Pattern recognition systems – Lab 5

Histograms of Oriented Gradients

1. Objectives

Histogram of Oriented Gradient descriptors, or **HOG** descriptors, are feature descriptors used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image.

The purpose of this lab is to implement the algorithm of extracting Histogram of Gradient Orientation (HOG) features. These features will be used then for classification and object recognition.

2. Theoretical Background

HOG features have been introduced by Navneed Dalal and Bill Triggs [1] who have developed and tested several variants of HOG descriptors, with differing spatial organization, gradient computation and normalization methods.

The essential thought behind the Histogram of Oriented Gradient descriptors is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The implementation of these descriptors can be achieved by dividing the image into small connected regions, called cells, and for each cell compiling a histogram of gradient directions or edge orientations for the pixels within the cell. The combination of these histograms then represents the descriptor. For improved performance, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. This normalization results in better invariance to changes in illumination or shadowing.

3. Algorithm Implementation

3.1. Gradient Computation

The first step of calculation is the computation of the gradient values.

The most common method is to apply the 1D centered point discrete derivative mask in both the horizontal and vertical directions. Specifically, this method requires filtering the grayscale image with the following filter kernels:

$$D_X = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \text{ and } D_Y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

Figure 3.1 Initial Image

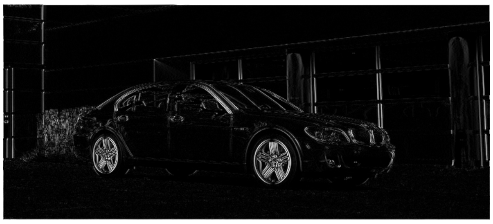


Figure 3.2 left: *X*-derivative of the initial image; right: *Y*-derivative of the initial image

So, being given an image I, we obtain the x and y derivatives using a convolution operation: $I_X = I * D_X$ and $I_Y = I * D_Y$

The magnitude of the gradient is $|G| = \sqrt{I_X^2 + I_Y^2}$

The orientation of the gradient is given by: $\theta = \arctan \frac{I_Y}{I_X}$

3.2. Orientation Binning

The second step of calculation involves creating the cell histograms. Each pixel within the cell casts a **weighted vote** for an orientation-based histogram channel based on the values found in the gradient computation. The cells themselves are rectangular and the histogram channels are evenly spread over 0 to 180 degrees or 0 to 360 degrees, depending on whether the gradient is "unsigned" or "signed". Dalal and Triggs found that unsigned gradients used in conjunction with 9 histogram channels performed best in their experiments. As for the vote weight, pixel contribution can be the gradient magnitude itself, or the square root or square of the gradient magnitude.

