# RDFIA Section 1 - HackMD

# 1 - SIFT

**1)** ***Show that kernels Mx and My are separable, i.e. that they can be written $M_x = h_y h_x$ and $M_y = h_x h_y$ with $h_x$ and $h_y$ two vectors of size 3 to determine.***

The matrix $M_x$ and $M_y$ can be easily computed from two vectors $h_x = [-1, 0, 1]$ and $h_y = [\frac{1}{4}, \frac{1}{2}, \frac{1}{4}]$.

Indeed,

$$h_y \times h_x^T = \begin{bmatrix} 1/4 \\ 1/2 \\ 1/4 \end{bmatrix} \times \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1/4 & 0 & 1/4 \\ -1/2 & 0 & 1/2 \\ -1/4 & 0 & 1/4 \end{bmatrix} = M_x$$

And the same goes for $M_y$.

This decomposition is also interesting as it highlights the two underlying kernels used in Sobel:

- The derivative kernel $h_x$
- The gaussian smoothig kernel $h_y$

**2)** ***Why is it useful to separate this convolution kernel?***

The convoluted image $J$ with the Sobel filter $M_x$ applied on image $I$ can be computed in two different ways:

$$J = I \times M_x$$

$$J = I \times (h_y \times h_x^T)$$

As the convolution operation is associative, the second equation allows the convolution of $I$ with $h_y$ and then with $h_x^T$. The convolution with smaller kernels ($1 \times 3$ and $3 \times 1$ kernels against a $3 \times 3$ one) allows faster computation. Indeed, this reduces the computational costs from $O(M \times N)$ to $O(M + N)$ for an $M \times N$ kernel.

**3)** ***What is the goal of the weighting by gaussian mask?***

The gaussian mask highlights the center of each patch, which improves the stability of the SIFT alogrithm. Indeed, the goal of visual words is to described the specificity of a patch (like a specific edge or an angle). The latter is represented by a local context of pixels. To gather this local context, the “focus” of a patch should be set on its center. A local context on the edge of a patch could not be fully observed, and therefore will be better described by the next overlapping patch. This need to “focus” the local context on the center of a patch is implemented with the gaussian mask.

**4)** ***Explain the role of the discretization of the directions.***

The discretization of the directions is a kind of pooling which makes the SIFT algorithm invariant to rotation and have consistent descriptors. Because the descriptors are calculated relatively to the orientation of the image, the rotation does not matter anymore.

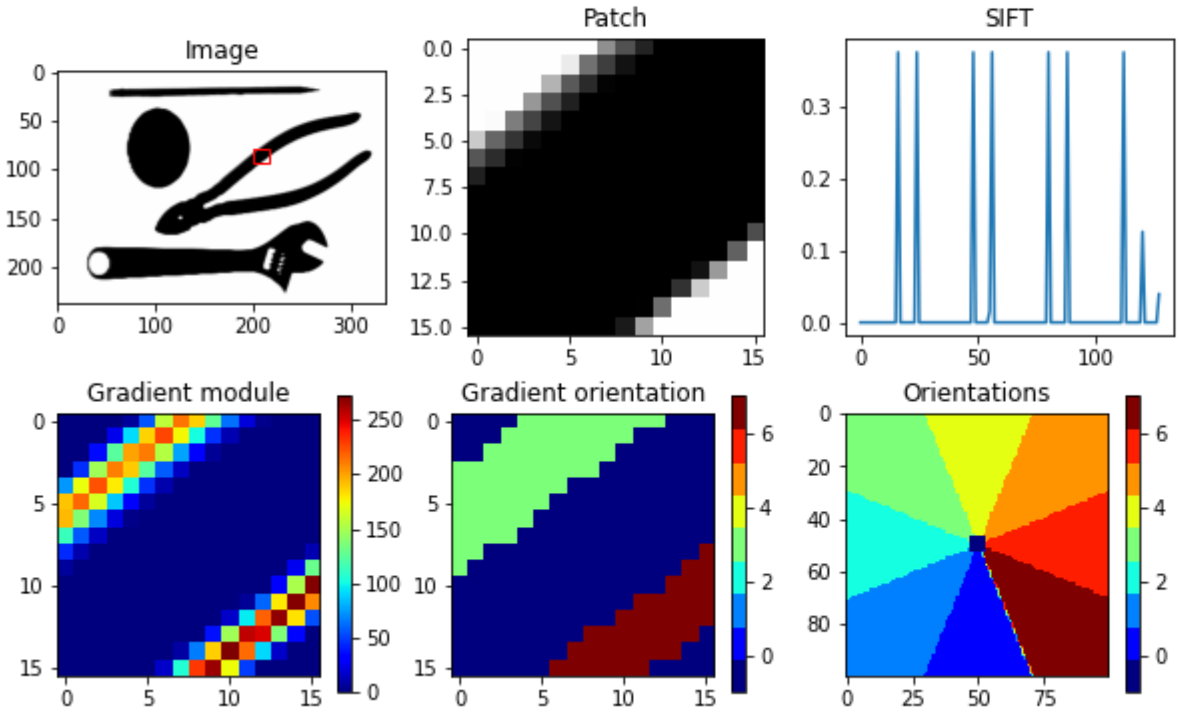
**5)** ***Justify the interest of using the different post-processing steps.***

