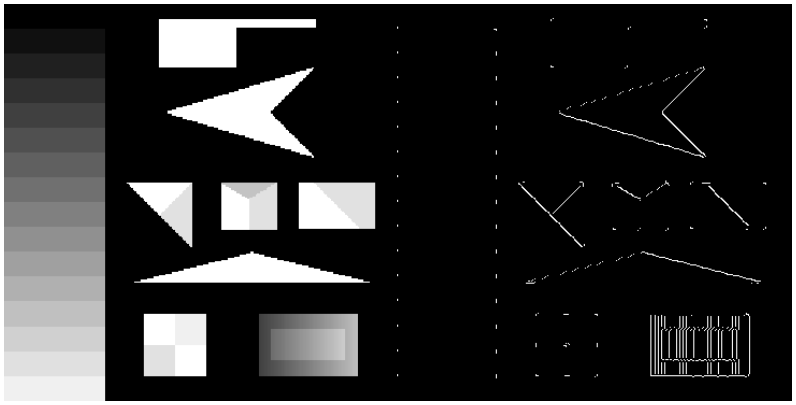
Jack Hanke
MSAI 495 Computer Vision
MP #5

**Canny Edge Detector Report**

In this assignment we implement Canny edge detection. This algorithm involves a series of preprocessing steps before calculating the final edges. We first slightly blur the image to remove sharp boundaries in intensity not associated with a true edge. We blur using a 3x3 gaussian filter. Next we calculate the partial derivatives of the image in x and y to compute the magnitude and direction of the largest increase. We use this magnitude to identify the location at which the image is changing the fastest. This is presumably an edge, but it could also be a textured surface with some changes that should not be considered an edge. To address this, we only mark pixel locations that are on a “shelf”. We do this using LUT, where we check if the magnitudes of the neighboring pixels are smaller in the perpendicular direction of the main pixel’s gradient. This check is coded by manually filtering between various theta ranges and the associated magnitudes of the specific neighbor pixels. This finally results in an image that has the true maxima marked. To complete the process, we next create a histogram of magnitudes (bucketed by integer). The magnitude that is larger than 80% of the other magnitudes is considered the high threshold, and 50% of that magnitude is considered the low threshold. These thresholds, called high and low, can then be used to create two images: Images that are white only where the magnitudes on the non maxima suppressed image are greater than high and low respectively. These two images form the “soft” and “hard” edges used in the final edge linking step. Image linking begins with the hard edges. Hard edges are added to the final image, as well as soft edges adjacent to a hard edge. This results in more complete edges while being robust to noise.

For my implementation, my decisions for individual preprocessing were made around how good the intermediate *lena* image was to the slides from class. This resulted in an edge labelling of Lena comparable with the final one shown in the slides.

All intermediate images are included in the project files for all test images. Though I'm happy with the lena result, the test image did trick my implementation.



From what I determined, it appears that the code for non maxima suppression seems to not handle this image well. Considering the other test images resulted in reasonable edge labeling, I'm confused as to why this labeling is so poor.