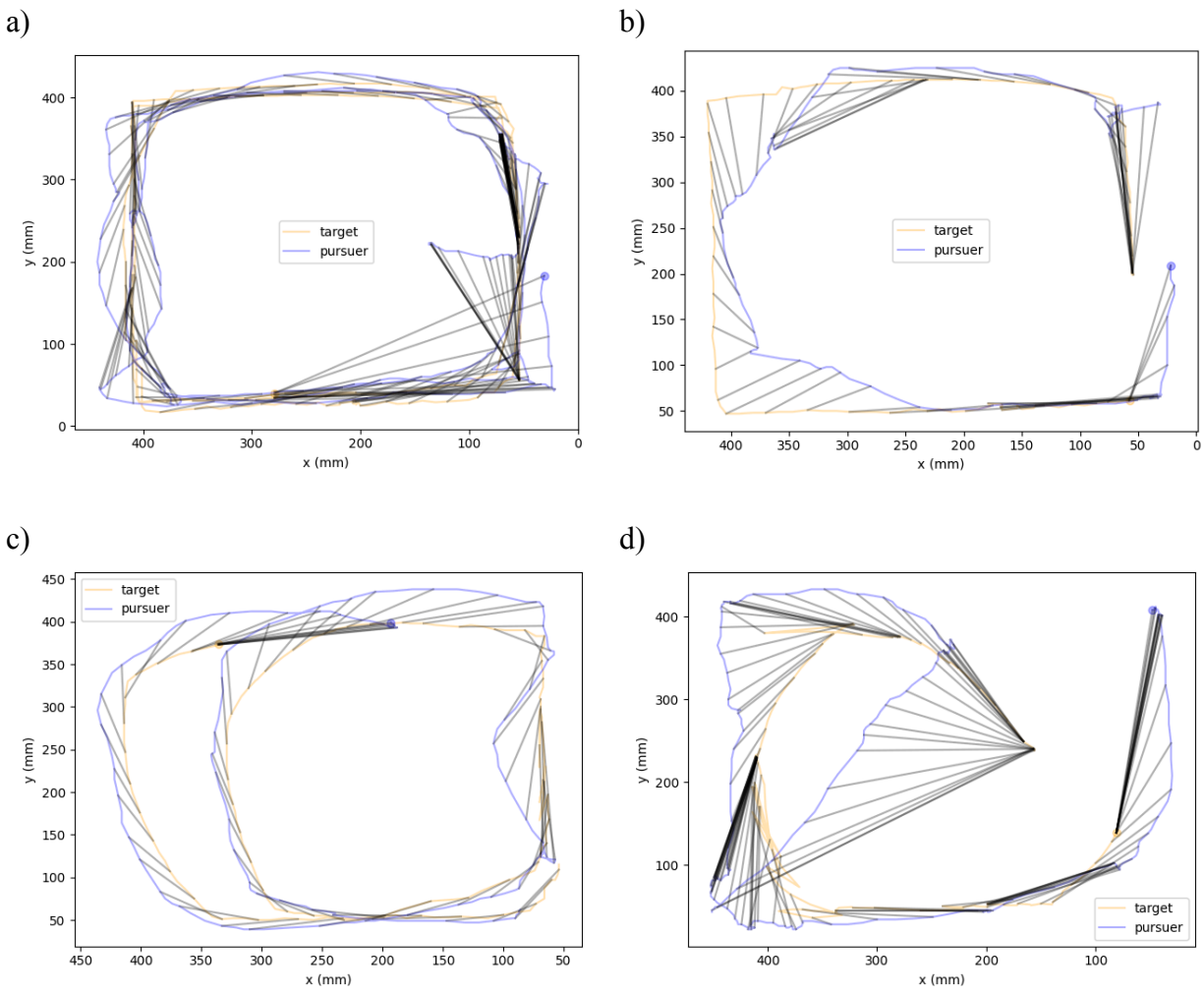


Figure 4: Example Trajectories.
(a): Mouse pursuing puck for three cycles in a clockwise direction. The mouse starts near the 3 o'clock position. **(b)**: Example trajectory where mouse anticipates puck turning near the bottom left corner. **(c)**: Mouse starts pursuit near the 12 o'clock position and chases counterclockwise for two cycles. **(d)**: Example trajectory showing punctuated dashes. The clustered black lines indicate the mouse taking a pause, and the spread-out black lines indicate the mouse moving rapidly and taking bouts.