- Using a threshold to discard vectors that have a small norm (< 0.5) helps to summarize our image with only its most important desciptors.
- Normalizing the descriptors removes the intensity of the underlying gradient from the SIFTs. It therefore focuses the SIFT on the direction of the most important gradient rather than their intensity. It also reduces the diversity of descriptors that will compose the Visual Dictionary, making it more robust to luminity variation for example.

**6)** ***Explain why SIFT is a reasonable method to describe a patch of image when doing image analysis.***

In image analysis, it is often desired to compare, identify or match two patch representing the same object (or part of an object) in two different images. Unfortunately, images are rarely taken in the exact same condition. Therefore, to robustly compare patches, we need them to be invariant to translation, flip, rotation, and scale. SIFT provides all those invariances.

**7)** ***Interpret the results you got in this section.***



The colored pictures are showing the intensity of the gradient and its orientation. This is why the edges on the bottom left pictures are colored in red. Because we divided our orientation space in eight, the bottom-center image has green (north-west) gradient on the upper edge and red (south-east) on the lower edge. The spikes in the histogram are the gradient intensity for each 4x4 window of the patch. We can observe eight spikes regrouped in two because the sift method computed with consecutive 4x4 window the gradient on the edges.

# 2 - Visual Dictionary

**8)** ***Justify the need of a visual dictionary for our goal of image recognition that we are currently building.***

SIFTs as is are not handful for the image recognition task. There are two main reasons to use a visual dictionary to compress them into a global descriptor:

- The variable encoding size of the vector of SIFTs is not handy. Indeed, depending on the size of a given image, the number of SIFTs can vary a lot. But if the encoding varies from image to image, how can we set the size of the input layer of a neural network for example? This why a fixed size encoding is prefered for image classification. This is one clear advantage of a global descriptor.
- A global descriptor also signicantly compresses the information. The visual dictionary step compress information from a large number of local descriptors (typicaly about a thousand) into a single global descriptor. This significantly reduces the size of the encoding in the process. As an example, an image of 512x512 pixels contains 1024 local descriptors. Each of them has 128 dimensions. Therefore, the total dimensions of the gathered local descriptors for a given image is about 100 000. Agglomerating them into a global descriptor usually reduces its dimension to a thousand. This compression can be more radical if desired, as it only depends on an hyperparameter. A compressed encoding as this one tends to improve the image recognition task.

