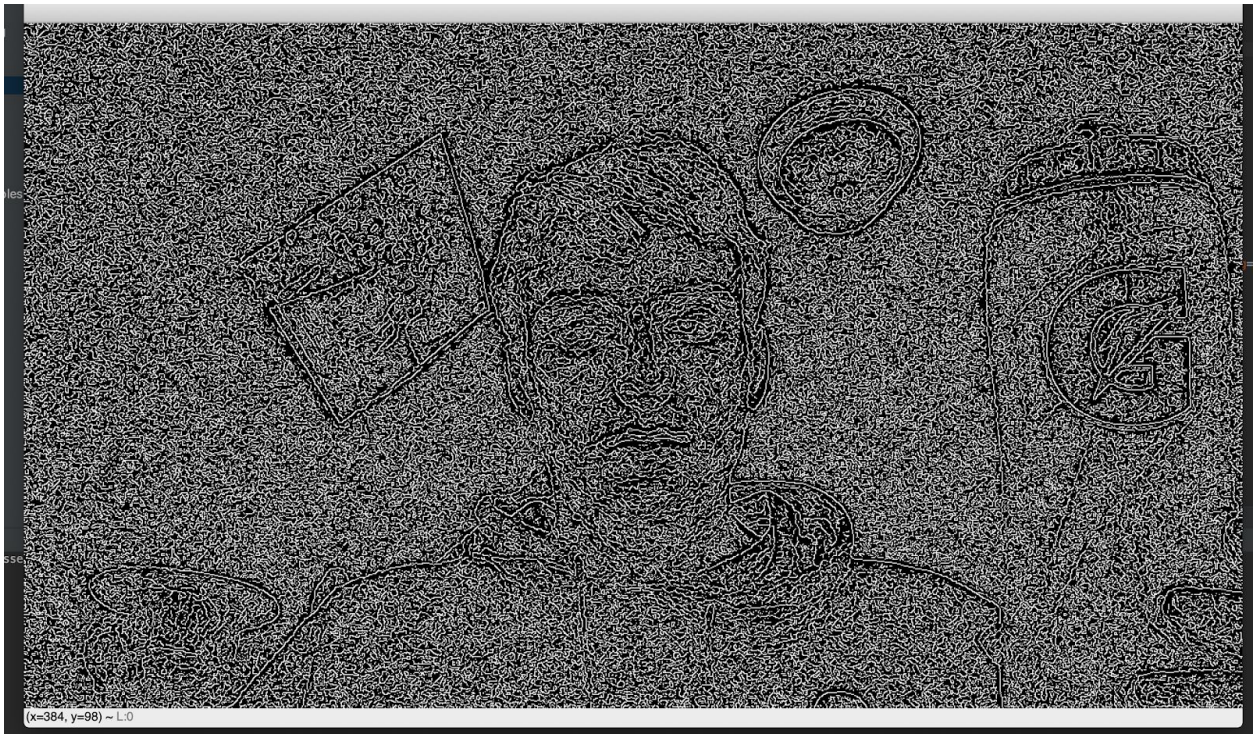
Lab 2 Jack Weissenberger

Experimenting with edge detection:
Canny parameters:
(0,0)



