Spoons

Michael Gallagher - 10325071

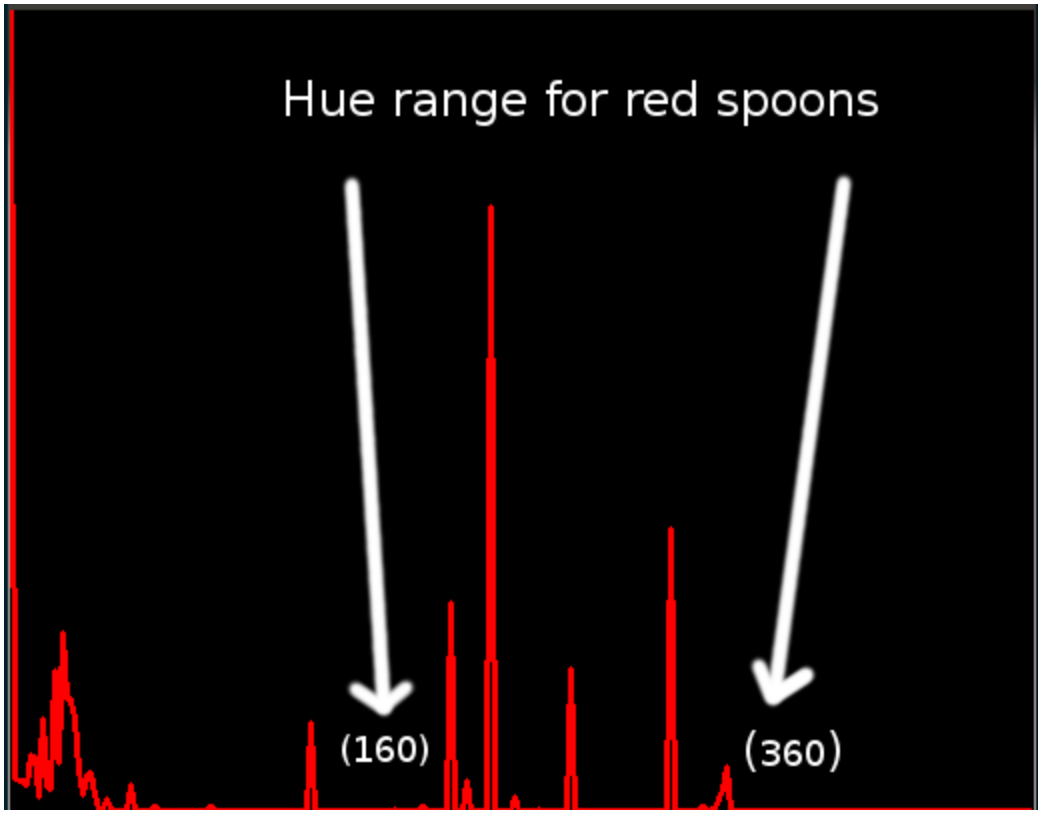
I feel like the spoons assignment was a good way to make us understand how computer vision might work in a practical way, as well as how opencv can make our lives a lot easier as it has so many methods built in.

For my lab I experimented with many methods which I will talk about later. For my actual submission I just relied on hue and saturation thresholding to isolate the red pixels of the spoon, and then count the pixels to determine if there was 0, 1 or 2 spoons. As well as counting the number of red pixels found, I also clone the original image and black out any pixels on the clone that are identified as not red. I then return and display this image. This made it much easier to identify if I had chosen the correct colour range.

```
Mat count_red_spoons(Mat image)
{
   Mat hls, histImage;
   cvtColor(image, hls, CV_BGR2HLS);
   Mat red_only = image.clone();

   int red_count = 0;
   for (int row = 0; row < hls.rows; row ++){
      for (int col = 0; col < hls.cols; col ++){
         int hue = hls.at<Vec3b>(row,col)[0];
         int saturation = hls.at<Vec3b>(row,col)[2];
         if ( ((hue  <= 2 && hue >= 0) || hue >= 160) && (saturation >= 120 )){
            red_count ++;
         } else {
            for (int channels = 0; channels < red_only.channels(); channels ++){
               red_only.at<Vec3b>(row,col)[channels] = 0;
            }
         }
      }
   }
   if (red_count < 500){
      printf("No spoons found!\tRed count is %d\n",red_count);
   } else if (red_count < 13000){
      printf("1 red spoon found!\tRed count is %d\n",red_count);
   } else {
      printf("2 red spoons found!\tRed count is %d\n",red_count);
   }
   return red_only;
}
```

I also used histograms to get an initial idea of what range the thresholding should cover. I compared the histograms of images with spoons and images without spoons and noticed a large jump in the histogram. The following image is a histogram of hue when 2 spoons were present in the image. The pixel density between 160 and 360 degrees spiked in images where spoons were present.



These 2 methods allowed me to correctly identify the amount of spoons in the images for all the sample images. However in the process of achieving this I tried other methods which I did not end up using in the final submission.

I used k-means to smooth the amount of colours in the image as the shine and shade on the spoons was initially causing my hue thresholding to fail to identify some parts of the spoon. The code for the k-means is as follows.

```
Mat get_k_means(string filename, int clusterCount, int iterations){
   Mat src = imread(filename, CV_LOAD_IMAGE_COLOR);
   Mat samples(src.rows * src.cols, 3, CV_32F);
   for( int y = 0; y < src.rows; y++ ){
      for( int x = 0; x < src.cols; x++ ){
         for( int z = 0; z < 3; z++){
            samples.at<float>(y + x*src.rows, z) = src.at<Vec3b>(y,x)[z];
         }
      }
```

```
    }

    Mat labels, centers;
    kmeans(samples, clusterCount, labels, TermCriteria(CV_TERMCRIT_ITER|CV_TERMCRIT_EPS,
10000, 0.0001), iterations, KMEANS_PP_CENTERS, centers );

    Mat new_image( src.size(), src.type() );
    for( int y = 0; y < src.rows; y++ ){
        for( int x = 0; x < src.cols; x++ ){
            int cluster_idx = labels.at<int>(y + x*src.rows,0);
            new_image.at<Vec3b>(y,x)[0] = centers.at<float>(cluster_idx, 0);
            new_image.at<Vec3b>(y,x)[1] = centers.at<float>(cluster_idx, 1);
            new_image.at<Vec3b>(y,x)[2] = centers.at<float>(cluster_idx, 2);
        }
    }
}
```

Using k-means resulting in these style images:

Finally, I looked into using erosion as a more redundant method of finding a spoon. The idea was that using erosion I should be left with just the head of the spoon, so it would be easier to identify how many spoons there were. I did not implement this method however as I already had the program identifying the spoons correctly.

Metrics

The program correctly identifies the number of spoons in each image. Therefore based off the sample I have been given the metrics are:
Recall = 1
Precision = 1
Accuracy = 1
Specificity = 1