**9)** ***Considering the points $\{x_i\}_{i=1..n}$ assigned to a cluster $c$, show that the cluster's center that minimize the dispersion is the barycenter (mean) of the points $x_i$:***

$$c = \min_c \sum_i ||x_i - c||_2^2 f = argmin_c \sum_i^n (x_i - c)^2$$

Now that we have the formula of the function we want to minimize, we can look for its derivative :

$$\frac{\partial f}{\partial c} = -2\sum_i^n (x_i - c)$$

The minimum will be obtained when the derivative is equal to zero :

$$\frac{\partial f}{\partial c} = 0 <=> -2\sum_i^n (x_i - c) = 0 <=> c = \frac{1}{n}\sum_i^n x_i$$

As we can see, this is the formula of a barycenter because it is a mean between points.

**10)** ***In practice, how to choose the "optimal" number of clusters?***

To have the optimal number of clusters, we need to make a trade-off between the number of clusters and the inertia. Inertia is the sum of squared distance of samples to their closest cluster center. For that, we can use the elbow method where we plot the inertia in function of the number of clusters. We choose the elbow point in the inertia graph where the improvement in the inertia value is not significant.

**11)** ***Why do we create a visual dictionary from the SIFTs and not directly on the patches of raw image pixels?***

The the SIFT provides an invariance on rotation, scale and reflectance.
This is really important if we want to have a universal dictionary to describe our images. Moreover, the sift method computes a smaller vector than the patch itself, thus we are compressing the memory we need for our dictionary.

We can see that each cluster center represents a different type of patch. For example, there is one that describe patches with vertical black line, one with diagonal etc…

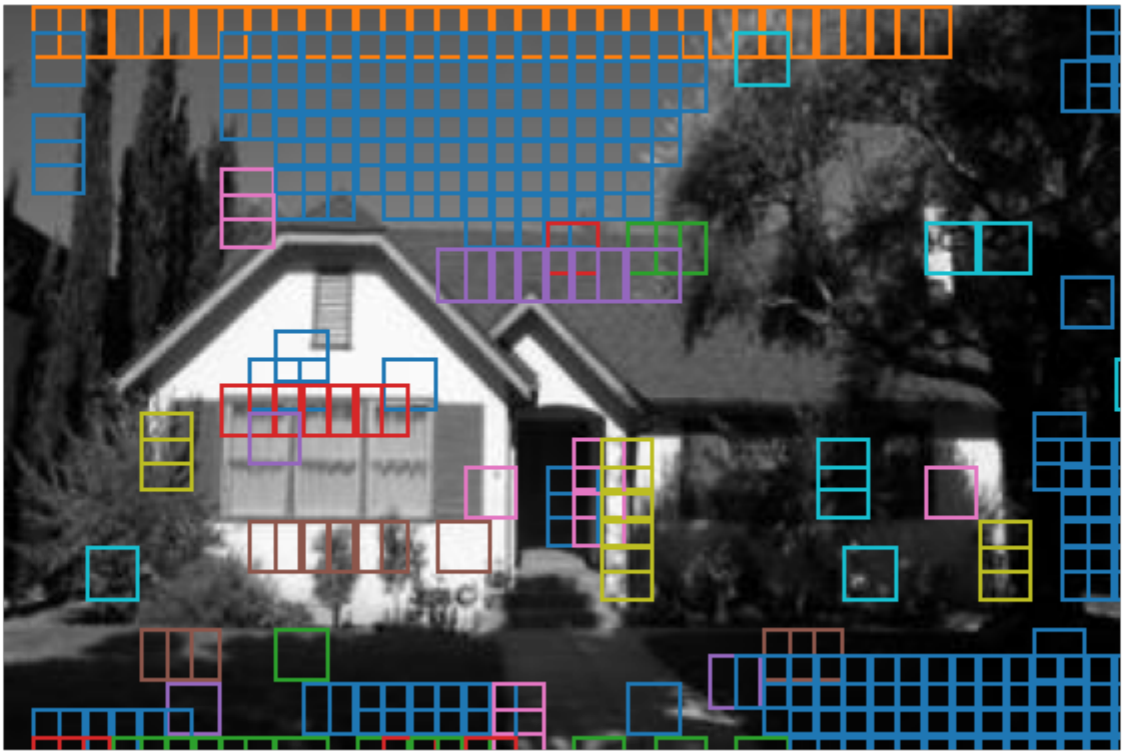
# 3 - Bag of Words (BoW)

