Kingdom_Species.ipynb

This is a pixel classifier that I wrote to find pixel "species" within an image. I found the biological terminology useful when thinking about this problem. Each pixel is cross referenced against existing species defined by RGB pixel ranges. If the species already exists that population is increased by one and the ranges (defined by +/- n*std) are recalculated. If not, a new species is created. In the end, the output is a list of the average pixel types. This was useful in a segmentation problem which had objects of very distinct color.

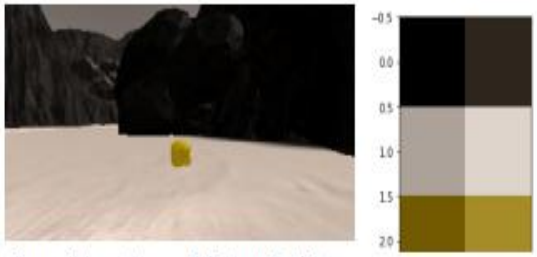

Figure 3: Input Image & Output Pixel Ranges

Model_training.ipynb

This notebook was part of a template from an image segmentation project I worked on through a robotics course on Udacity.com. The goal was to implement a SegNet (as described here https://arxiv.org/pdf/1511.00561.pdf) consisting of a set of decoders, followed by a 1x1 convolutional net, and finished with an equal number of encoder. An Adam optimizer was used with a categorical cross entropy loss function. Also, a dropout with rate of 0.25 was used to prevent overfitting. I deleted some cells which contained code that I did not write, edit, or interact with extensively.

Perception.py

This code contains a suite of functions that were used for a robot mapping project where a robot too in an image, applied basic RGB thresholding, and then computed a series of transformations to obtain an overhead map. The math that makes this possible is the perspective transform, which takes four points within an image as a reference and fills a perfect square with the contents of the polygon created by these points. The cv2 python module was used for these transformations.

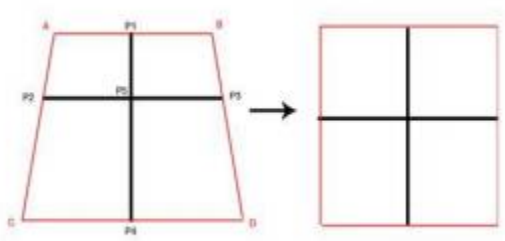
Figure 4 Perspective Transform Visualization