Figure 3.3 Initial Image

Figure 3.4 Magnitude of Gradient

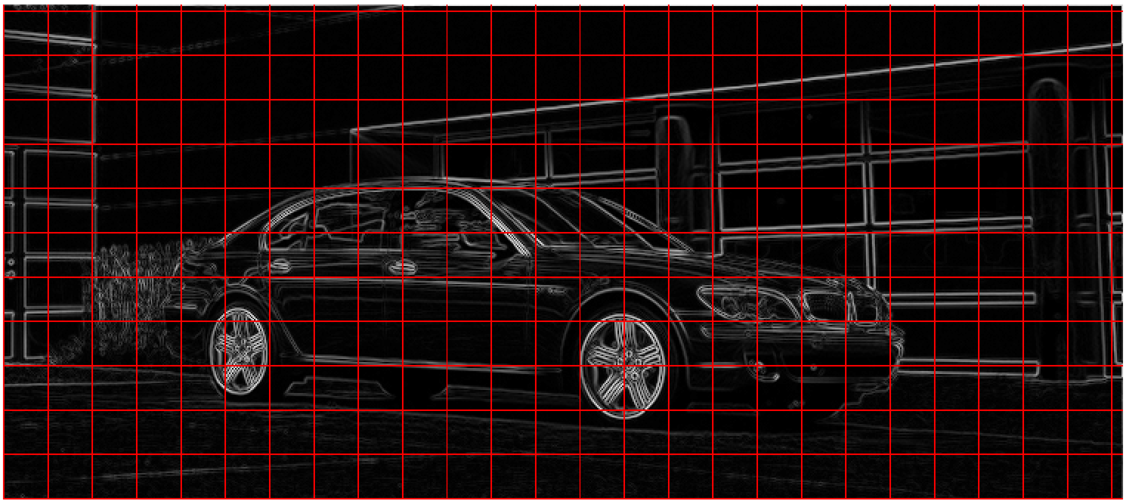


Figure 3.5 Cell division example

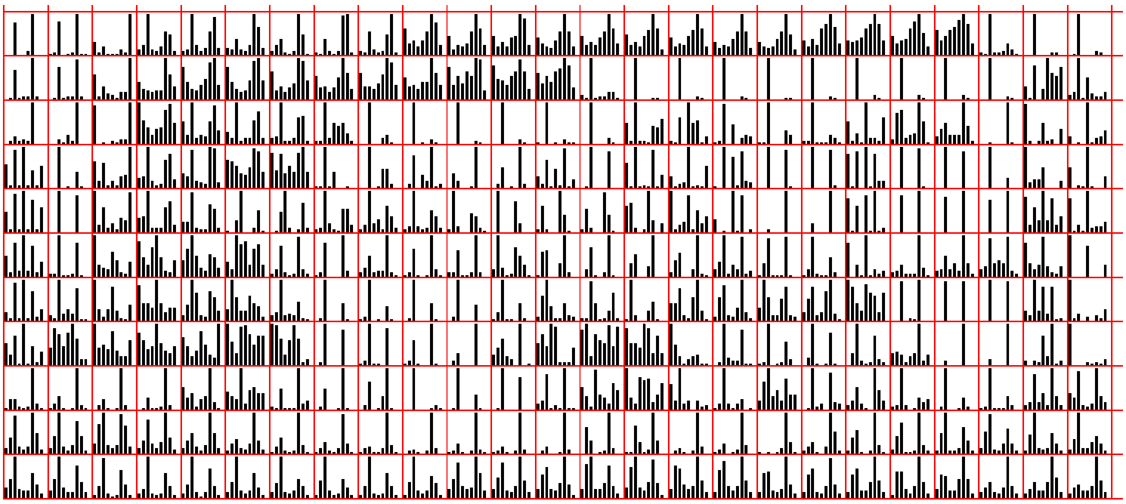


Figure 3.6 Histogram of Oriented Gradient descriptors (normalized inside each cell)

3.3. Descriptor Blocks

In order to account for changes in illumination and contrast, the gradient strengths must be locally normalized, which requires grouping the cells together into larger, spatially-connected blocks. The HOG descriptor is then the vector of the components of the normalized cell histograms from all of the block regions. These blocks typically overlap, meaning that each cell contributes more than once to the final descriptor. Two main block geometries exist: rectangular R-HOG blocks and circular C-HOG blocks. R-HOG blocks are generally square grids, represented by three parameters: the number of cells per block, the number of pixels per cell, and the number of channels per cell histogram.

3.4. Block Normalization

There are different methods for block normalization. Let v be the non-normalized vector containing all histograms in a given block, $\| v_k \|$ be its k-norm for $k = 1, 2$ and e be some small constant (whose value will not influence the results). Then the normalization factor can be one of the following:

$$\text{L2-norm: } f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}}$$

$$\text{L1-norm: } f = \frac{v}{\|v\|_1 + e}$$

$$\text{L1-sqrt: } f = \sqrt{\frac{v}{\|v\|_1 + e}}$$

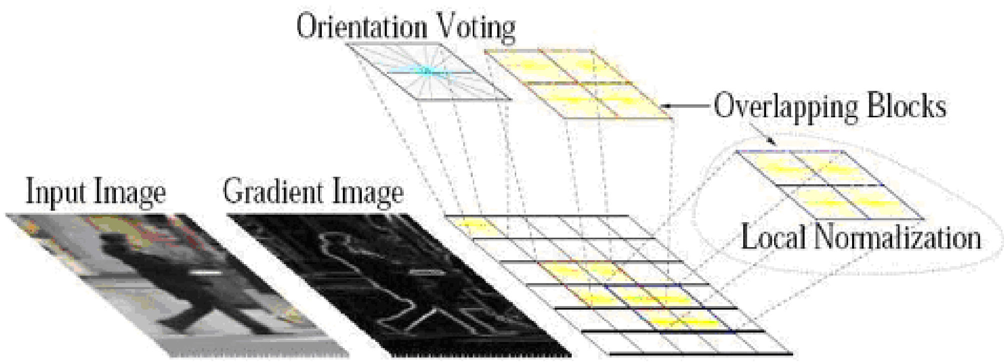


Figure 3.7 Block normalization scheme

4. Similarity Measure

For comparing the similarity of two vectors we can use several metrics.

