**Least Significant Bit Steganography**

For the final Signals and Systems class, I took, EE 424 Intro to Digital Signals Processing, we were assigned to choose an open-ended final project, which we had to present on. Me and a partner chose the topic of steganography, which involves hiding a piece of data within something else. Specifically, our goal was to utilize the Least Significant Bit technique to hide different forms of media within 2D images using MATLAB code. The Least Significant Bit technique will clear N least significant bits from each 8-bit Red, Green, and Blue value for every pixel and set them to the target media to hide within the image. This allows us to slightly alter the original image, with an undetectable amount, while hiding pieces of media such as text files, other 2D images, or even full feature movies.

The overall code process was very similar for hiding different types of media within 2D images. We first began by writing text files, by reading out every char and breaking it down bit by bit. We utilized a 3 level nested for loop to clear and set N bits, to be set with the 1D array of bits from the text file. We found that with an N value of 3 or more, the original image to hide data within started to become noticeably corrupted. Despite this, it still was possible to fully recover the hidden media. A similar process was followed for hiding 2D images, which just involved updating how the images were formatted into a 1D array of bits from each pixel’s colors.

To further push our code, we explored writing data out as a bitstream, which allowed for higher amounts of compression due to how Windows compressed 2D images with lossless data. This allowed us to hide a direct copy of an image within itself, since we were reading the original image out uncompressed in MATLAB to hide it within. We also created a script to upscale the resolution of an image, which allowed us to fully hide the feature-length film of The Bee Movie within a picture of our friend Caden. This comparison can be seen below, with the full embedded movie hidden in the image to the right.

To analyze the results of hiding media, we used the Structural Similarity Index, which compares the likeness of two images, in which we could compare the original image and the modified image with hidden contents, to see how similar they appeared to the eye. The Structural Similarity Index ranged from 0 to 1, with 1 being perfectly identical. We found that with the bitstream implementation we could not hide compressed text files, but could achieve a lower N value and higher similarity with 2D compressed image, due to lossless data loss with Windows image compression. This provided for some nice comparisons in both our report and presentation, can be seen below.