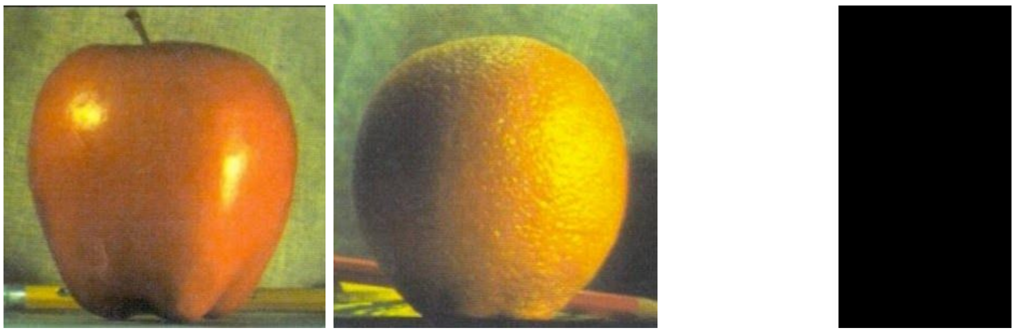
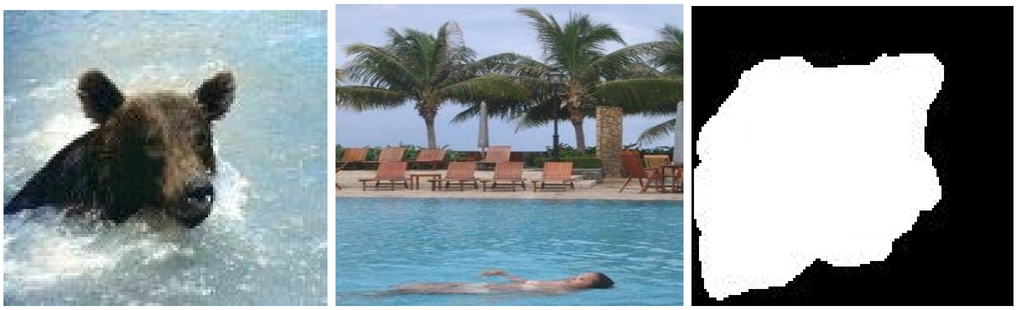
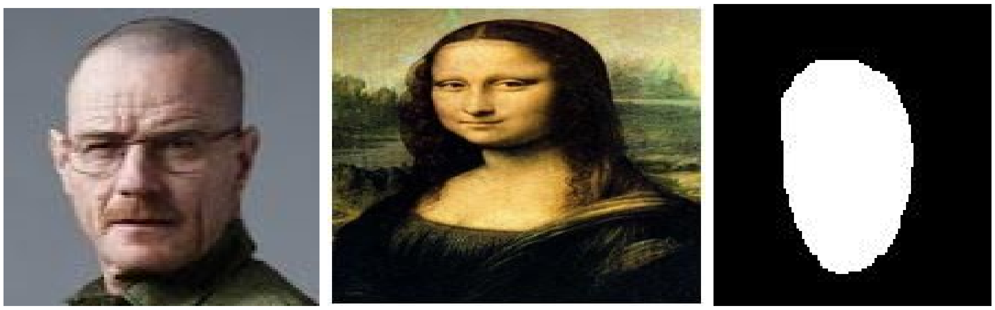
CSCE 489/689 -Computational Photography

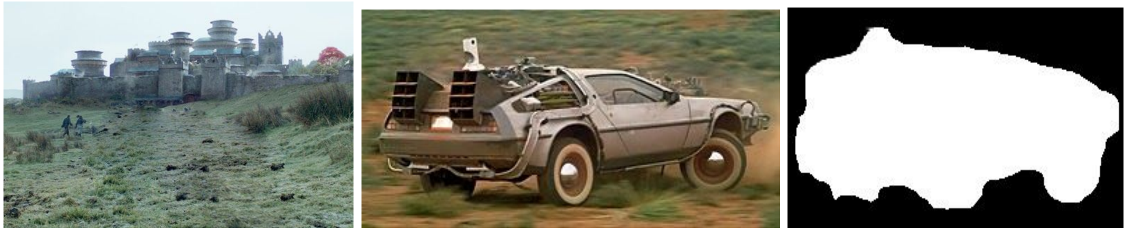
Programming Assignment 3
Deadline: March 2, 2020
Report by:
Jorge Farinacci

