HOG “histogram of gradients”

It generalizes the object in a way that the output for the same object is as close as possible under different conditions, this is called feature descriptor.

Input: image of object (face in this case)

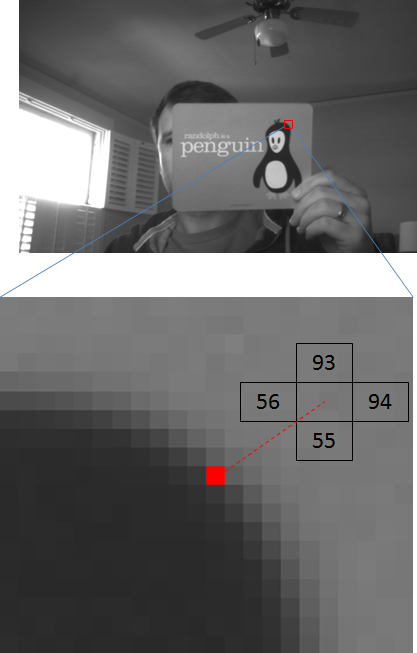
Output: vector of histograms of gradients

Method:

* Calculating gradient vector:

It is a measure of the change in pixel value along x-direction and y-direction for each pixel.

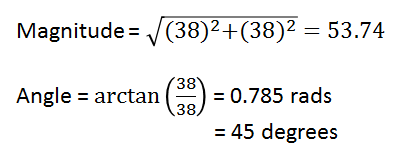
Ex:



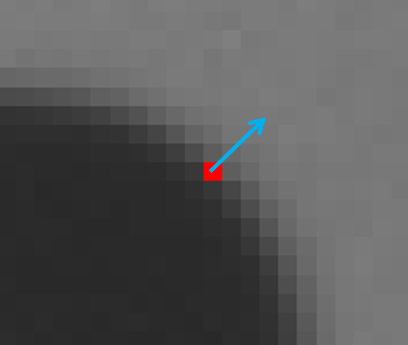
* This is a grayscale image, so the pixel values just range from 0 - 255 (0 is black, 255 is white). The pixel values to the left and right of our pixel are marked in the image: 56 and 94. We just take the right value minus the left value and say that the rate of change in the x direction is 38 (94 - 56 = 38).
* You can compute the gradient by subtracting left from right or right from left, you just have to be consistent across the image as the only change is in the sign which will be ignored in the following calculations.
* Then calculate the change along y-direction in the same way (93 – 55 = 38).
* Now we have our gradient vector:



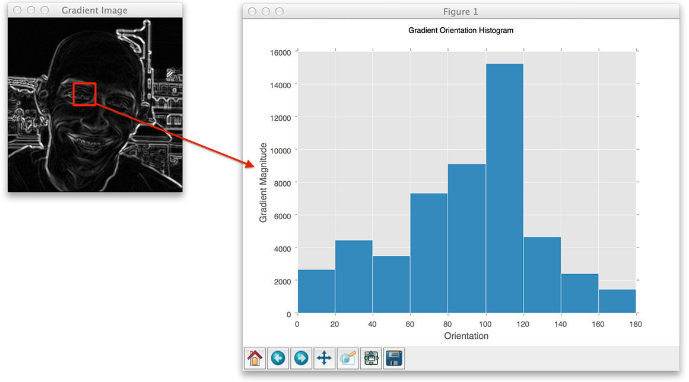
* We can use these equations to represent this vector as magnitude and angle.



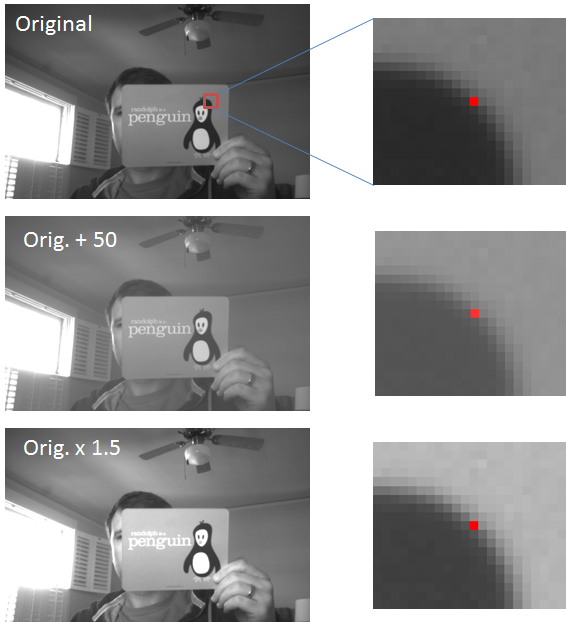
* the angles are between 0 and 180 degrees instead of 0 to 360 degrees. These are called **“unsigned” gradients** because a gradient and it’s negative are represented by the same numbers. In other words, a gradient arrow and the one 180 degrees opposite to it are considered the same. But, why not use the 0 – 360 degrees? Empirically it has been shown that unsigned gradients work better than signed gradients.
* We can now draw this vector as an arrow on the image and this operation is applied to each pixel in the image.



