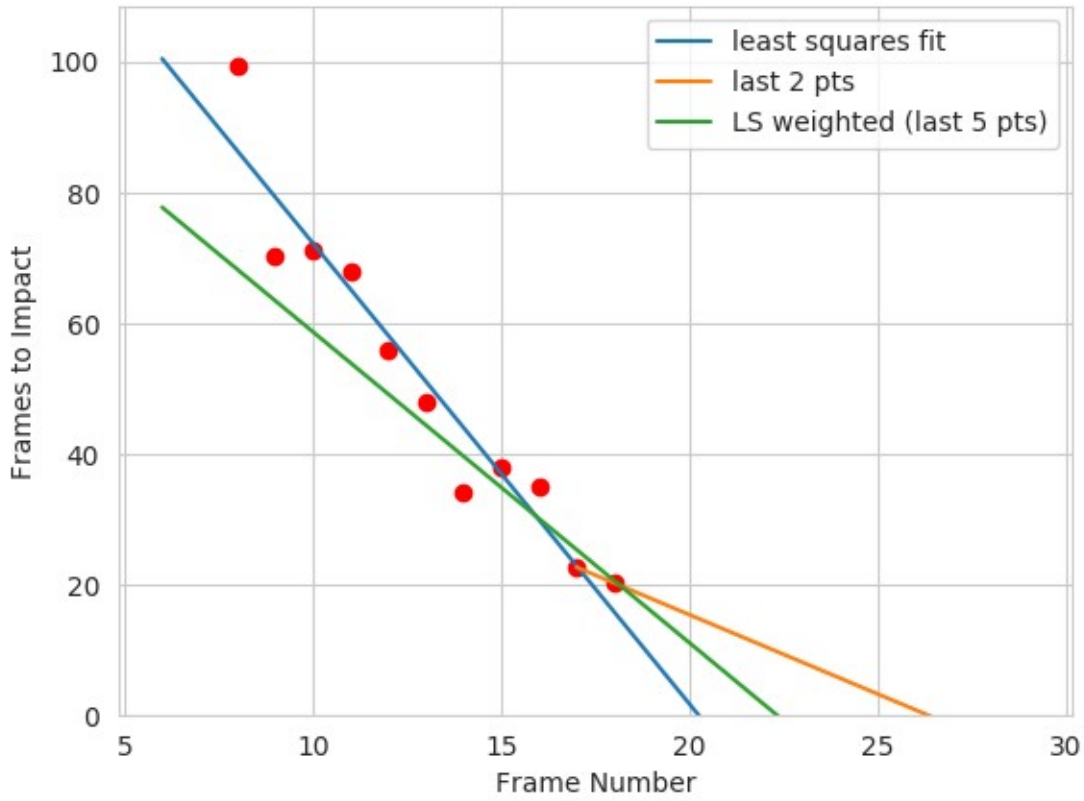
ECEn 631 3D Time to Impact – HW 7
Seth Nielsen