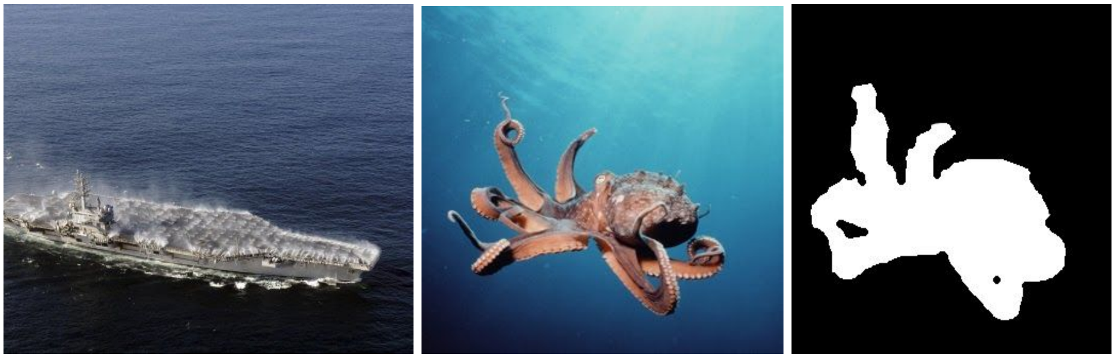
Input image(s):

