Computer Vision 1: Exercise Sheet 10

Summary:

1. The computation of gradients in SIFT and HOG
2. The cells in HOG

1 The computation of gradients in SIFT and HOG

The gradient of a pixel at position (x, y) in an image is computed as:

$$\begin{aligned} g_x &= L(x+1, y) - L(x-1, y) \\ g_y &= L(x, y+1) - L(x, y-1) \end{aligned} \tag{1}$$

with the x-axis runs through the rows and the y-axis runs through the columns. In `scikit-image`, g_x, g_y are denoted as g_{row}, g_{col}, respectively. For g_{row}, the gradients at the border rows (the first and the last rows) are 0. For g_{col}, the gradients at the border columns (the first and the last columns) are 0.

- Generate a 3×3 binary image with two white pixels at positions $(0, 1)$ and $(1, 0)$. The image should look similar to the left image in Figure 1.
- What is the only pixel with the non-zero gradient? What are the gradient components?
- Compute the gradients of all pixels in this image. You can use the function `skimage.feature._hog._hog_channel_gradient`.

  Note: if the version of `scikit-image` in your computer does not have this function, please go to the source code on Git at `https://github.com/scikit-image/scikit-image/blob/master/skimage/feature/_hog.py` and copy the function `_hog_channel_gradient` there.
- (On paper) Draw the non-zero gradient. What are its magnitude and orientation?

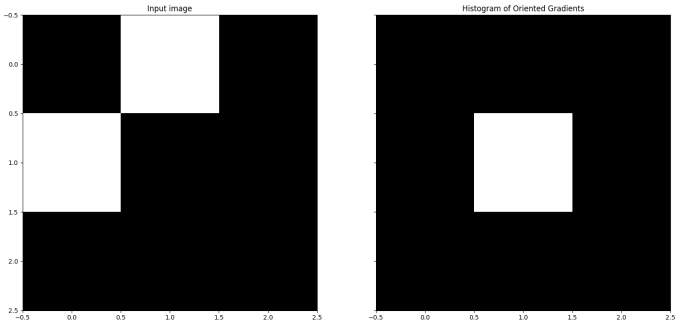

Figure 1: A 3×3 test input image (left) and its histogram of gradients (right).

2 The cells in HOG

As mentioned in the lecture, the histograms of gradients are the descriptors of cells of an image. In this exercise, the term "cell" is used to describe any square region in an image with size $K \times K$. For example, a cell with $K = 4$ has $4 \times 4 = 16$ pixels in total.

In `scikit-image`, the function for computing HOG features is `skimage.feature.hog`. The number of bins for the gradient histograms is defined by the parameter `orientations`. The size of each cells is defined by the parameter `pixels_per_cell`. This function returns a feature vector and a HOG image visualizing the histograms. To include this HOG image in the output of the function, set the parameter `visualize=True`. In this HOG image[1], each pixel at the center of each cell has a value equivalent to the sum of all bins of the normalized histogram of that cell.

- Compute the feature vector and the HOG image of the 3×3 binary test image generated above with `orientations = 1, pixels_per_cell=(3, 3), cells_per_block=(1, 1)`. Visualize the HOG image in `matplotlib`. It should look similar to the right image in Figure 1.
- Why does the non-zero pixel of this HOG image have value $0.1571348 \approx \frac{1.414}{9}$?
- Why does the HOG feature vector have size 1? Hint: how many cells does the 3×3 input image have?
- Generate another binary image of size 6×6 with each of its four 3×3 quadrants identical to the 3×3 image we have. Compute the HOG image for this new image. See Figure 2 for the expected output. This task should verify that the cells in an image are non-overlapping.
- Why does the HOG feature vector of this new image have size 4? Hint: how many cells does the 6×6 input image have?

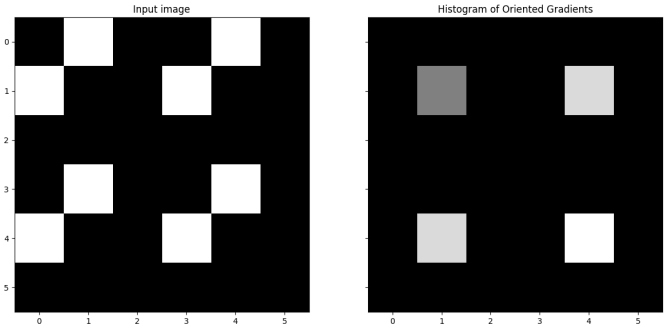

Figure 2: A 6×6 test input image (left) and its histogram of gradients (right).

[1]Recall that in HOG, the cells are divided so that they do NOT overlap each other.