CSI 758 Spring 2017 MID TERM

Bruce Goldfeder

January 4, 2018

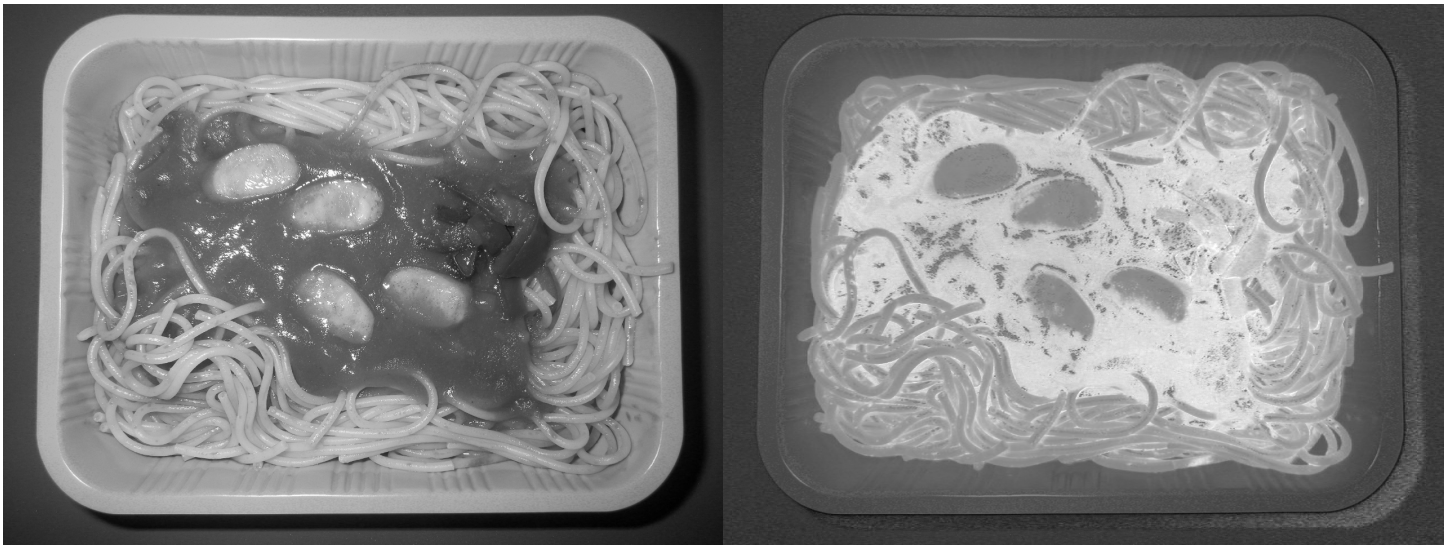
For all of the questions in this exam we will be using the following two input image files. The first being the quite delicious looking meal with a bit of greenery on right. The other is a truth image to be used throughout as a mask and also as a check of our algorithms. These images are shown below:

(a) Yummy Food (b) Truth

Figure 1: Input Images

1 Question 1

The R, G and B rectilinear coordinates are converted using the function **ToSpherical(filename)**. This function uses the given formulas to convert the coordinates returning three images - ρ, θ, and ϕ. The justification is that the ρ component is the intensity and the two angle components are dedicated to the hue. The images are shown below:



(a) rho (b) theta

(c) phi

Figure 2: Spherical Coordinate Images

This function can be tested using the following code after importing in the python file included called "midterm.py".

```
rho, theta, phi = ToSpherical ( "mydir/1.JPG" )
sm.imshow( rho )
sm.imshow( theta )
sm.imshow( phi)
```

2 Question 2

To test this conversion I employed the reversed transformation and get an image that is very close to the original. The function **ToRect(rho,theta,phi)** returns the image file which closely matches the original input image. This image is shown below: You can run my program by:

(a) Converted Image

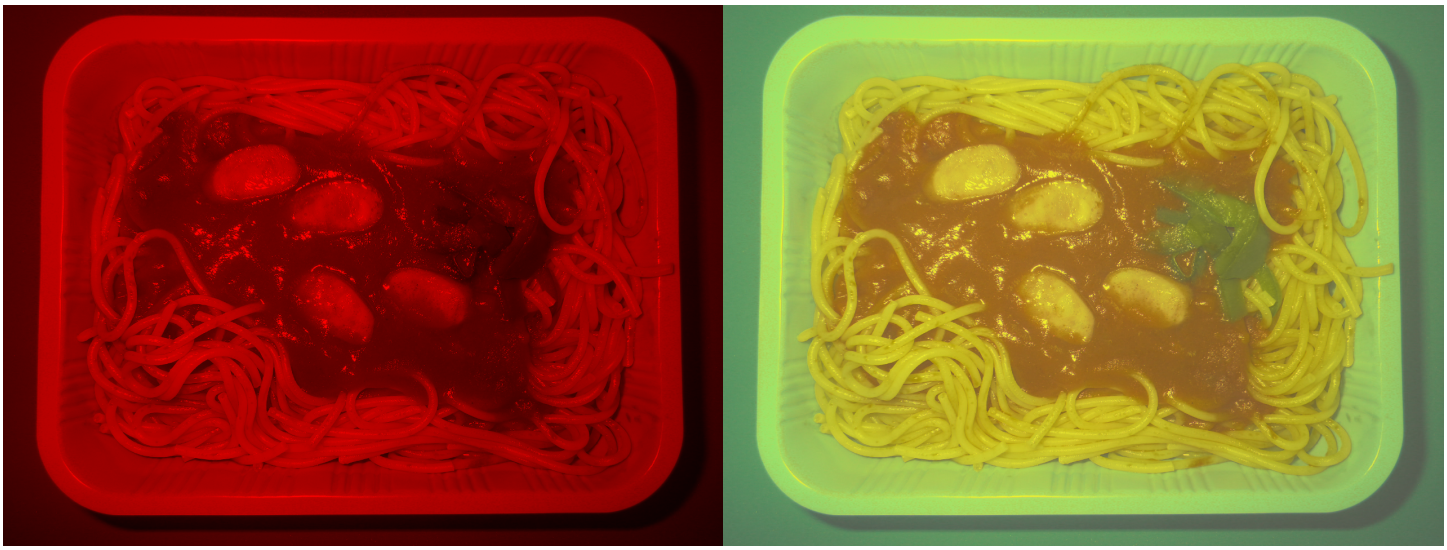
Figure 3: Spherical Coords Converted back to Rectilinear

```
mg = ToRect(rho, theta, phi )
sm.imshow(mg)
```

3 Question 3

The function **RotateGreen** which receives as an input the file name of the food image and returns a matrix with the theta shifted by a value of $\pi/4$ which is b$[\vec{x}]$. The images below represent that image in spherical coordinates and then again after it is converted back to rectilinear coords. The conversion to rectilinear is accomplished using the following code:

```
mg = RotateGreen( "mydir/1.JPG" )
sm.imshow(mg)
mg2 = ToRect(mg[:,:,0],mg[:,:,1],mg[:,:,2])
sm.imshow(mg2)
```



(a) Matrix in Spherical Coords (b) Converted back to Rectilinear Coords and RGB

Figure 4: Green Shifted image in Spherical Coords and in Rectilinear Coords

You can run my program by:

```
mg = RotateGreen( "mydir/1.JPG" )
sm.imshow(mg)
```

4 Question 4

The **GetStats** function returns four floats representing the mean and the standard deviation for the theta pixels and phi pixels. This returned:

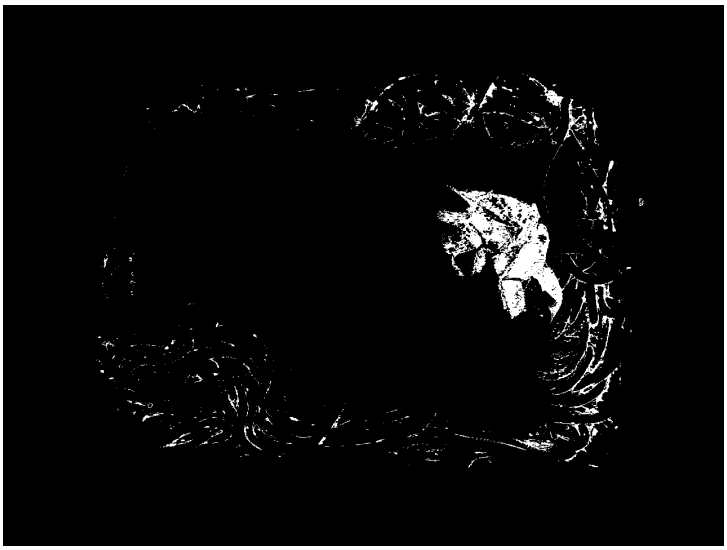
```
>>> m1,s1,m2,s2 = GetStats("truth.png","1.jpg")
>>> m1,s1,m2,s2
(1.3054029473241031, 0.09934194603367931, 0.78085135352757407, 0.10681206316251182)
```

The call should be:

```
m1, s1, m2, s2 = GetStats( "mydir/truth.png", "mydir/1.JPG")
```

5 Question 5

Using the function named **Threshold** I applied both thresholds. I used the outputs from **GetStats**, used the value of one (1) for α and β. The resulting image is a matrix of binary-valued pixels (either 1 or 0)and is below:



(a) Binary-Valued Output

Figure 5: Threshold output using alpha and beta of 1

```
answ = Threshold( m1, s1, m2, s2, alpha, beta, "mydir/1.JPG" )
sm.imshow( answ )
```

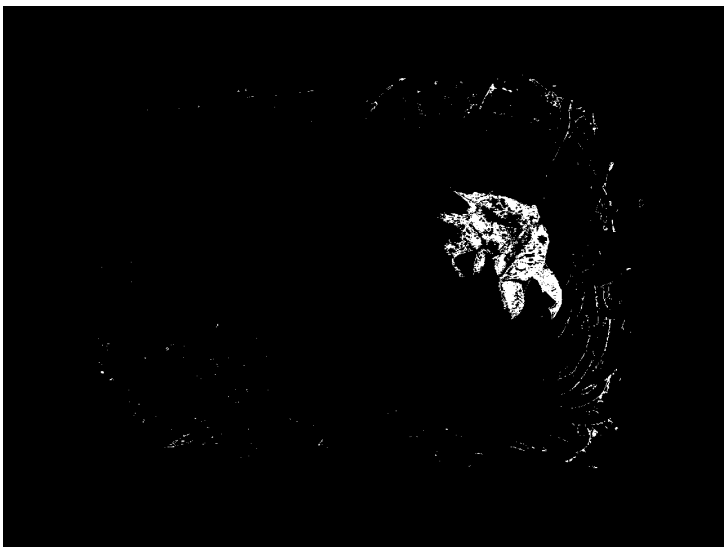
6 Question 6

The function named **SimAnn** is a simulated annealing program which finds the optimum values of α and β. The inputs are the name of the truth file and the name of the food file. The outputs are the determined values ofα, β and an image that shows the pixels that are classified as on-target when using these values.

I used a random from -1 to 1 for both alpha and beta independently. As the Threshold function failed on negative numbers I put in a floor and ceiling of 0.1 and 3 respectively if the new guesses were outside of those ranges. I chose a default scaling down of 0.99 for the sake of time. Most successful runs had a resulting cost of less than 19,000. The results for alpha and beta are:

```
>>> alpha,beta
(0.8404207530543323, 0.7200160149024841)
```

The final image is shown below:



(a) Final Output

Figure 6: Output from the Simulated Annealing

The call is:

```
alpha, beta, mg = SimAnn( "mydir/truth.png", "mydir/1.JPG")
print( alpha, beta )
sm.imshow( mg )
```