Week 8 Deliverables for Milestone D1:

* **Code in github:** Done
* **Screenshots:** Done
* **Description of Process:** This section is code designed to do three things: Identify target colour pixels from an image, use percentiles and standard distribution to reduce the pixels to a useful range, and find the central point of all the pixels. This is so that this program can tell the navigation program where the target is. This was developed by pulling together code information from many different sources, piecing them together over the course of four weeks, and iterating on them significantly. One thing did not work was using the statistics library for python because it would just crash the program. I overcame that by manually recreating the maths for percentiles of confidence intervals and writing it into the code. Another thing that didn’t work was trying to identify colours that were similar to those of the test environment. I wanted initially to make a program that could work for detecting colour based targets that may have been in the turtle’s natural environment, but I ended up not being able to tighten the parameters enough without it losing its utility, so I instead decided to focus on the red and purple colour ranges and make the target that colour, which worked.
* **Description of operation:** There are four parts to the code. Part 1 is basically comparing the image to the desired colour range and producing and image that just highlights those pixels. This is done by using the open cv library and doing transformations on images to create the image. Part 2 scans over that result image, in which every pixel except the detected ones in range are black, and then creates a touple list of coordinates that correspond to every pixel that isn’t black, thus saving the pixel location of every desired pixel. Part 3 is trimming the pixels down to exclude far outliers that aren’t actually the target we’re looking for. It does this by creating a list of the x and y variables and then running each of them through an algorithm that excludes any that fall outside of the 5% and 95% confidence interval of a regular distribution. This is then recombined to be a new list of filtered variables. This new list is then used to find the maximum and minimum of the X and Y values and then use that to determine not only the corners of the field of view but also the central pixel value. This central value will then be passed back to the navigation stack that called for it. Part 4 is just creating a black and white image based on the filtered pixels detected in order to show the developer if they’re tweaking the program in the right way.