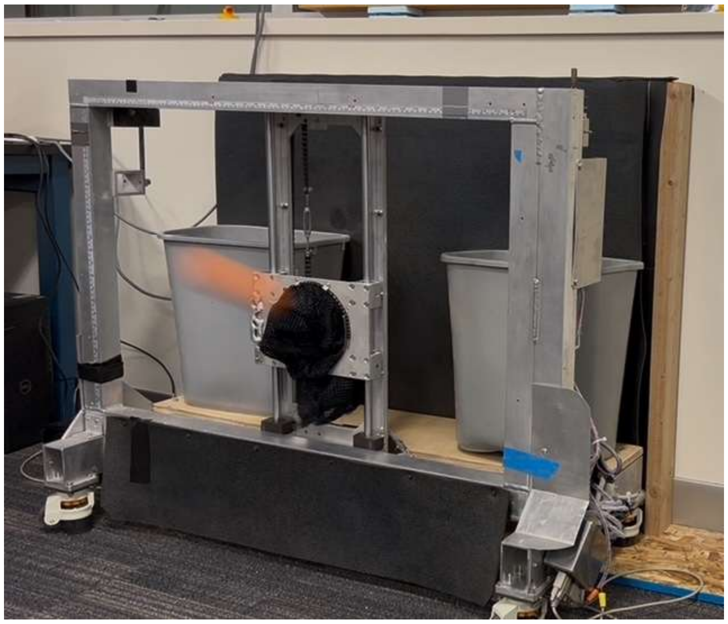
# Baseball Catcher

For this project, the goal was to catch projectile baseballs. Two flea cameras were set up side-by-side behind and above the catcher to record the motion of the ball, which was launched from a machine behind a black curtain.