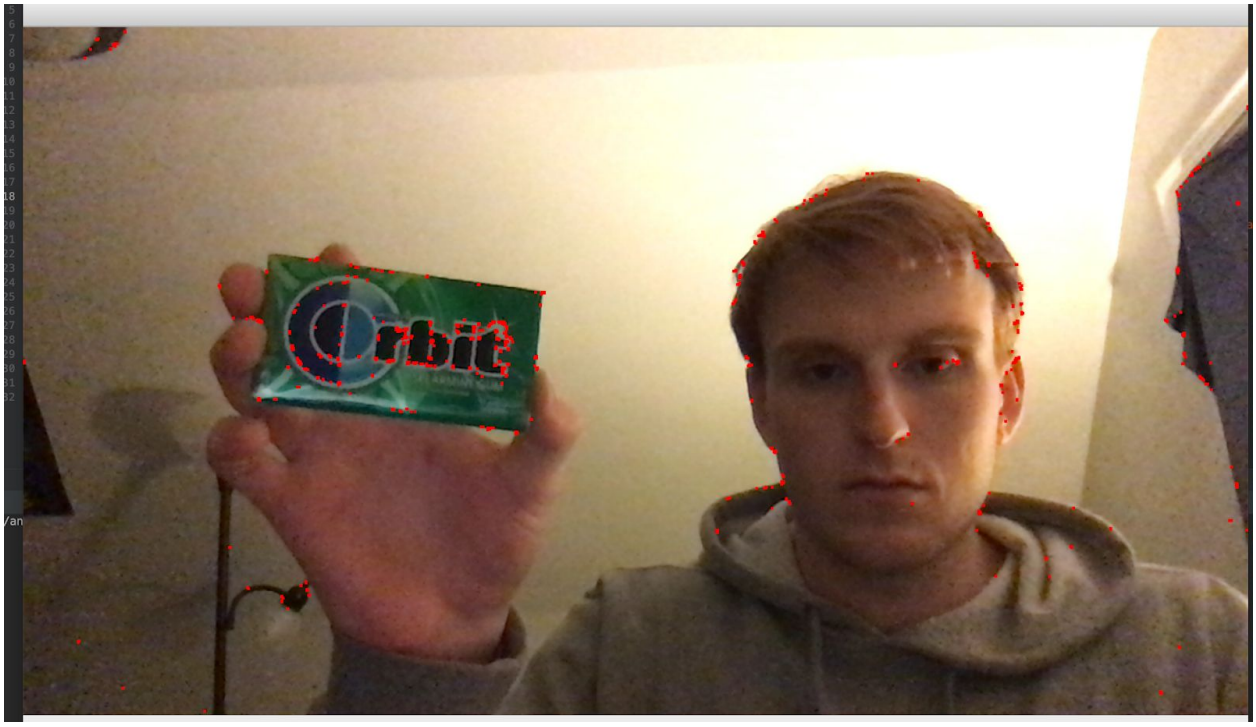
(1000,1000) Was a black screen
(100, 200)

Clearly as the parameters increase the lower the higher for the edges are until they are not visible and as the parameters get smaller the tolerance for an edge gets much lower and eventually every little change. Personally I found the best combination to be about 100, 200 to get the best results

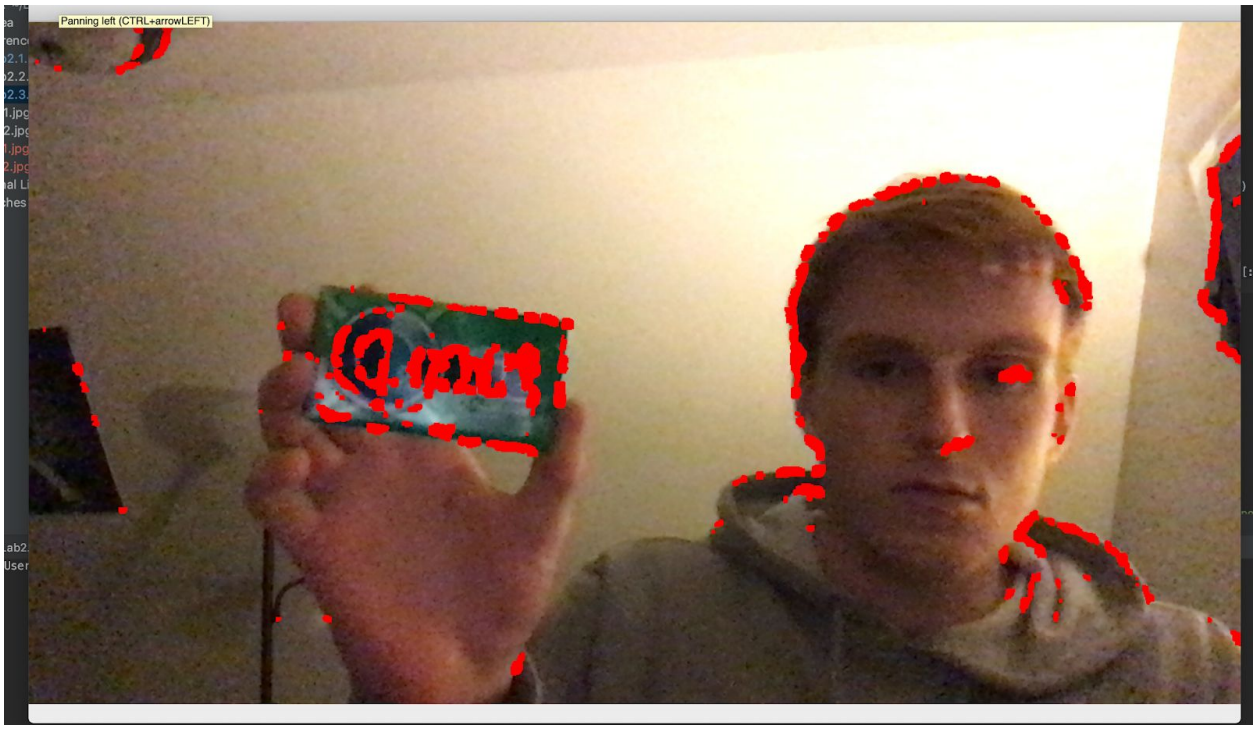
Experimenting with corner detection

From the documentation, we can see that the parameters we are tuning in the cornerHarris function are: 1. The block size/ neighborhood size, 2. Ksize which is the aperture parameter for the sobel operator and 3. Is the Harris detector free parameter.