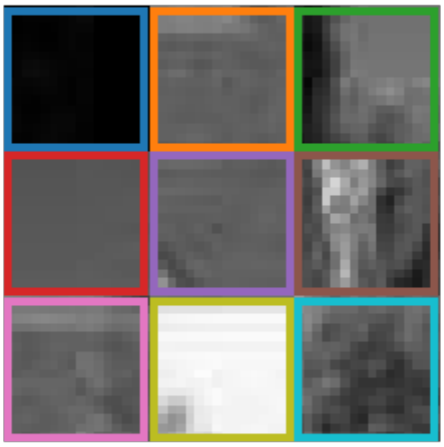
**13)** ***Concretely, what does the vector $z$ represent of the image?***

The vector $z$ is the histogram of visual words. Indeed, each descriptor is encoded as a one-hot and are then sum up in the $z$ vector. This corresponds to the definition of an histogram.

**14)** ***Show and discuss the visual results you got.***

Here are the results we obtained:





Even though it is a bit hard to interpret all the results, we can see that the uniform regions are correctly represented by the blue SIFT (which is the null SIFT we manually inserted in the Visual Dictionary). Some vertical signal are also detected in the yellow SIFT. Lastly, the added padding is detected by the orange SIFT.

**15)** ***What is the interest of the nearest-neighbors encoding? What other encoding could we use (and why)?***

The nearest neighbour encoding the a fast to implement and simple encoding that has good results.

Alternatives:

- Could use GMM instead of kmeans: more stable on complex to separate words
- Could use an encoding based on the distance of this visual word to each word from the visual dictionary: finer encoding of words

**16)** ***What the interest of the sum pooling? What other pooling could we use (and why)?***

The sum pooling is interesting because it highlights the most frequents words in the image.
This can be also a problem because we are not giving importance to descriptors that could discriminate localy when comparing the images.

Other pooling exists such as max pooling and mean pooling.

**17)** ***What is the interest of the L2 normalization? What other normalization could we use (and why)?***

The L2 normalization is used to obtain a unitary vector that is comparable. We could use other norms too because every distance are equivalent.

# 4 - SVM (Work 1-c)

**1)** ***Discuss the results, plot for each hyperparameters a graph with the accuracy in the y-axis***

We plotted in the notebook the graph of the accuracy depending on the value of C. We can see that there is a clear value of C that maximize the accuracy on the validation set.

### 2) ***Explain the effect of each hyperparameter***

- C: Is a regularization parameter and is linked to the notion of “Soft Margin” of SVM. The C hyperparameter is the coefficient upfront the penality term for missclassified samples in the cost function. The bigger the C, the more penalty SVM gets when it makes misclassification.
- Kernel: Specifies the type of hyperplane used to separate the data. Using ‘linear’ will use a linear hyperplane (a line in the case of 2D data). ‘rbf’ and ‘poly’ uses a non linear hyper-plane. Non-linear kernel can surely better separate data, but may suffer from over-fitting depending on the nature of the dataset.
- Decision function: either “one-vs-all” or “one-vs-one”. This hyper parameter comes from the fact that SVM is a binary classifier. This decision function is necessary to enable multi-class classification. In “one-vs-all” mode, $n$ binary classifiers are trained. Each of them classify one class against all the others. The “one-vs-one” mode train $\frac{n(n-1)}{2}$ binary classifiers. Each binary classifier is trained to distinguish two classes. In both mode, the consensus is established on the confidence score of the prediction of each classifier.

### 3) ***Why the validation set is needed in addition of the test set ?***

The validation and test sets have very different roles. The test set should only be used to control the coherence of its result with the training set and therefore detect under/ over fitting. On the other hand, validation set is used to tune the hyperparameters of the model. If this seperation is not respected, and we instead use the test set for hyperparameter tuning, we will, in this case, over fit our model to this dataset. The model could lead to surprising result once in production.