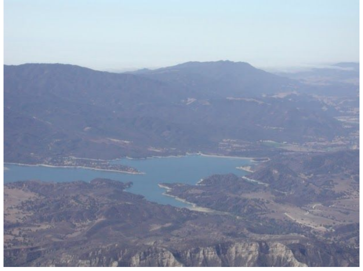


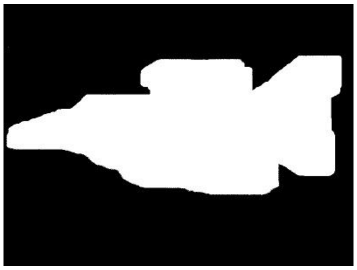


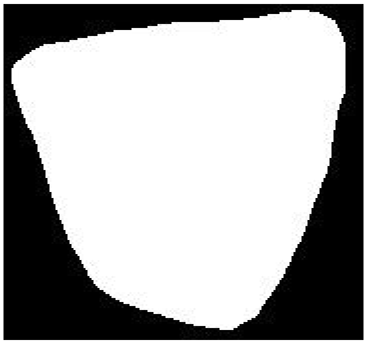


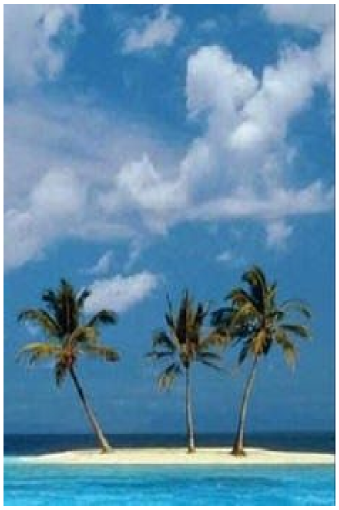


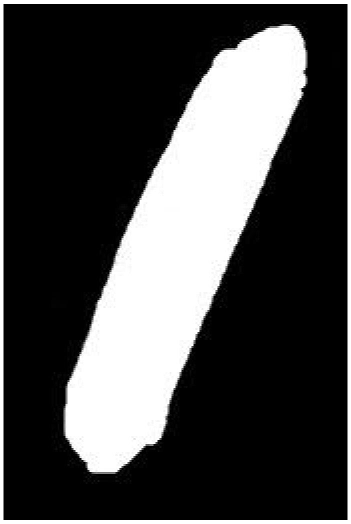




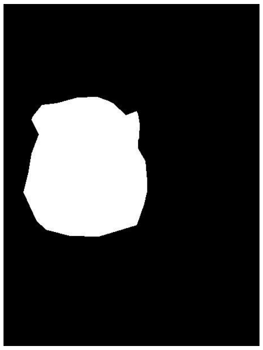




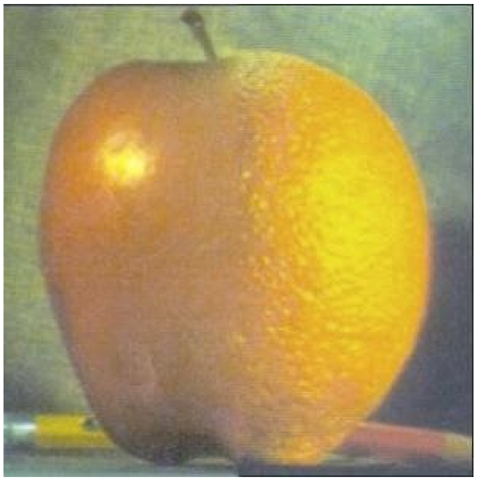
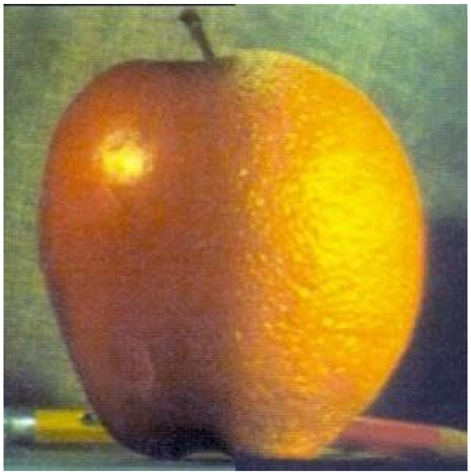
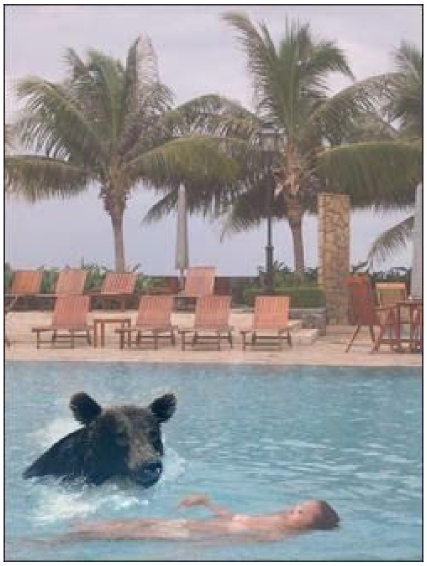
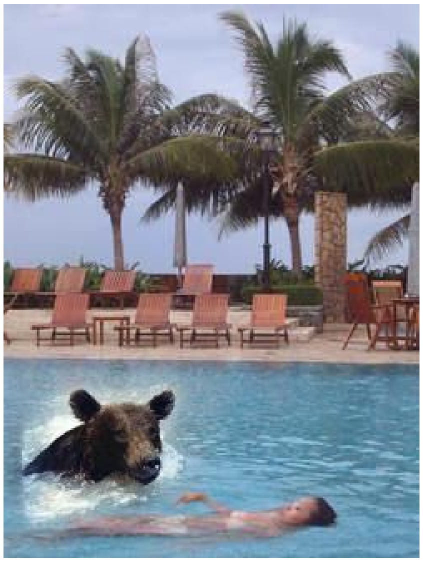
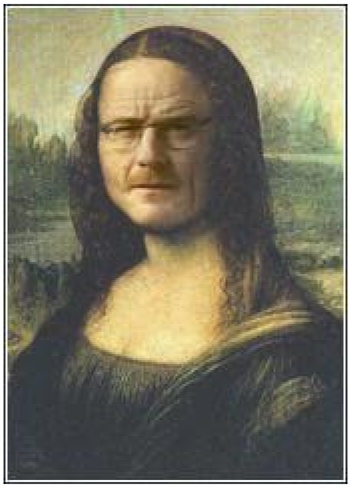
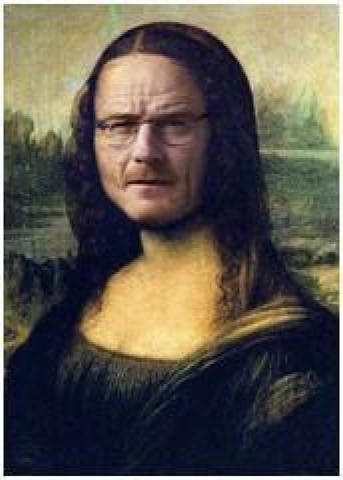