1. **Euclidean metric**
   Given two vectors $P = (p_1, p_2, \ldots, p_n)$ and $Q = (q_1, q_2, \ldots, q_n)$ the distance is computed as:
   $$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$$

2. **Cosine similarity** is a measure of similarity between two vectors of n dimensions by finding the cosine of the angle between them. Given two vectors of attributes, A and B, the cosine similarity, θ, is represented using a dot product and magnitude as:
   $$similarity = \cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|}$$

   If the vector $A = (p_1, p_2, \ldots, p_n)$ and $B = (q_1, q_2, \ldots, q_n)$ then the similarity between them is given by:
   $$\cos(\theta) = \frac{p_1 q_1 + p_2 q_2 + \ldots + p_n q_n}{\sqrt{p_1^2 + p_2^2 + \cdots + p_n^2} \cdot \sqrt{q_1^2 + q_2^2 + \cdots + q_n^2}}$$

5. Implementation Details

Being given an image, we will extract the gradient magnitude and orientation as follows:
$|G| = \sqrt{I_X^2 + I_Y^2}$, where:

$$I_X = I * D_X \ , \ I_Y = I * D_Y \ ,$$

$$D_X = [-1 \quad 0 \quad 1] \ , \ D_Y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

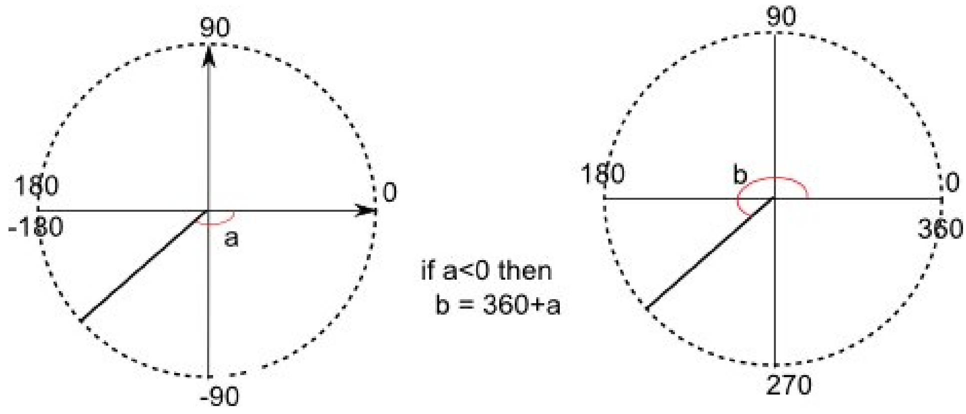
$*$ is the convolution operator

For the orientation, we will use the function *atan2* that returns a value in the interval $(-\pi, \pi]$.
So, the orientation of the gradient, for a pixel is $\theta = atan2(I_Y, I_X)$ radians.

The angle transformed into degrees is $\alpha = \theta * 180/\pi$, that will give values in the range (-180, 180] degrees.

For the ‘signed’ gradient we will need to ‘translate’ the range of the gradient from (-180, 180] to [0, 360) degrees. This is done using the formula:

$$\alpha_{signed} = \begin{cases} \alpha, if \ \alpha \geq 0 \\ \alpha + 360, if \ \alpha < 0 \end{cases}$$



6. Practical Work

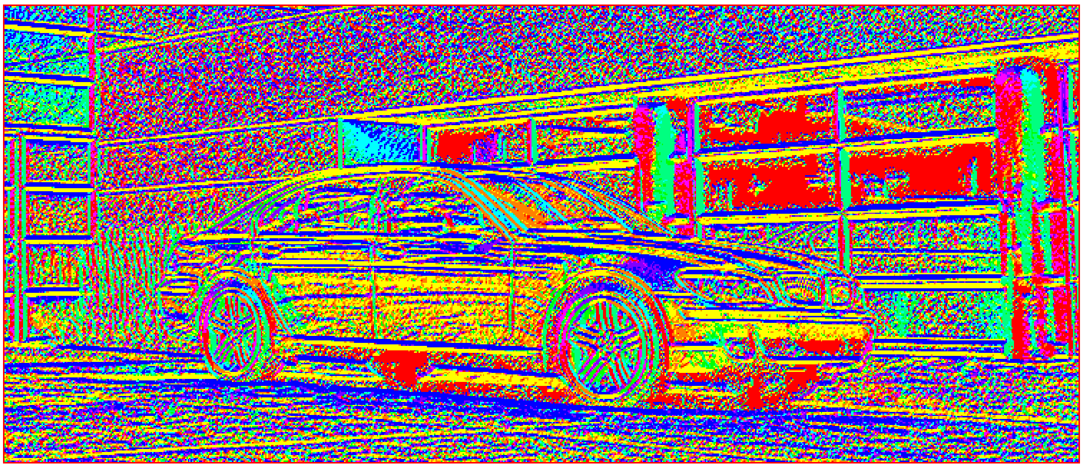
You will use as input, grayscale images with 8b/px: *car1.bmp*, *car2.bmp*, *pedestrian1.bmp*, *pedestrian2.bmp*, *other.bmp*

1. Add a new function to Diblook that computes the magnitude of the gradient and the orientation using the given formulas. Display the gradient magnitude, normalized in the range [0, 255].