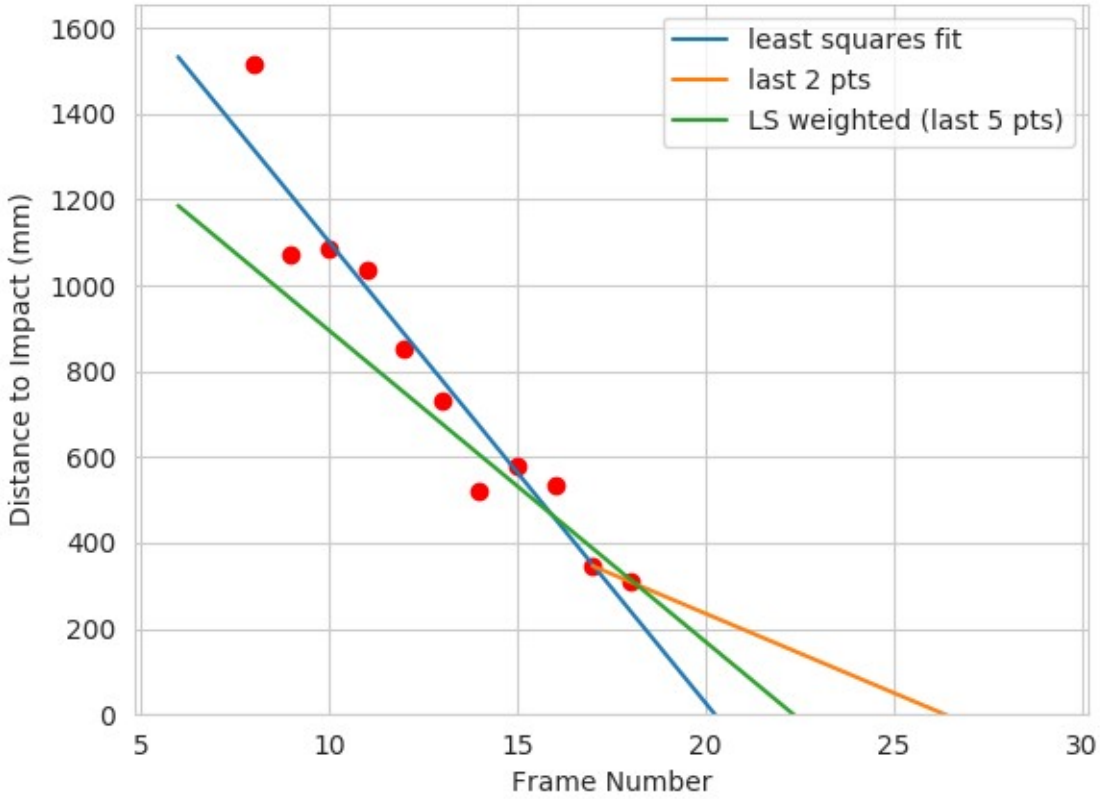
Task 1



To estimate the time of impact, I took two pairs of features, each pair consisting of features on opposite sides of the y-axis of the optical center. I compared the x-distance between the features of each pair (x) with their x-distance from the previous image (x'). I then used the equation $t = a / (a - 1)$, with $a = x / x'$. I then averaged the result for both pairs and created a vector of estimates for each image measured in "Frames to Impact".

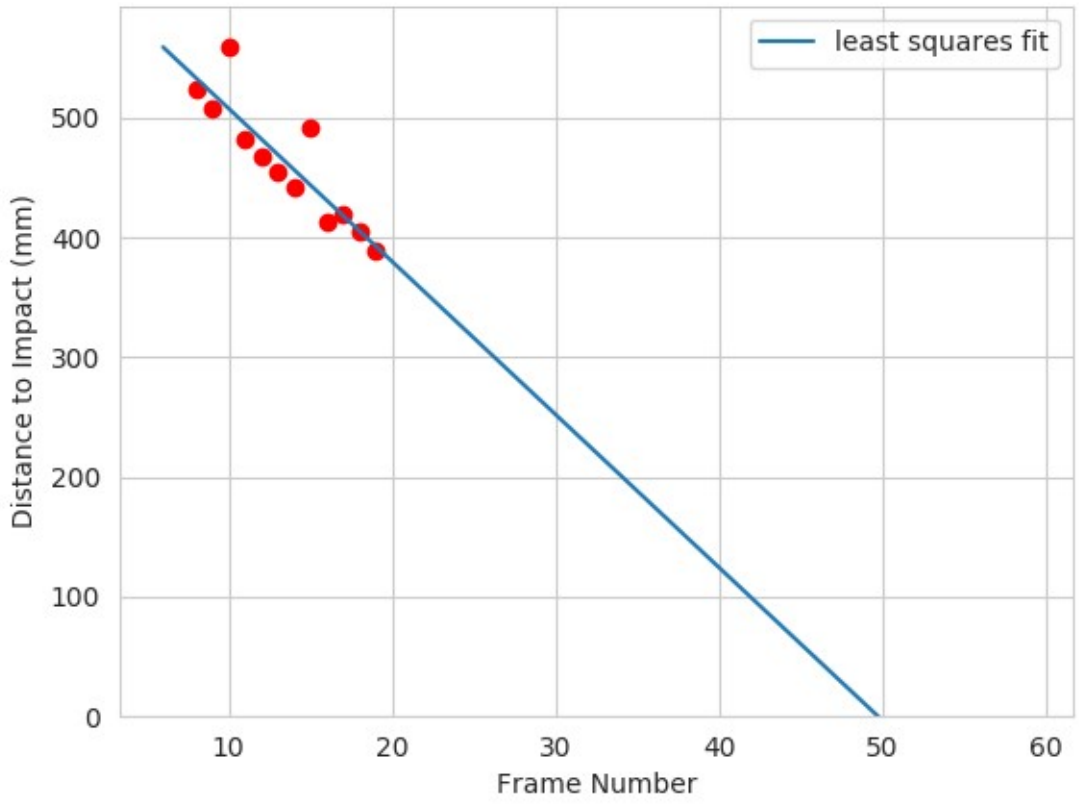
I then performed a least-squares approximation to fit a line to the data. This seemed to give a frame number too low, so I tried also extrapolating the slope of the last two data points to see where that reached. I then decided on using a weighted least squares approximation, where the last five data points had a weight 10 times higher than the others, thus giving a nice value right in the middle of the two extreme predictions.

Task 2



The process for this task was the same as for Task 1. The “Frames to Impact” values were simply multiplied by the distance traveled between each frame (15.25 mm) to get the estimated “Distance to Impact” measured in mm.

Task 3



To create this graph, I cropped the top and bottom of the image to create a region of interest that would only contain spray can pixels or background (the wall). Thus, using the OpenCV Canny function, I was able to accurately find the edges of the spray can with very little noise. I saved the values for the max x-position and the min-y position to get the size of the spray can in pixels, and then computed the distance to the can with the equation $Z = fx * X / x$, where $X = 59\ mm$ (the real width of the can) and $x = x\_max - x\_min$ (from the edges found by the Canny function). Saving a vector of data points and performing a least squares fit produced the plot seen above.

The data had much less noise in this task, and the estimated distance to impact seems more correct than the previous task. I believe this is due to the Canny edge detection being more reliable than using good features with template matching, which can be quite noisy and often loses features from frame to frame.