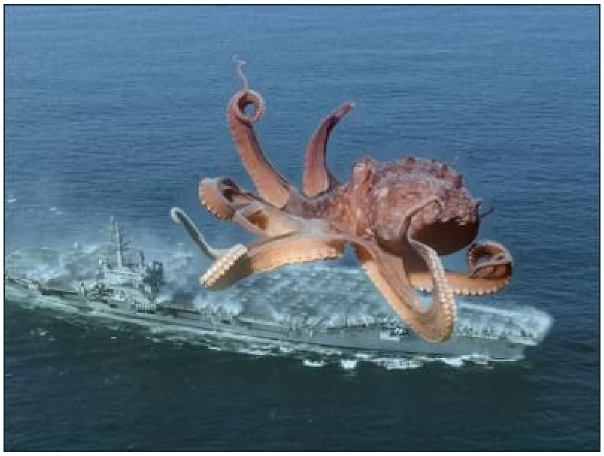


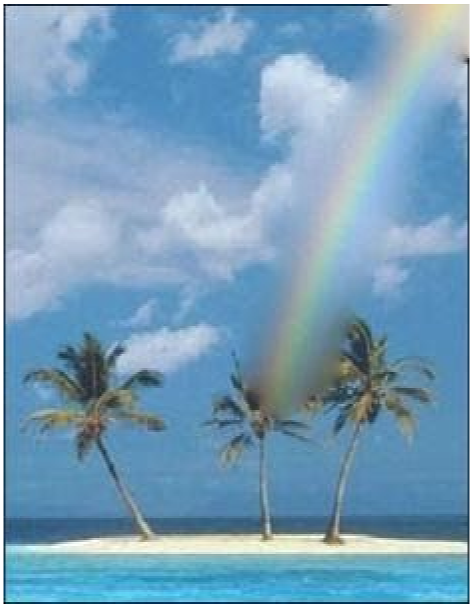
My own test image



Left hand side is poisson blended and the right hand side implements the laplacian-gaussian pyramid.

	
SP	SP

∇f	∇f
	

Analysis

I’ve implemented both phases of the assignment in two separate files, pyramid.py and poision.py. For phase 1 (pyramid.py) we needed to develop 2 laplacian pyramids for source and target images and a gaussian pyramid for the mask. The laplacian pyramid was built via a difference of gaussian methods, such that it required a gaussian pyramid as input to make the pyramid. Initially I tried to do this by using a laplacian of gaussian function, but that didn't work, ended as it ended with grayscale images, despite the fact that i kept a colored subsampled image to scale back up with.

Poisson.py was substantially different, the method required a large sparse matrix to compute the laplacian and blend the images. Inherently this was slow and resulted in millions of needless calculations.As per instruction i used scipy sparse methods to reduce and vectorize matrix and convolve it with the source images, and in the target i zeroed out the region where the the source would be implanted and i used a matrix solver equation to compute values at the blanked out points. Finally I normalized the image to range 0 and 1.