2. Add a new function to Diblook that represents all the pixels gradient orientation according to the following color encoding in *nb_bins = 9* bins:

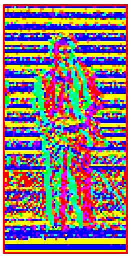
Bin number	Orientation degrees range	Color {R,G,B}
0	[0, 40)	{255, 0, 0}
1	[40, 80)	{255, 128, 0}
2	[80, 120)	{255, 255, 0}
3	[120, 160)	{0, 255, 0}
4	[160, 200)	{0, 255, 128}
5	[200, 240)	{0, 255, 255}
6	[240, 280)	{0, 0, 255}
7	[280, 320)	{128, 0, 255}
8	[320, 360)	{255, 0, 255}

Notice that you might need to change the palette of the image for performing this operation (for drawing colored pixels). For the image *car1.bmp* the result should look like:

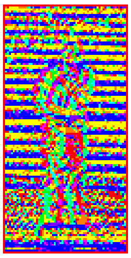


The same result for images:

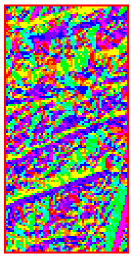
pedestrian1.bmp



pedestrian2.bmp



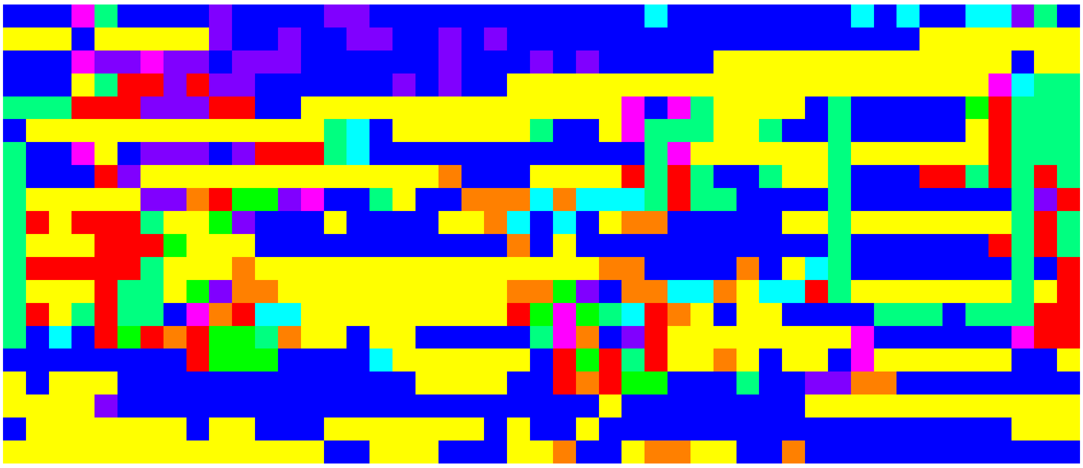
other.bmp



3. Implement a function that computes the HOG features for a given image divided in a specified number of rectangular cells. Each cell will be filled in with the corresponding color of the bin that has the maximum accumulated value. Use the signed gradient when computing the HOG features. Do not use any normalization scheme (just steps 1 and 2 of the algorithm implementation presented in the theoretical background).

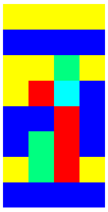
   For this operation implement a dialog box from which you will read the cell width, cell height and the number of bins of a cell.

   For image *car1.bmp* the result for *nb_bins = 9*, *cell_width = 16*, *cell_height = 16* should look like:

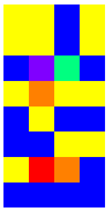


The result for other input images:

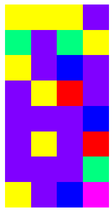
pedestrian1.bmp



pedestrian2.bmp



other.bmp



4. Find the similarity between two images (objects) by the cosine similarity measure applied on the corresponding HOG features vectors (each containing a concatenation of all the bin values from all image cells => *nb_bins* x *nb_cells* dimensions). Use the following parameters: *nb_bins = 9, cell_width = 16, cell_height = 16.*

   The cosine similarity measure between images *pedestrian1.bmp* and *pedestrian2.bmp* should be approximately *0.80*

   The cosine similarity measure between images *pedestrian1.bmp* and *other.bmp* should be approximately *0.39*

7. References

[1] N. Dalal, B. Triggs: *Histograms of oriented gradients for human detection,* IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005.