* Combining histogram.
* We use histogram with n bins (8, 9, 12, …) and we choose n which give us better accuracy.
* We use histogram with n = 8 according to this paper.
* angles on horizontal axis and magnitude on vertical axis.
* For each gradient vector, it’s contribution to the histogram is given by the magnitude of the vector (so stronger gradients have a bigger impact on the histogram). We split the contribution between the two closest bins. So, for example, if a gradient vector has an angle of 85 degrees, then we add 1/4th of its magnitude to the bin centred at 70 degrees, and 3/4ths of its magnitude to the bin centred at 90.



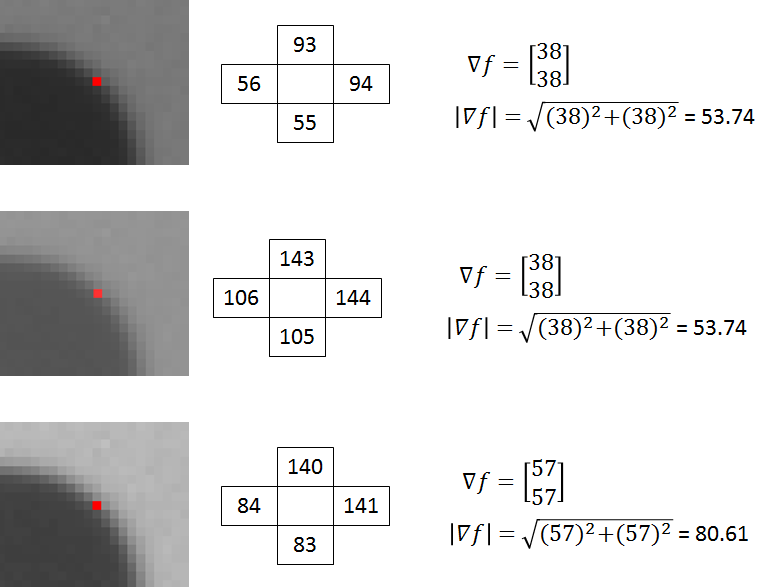
* How it’ll be applied on the image?
* We divide the image into cells of (1X1, 4X4, …..) and we choose the number of pixels per cell according to which give us the best accuracy.
* We use 12X12 pixels per cell.
* Now we have multiple histogram for a single image we combine them sequentially in a vector.
* Normalization
* Let’s take a moment to first look at the effect of normalizing gradient vectors in general.
* By normalizing your gradient vectors, you can make them invariant to *multiplications* of the pixel values (different illumination conditions)
* Ex:



(caption)[The first image shows a pixel, highlighted in red, the second image, all pixel values have been increased by 50, the third image, all pixel values in the original image have been multiplied by 1.5]

Notice how the third image displays an increase in contrast. The effect of the multiplication is that bright pixels became much brighter while dark pixels only became a little brighter, thereby increasing the contrast between the light and dark parts of the image.

Let’s look at the actual pixel values and how the gradient vector changes in these three images. The numbers in the boxes below represent the values of the pixels surrounding the pixel marked in red.



* The gradient vectors are equivalent in the first and second images, but in the third, the gradient vector magnitude has increased by a factor of 1.5.

If you divide all three vectors by their respective magnitudes, you get the same result for all three vectors: [0.71  0.71]’.

* By dividing the gradient vectors by their magnitude we can make them invariant (or at least more robust) to changes in contrast. (normalizing to unit length)
* Normalizing a vector does not affect its orientation, only the magnitude.
* **Histogram normalization**
* Rather than normalize each histogram individually, the cells are first grouped into blocks and normalized based on all histograms in the block.
* Each block consists of nXn cells 50% overlapping and we choose to work with 4X4.
* The effect of the block overlap is that each cell will appear multiple times in the final descriptor but normalized by a different set of neighbouring cells.
* Ex :



[caption](2X2, 50% overlapping)

* All chosen parameters according to this paper :

<https://link.springer.com/content/pdf/10.1007%2F978-3-642-02172-5_24.pdf>