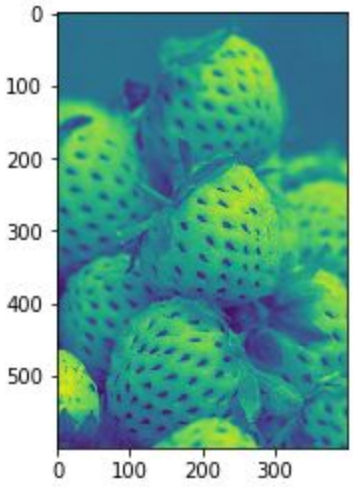
CSE 344: Computer Vision

Homework 5 ; Arka Sarkar 2018222

Part 1 : LBP : Local Binary Pattern

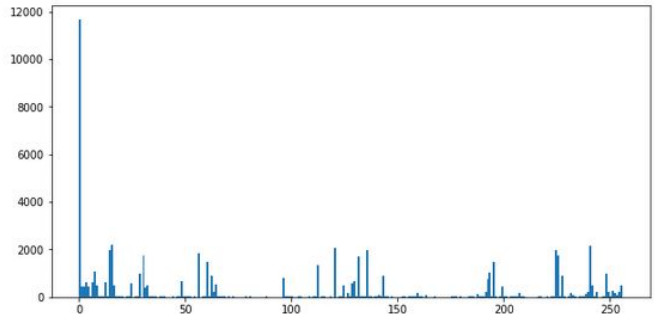
```
image = cv2.imread('straw.png')
image = cv2.cvtColor(image, cv2.CC
print(image.shape)
plt.imshow(image)
plt.show()
```

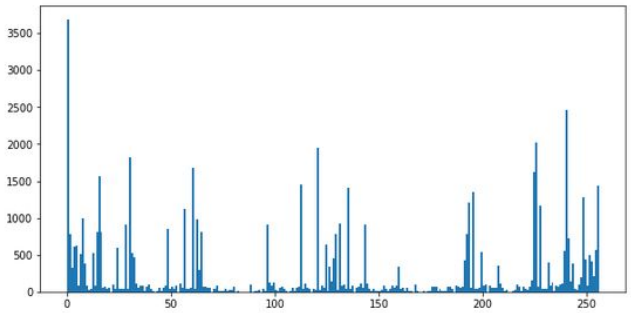
```
(600, 400)
```

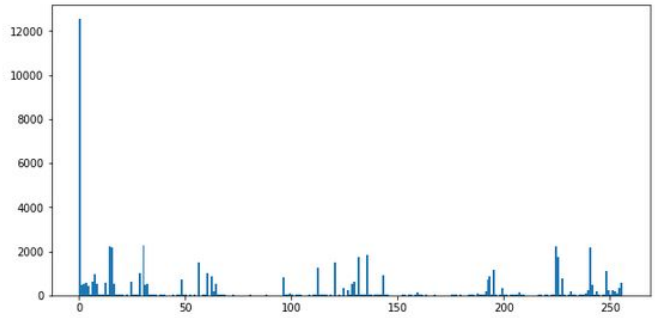


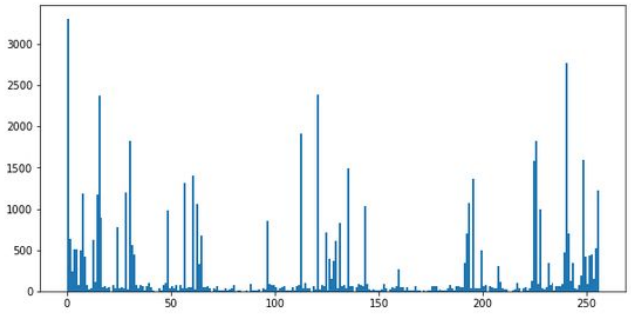
Input

Histograms generated for the 4 patches of the image:









Part 2 : <u>HOG : Histogram of Gradients</u>

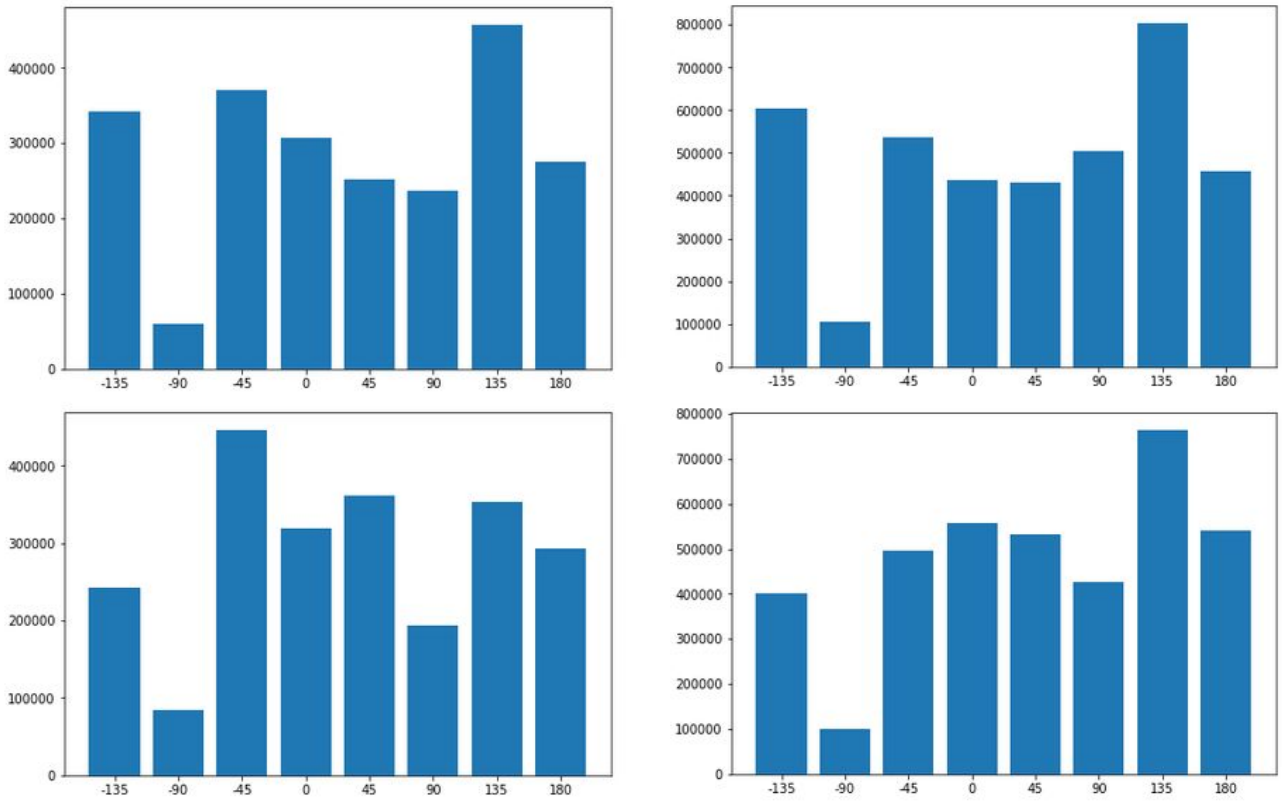
1. We compute the gradient's magnitude and direction.
2. Key angles are chosen (say 0,45,90,135,180,-135,-90,-45) to form bins for the histogram.
3. For a pixel in a patch, based on the proximity of the gradient's direction with the two key angles between which the direction lies, its magnitude is shared by the bins of the two angles.
4. When we share magnitudes of each pixel in the patch this way and accumulate these magnitudes for the bins, the result is called HOG feature.

Algorithm

```
def HOG_cell_histogram(cell_direction, cell_magnitude, hist_bins):
    HOG_cell_hist = np.zeros(shape=(hist_bins.size))
    cell_size_x, cell_size_y = cell_direction.shape

    bins_gaps = abs(hist_bins[0] - hist_bins[1])

    for row_idx in range(cell_size_x):
        for col_idx in range(cell_size_y):
            curr_direction = cell_direction[row_idx, col_idx]
            curr_magnitude = cell_magnitude[row_idx, col_idx]

            diff = np.abs(curr_direction - hist_bins)
#             print(diff, np.where(diff == np.min(diff)))
            try :
                min_idx = np.where(diff == np.min(diff))[0][0]

                if(min_idx == 0):
                    HOG_cell_hist[min_idx] += curr_magnitude

                elif(min_idx == hist_bins.size-1):
                    HOG_cell_hist[min_idx] += (abs(curr_direction - hist_bins[-1])/bins_gaps)*curr_magnitude
                    HOG_cell_hist[min_idx-1] += (abs(curr_direction - hist_bins[-2])/bins_gaps)*curr_magnitude
                else:
                    HOG_cell_hist[min_idx] += (abs(curr_direction - hist_bins[min_idx])/bins_gaps)*curr_magnitude
                    HOG_cell_hist[min_idx+1] += (abs(curr_direction - hist_bins[min_idx + 1])/bins_gaps)*curr_magnitude
            except:
                continue

    return HOG_cell_hist
```

```
def generate_HOG(mag, theta, image, ratio = 0.25, plot = False):

    m,n = image.shape

    bins = np.array([-135, -90,-45, 0, 45, 90, 135, 180])
    HOG_hist = np.array([])

    for i in range(0,m, int(m*ratio)):
        for j in range(0,n, int(n*ratio)):

            curr_mag = mag[i : i+int(m*ratio),j : j+int(n*ratio)]
            curr_direc = theta[i : i+int(m*ratio),j : j+int(n*ratio)]
            bins = np.array([-135, -90,-45, 0, 45, 90, 135, 180])
            hist = HOG_cell_histogram(curr_direc, curr_mag, bins)
            if(plot):
                fig = plt.figure()
                ax = fig.add_axes([0,0,1,1])
                x_axis = ["-135", "-90","-45", "0", "45", "90", "135", "180"]
                ax.bar(x_axis,hist)
                plt.show()
            HOG_hist = np.concatenate((HOG_hist,hist))

    return HOG_hist
```

Generated Histograms for the 4 patches of the image :




```
print("The HOG vector is :", HOG_hist)
```

```
The HOG vector is : [341924.19920169  59117.1450625  369739.03985475 306335.25845045
 251922.69229994 235904.47076609 457878.95133474 275732.32372297
 242254.95258691  84358.41472952 446924.71205711 319780.23196215
 361819.98730511 193749.95684741 353645.53135928 293397.94790732
 604223.02115725 104418.51306052 536147.97630065 436636.48648831
 430202.1373679  503188.11846817 804630.40704936 458388.38522324
 400195.26249003 100923.53328205 494990.79147338 558429.48014063
 532339.94938353 427114.63601463 765009.55518977 539939.36452142]
```