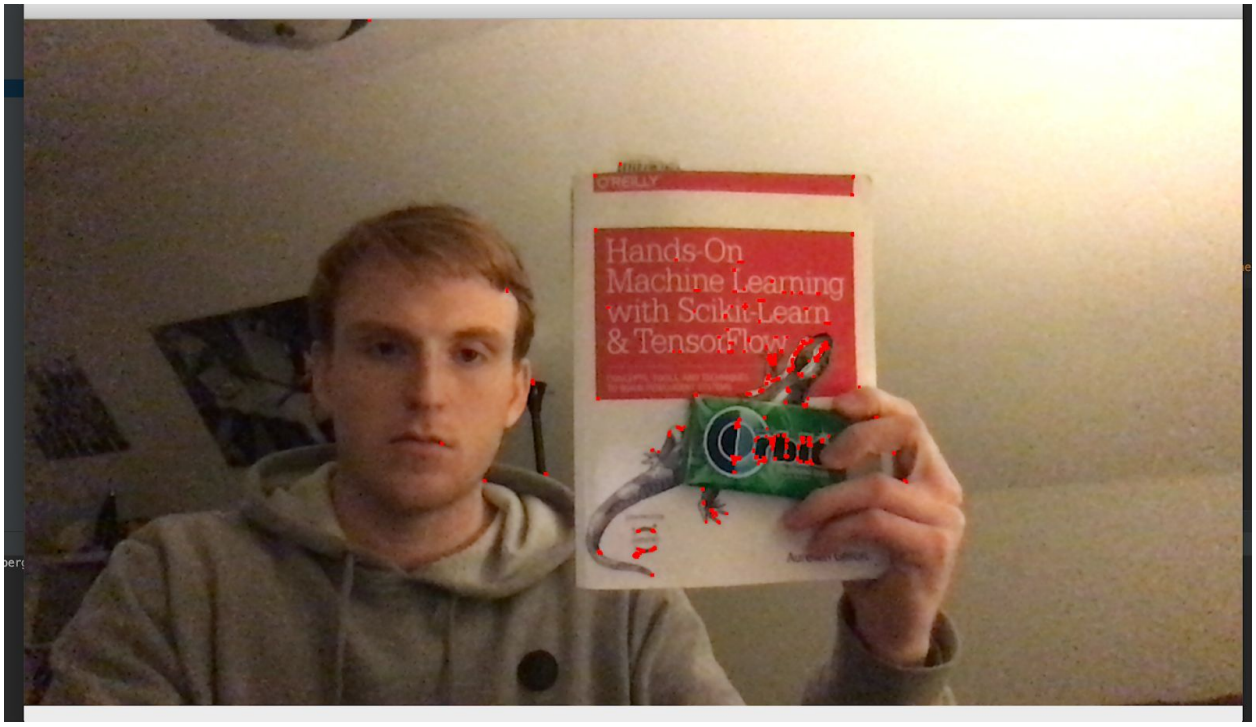
When the first parameter is changed the lower it is the more sparse the measured corner points are. It almost seems like the diameter of the circles themselves are increasing. This is visible in the next two pictures
First parameter is 2:



First parameter is 10:



The second also dealt with how many points were visible. When I raised the next parameter from the previous two pictures. The tolerance for what determined a corner seemed to be raised as only sharper corners were visible.



Increasing the last parameter had a similar effect. The corner measurements we're more sparse and occurred less often.