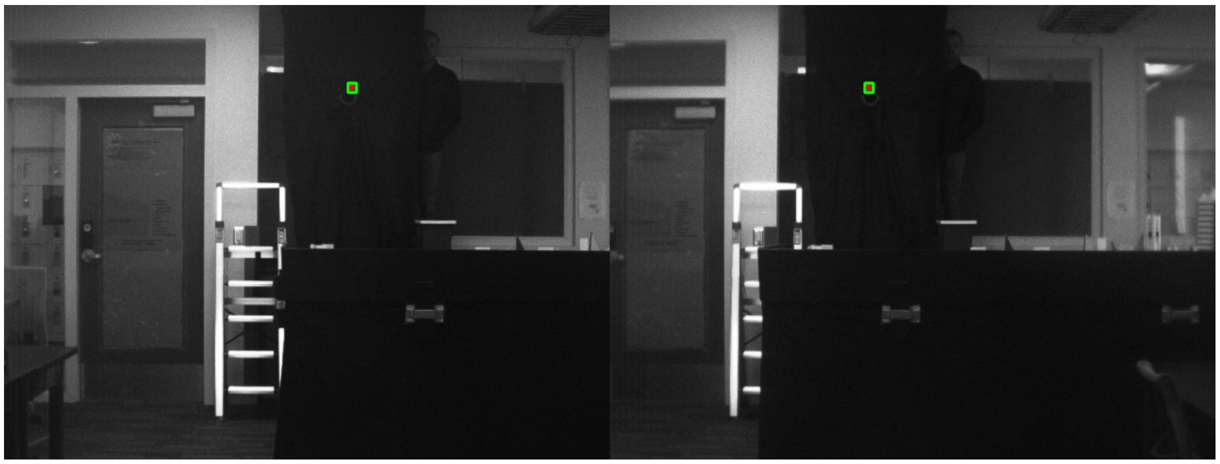
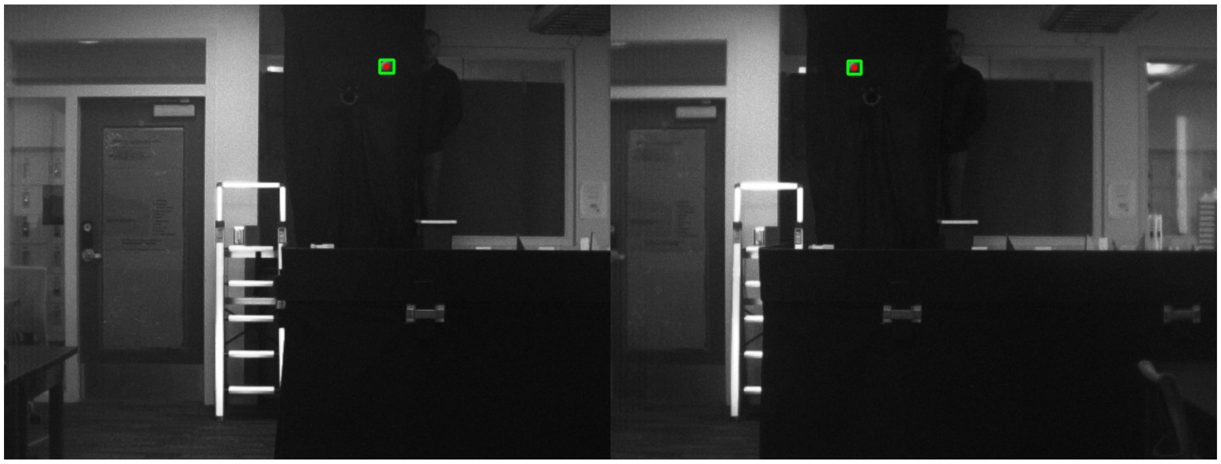
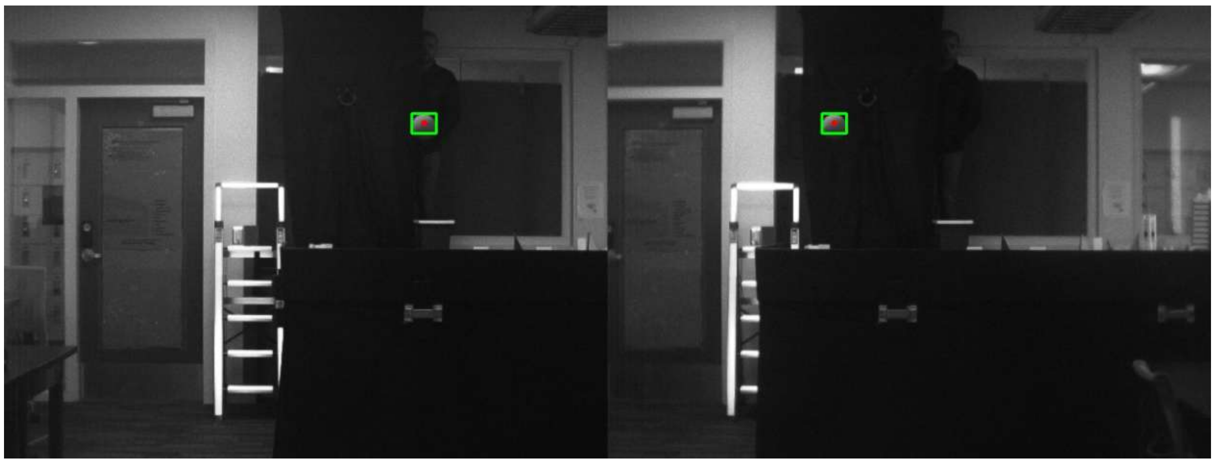


Here are some of the various steps we took in the project:

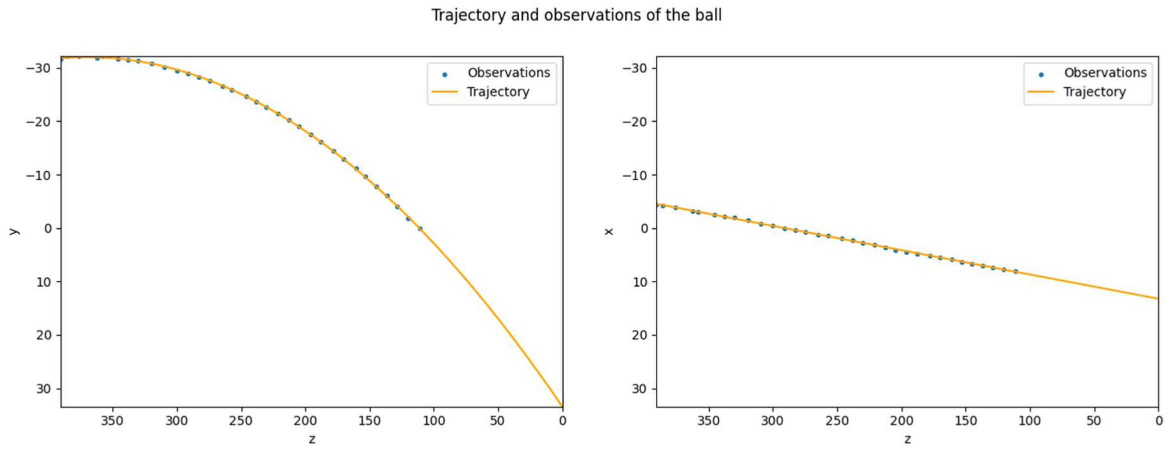
1. **Calibrate the stereo system**: For each camera, a series of images were taken of a large chessboard. Then OpenCV corner detection and calibration algorithms were used to find the fundamental matrix, essential matrix, rotation matrix, and translation vectors between the cameras.
2. **Detect the baseball**: In our algorithm, for each camera, we determined the pixel values for the area around the black curtain. Then, using that as our field of interest, we thresholded the gray-scale images and repeatedly applied OpenCV's findContours() function to look for the baseball until it was found. After that, the previous image (before the ball was detected) was stored for reference.
3. **Track the baseball**: Once the ball was found, our next goal was to find the pixel values of its location at each frame. First, we looked at the previous location of the ball and took a small area of interest around it (to reduce computation time and improve accuracy). Then we took the absolute difference between the region of interest of the previous image (without the ball) and the current frame.

   After thresholding the difference, we used OpenCV's findContours() function to determine the location of the baseball. Specifically, the center of the contour was stored as the pixel-valued location. To further improve the accuracy of this method, we rejected all contours which were too far from the previous location of the ball. Furthermore, if more than one contour was detected, we selected the one with the largest area.

The following images show some of the results from tracking the baseball for both cameras at each frame.

4. **Convert the pixel values of the baseball to 3-D coordinates**: Using the pixel coordinates of the ball in each frame, and the parameters found from the stereo calibration, we used OpenCV's undistortPoints() and triangulatePoints() functions to map the points from the pixel frame to the world frame. For each frame, this gave us the x, y, and z coordinates of the baseball with respect to the center of the two flea cameras.
5. **Estimate the future trajectory of the projectile**: Using the 3D coordinates of each frame, we ran a simple 2-degree polynomial interpolation on the y-coordinates and a 1-degree polynomial interpolation on the x points (see below). This gave us a function to approximate the x and y coordinates of the ball at each time step. We used this to estimate the location of the ball when z was 26 inches in front of the camera (which is the location of the catcher)



6. **Move the catcher**: After getting an estimate for where the ball will land in relation to the camera, we offset these values by the translation vector from the center of the cameras to the (0,0) coordinate location in the catcher. This translation vector was initially estimated using a measuring tape and later fine-tuned during testing. At every 5 frames, we ran the previous step to get an estimate for the final location of the ball and adjusted the location of the catcher to match it.

After quite a bit of fine-tuning and testing, we finally managed to get the catcher to consistently catch the baseball. An unedited video of 20 consecutive attempts can be found at https://youtu.be/KOQP9qNOZIU. An edited video of the same attempts is at https://youtu.be/2LnrD1ZdF_U. In the 20 attempts, we had 17 catches, 3 rims, and 0 misses.