Robust Long Trajectories

Figure 4a shows an example trajectory of the mouse chasing the puck, where the mouse chased the puck clockwise for three cycles. The puck showed robust turning behavior when near corners, and the mouse sometimes took a bite of the chocolate chip and paused briefly. Most trials end with the mouse successfully detaching the entire puck from the gantry, and the closed-loop system is stopped manually. Example videos can be found here.

Increased Engagement with Speed

As the speed of the gantry was increased from 1000 to 3000 (gantry feed rate), the mouse appeared more engaged in pursuing the puck. For speeds below 2000, it's frequently observed that the mouse wanders around in the arena only occasionally walking within escape distance of the puck, but doesn't engage in pursuit even when the puck is escaping away. The motions that resemble the mouse pursuing the puck are better described as sniffing and following the puck rather than pursuit. For speeds above 2000, the mouse is much more likely to start pursuit once stumbling into the escape range of the puck, and as the puck moves the mouse turns and starts a pursuit. For speeds at 3000 and 3500, the mouse shows very few instances of successfully detaching the puck from the gantry. The longest trajectories last around a minute ending in the mouse being distracted and giving up. Videos that demonstrate this can be found here.

It's interesting what leads to this behavior, as this suggests it's not only the chocolate chip that's motivating the mouse's pursuit. There were multiple instances during the experiment where the mouse was pursuing a puck with a chip when a piece of another chip was sitting on the ground in the arena. One possible hypothesis is that the faster motion of the puck triggers a visual reflex that leads the mouse into pursuit.

Preparing for Turns

One mouse – SC22 – displayed behaviors indicating it anticipated the chip's turning movement (Figure 4b). When it's chasing the puck near a corner, it would turn towards open space and try to catch the puck on its path escaping from the corner. Such behavior doesn't happen for every corner encountered but can happen multiple times in a single pursuit. Example videos can be found here.

Bias Towards Wall

For both mice, many trajectories take the shape of a rounded-corner rectangle, where the mouse prefers to be closer to the wall of the arena which results in the chip being chased further away from the edge. An example is shown in Figure 4c, where for parts of the trajectory the mouse is positioned at an outer circle of the puck. The resulting trajectory is one where the mouse pursues the puck from an outer circle while the puck escapes along an inner circle. This could be partly explained by a system artifact where the mouse will appear further from the center of the camera's field of view due to it being taller than the puck. However, the phenomenon is also accounted for partly by the mouse leaning closer to the wall during pursuits and chasing the puck away from the edges. Example videos can be found here.

Other

Some other behaviors are less robustly displayed. SC23 showed interest in an empty puck without a chip attached to it and pursued the empty puck for 30 seconds, suggesting that the

pursuit behavior had a reflexive or habitual component rather than motivated by the food reward. Both mice also showed circular surrounding motion before the pursuit began as if scouting the surroundings to ensure prey is safe for pursuit. Finally, SC23 showed pauses between quick dashes towards the puck, as if trying to beat the puck by reaction time (Figure 4d). These are observations that require more data to validate.

Discussion

Here we present a closed-loop system controlling prey in real-time producing robust pursuit behaviors of mice. Long and diverse trajectories of mice pursuing prey were observed. Such a system sets the foundation for further investigation of pursuit strategies employed by mice and enables downstream data analysis of the mice and puck trajectories. However, the main shortcoming is that only two mice were used to test the system and only naive escape has been implemented.

Several additional escape policies test the hypothesis of whether mice employ naive pursuit or proportional pursuit. Instead of having the puck escape in the same direction away from the mouse, a fixed angular offset can be added to see if the mouse can learn this fixed offset. In addition, the future position of the mouse can be used instead of its current position for feeding into the escape policy. Moreover, a terminal condition should be set where the system declares the mouse has successfully captured the prey, which will help with downstream data analysis as pursuits will have a well-defined terminal condition. Deeplabcut can be used to further analyze the video data, as well as fitting models of different pursuit strategies and make quantitative claims about mice pursuit strategies.