2 Sift Descriptors

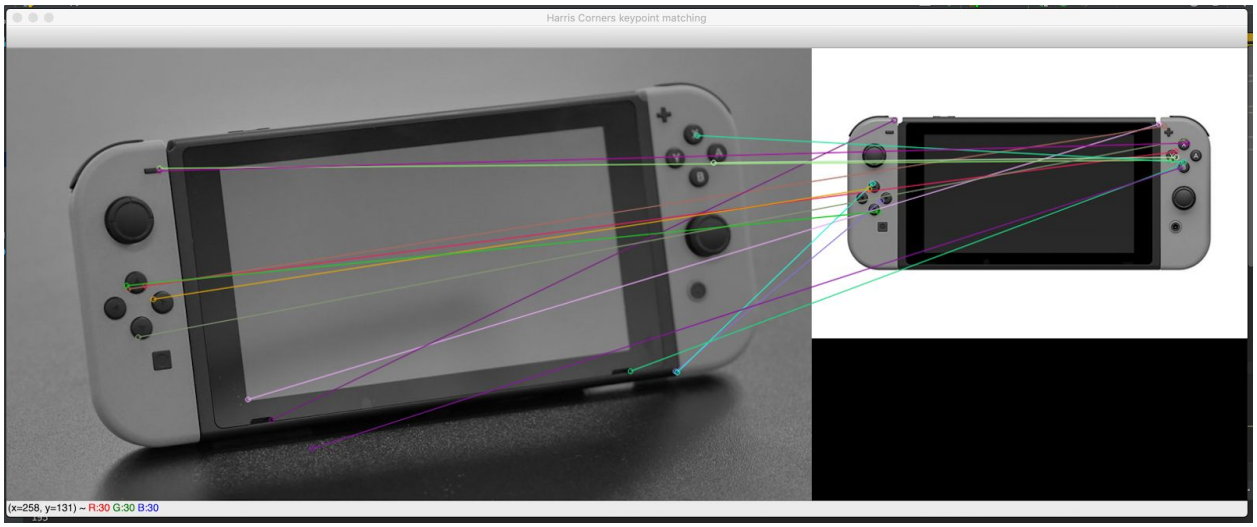
I used the ORB_create function to program Sift on an image. The first parameter I varied was the number of features. The first image has 100 and the second has 10,000, clearly the second was able to find more.

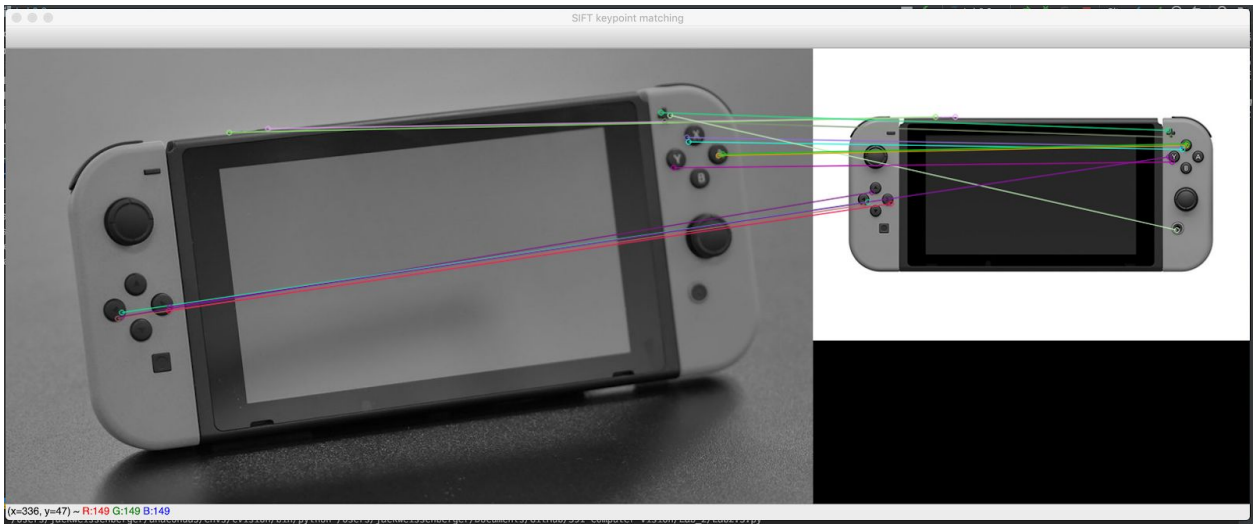
Next I increased the scale which also found a lot more features which you can see compared to the image above.

So clearly allowing the scale to change and adding more features to look for allows the image to find more points within each image.

3 Keypoints and matching

Here we have the keypoint matching using first Harris corner and then SIFT





As you can see these are two images of a gaming system taken at two different angles. Clearly SIFT does a far better job at matching up points between the two images. Neither is 100% correct at matching the points between the two images but we can see that SIFT seems to do a much better job.