Reference

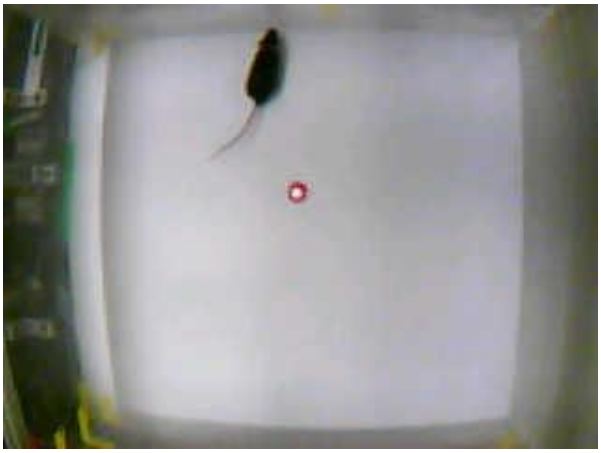
Goldstein S, Wang L, McAlonan K, Torres-Cruz M, Krauzlis RJ (2022) “Stimulus-Driven Visual Attention in Mice.” *Journal of Vision* 22 (1): 11. doi:10.1167/jov.22.1.11.

Simon D (2006) *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Hoboken, NJ: John Wiley & Sons, Inc.

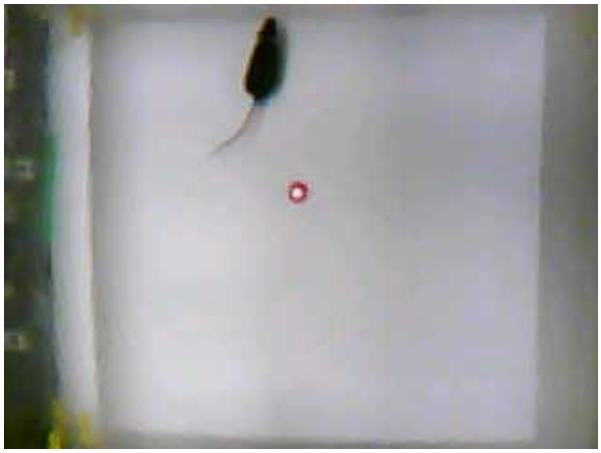
Supplementary Materials

Computer Vision Pipeline

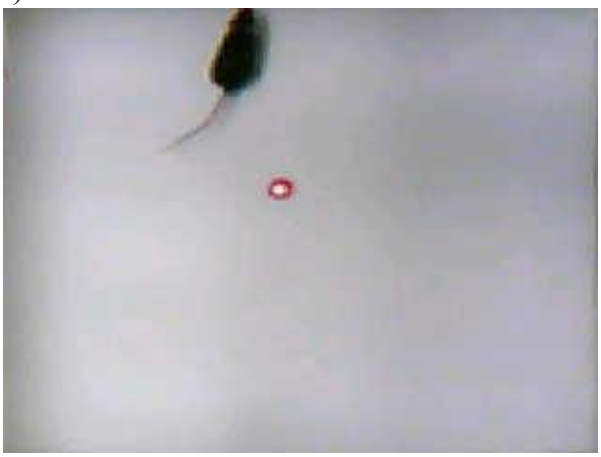
a)



b)



c)



d)

Supplementary Figure 1: Computer vision pipeline
(a): Raw image. **(b)**: After lens correction. **(c)**: After rotation correction, mapping the corner of the arena to the corner of the image. **(d)**: After blob detection of both mouse and puck.

Pseudocode

```
While True:
        mouse_xy, chip_xy = from_camera()
        chip_KF.predict(u=gantry[-3])
        chip_KF.update(measurement=chip_xy)
        predict_chip_xy = chip_KF.chip_xy + sum(gantry[-3:0])
        escape_vector = predict_chip_xy - mouse_xy
        correct for corner and edge conditions
        if length(escape_vector) < threshold:
                gantry_cmd = escape_vector * velocity_correction
```