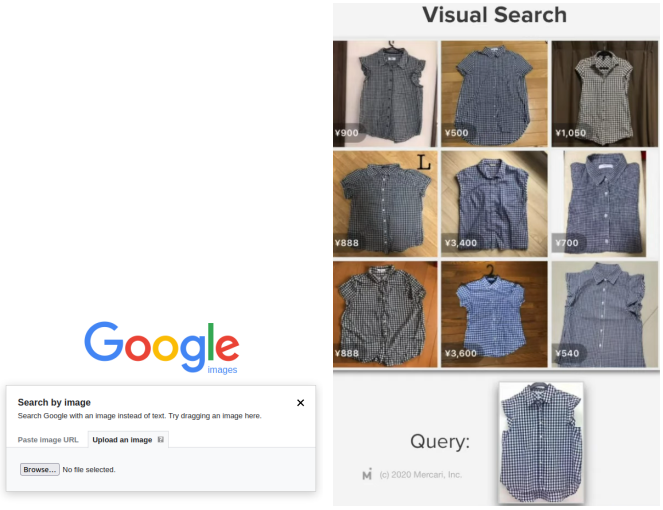
Low-level Vision-based Image Retrieval for Museum Paintings

Marcos V. Conde, José M. Lopez-Camuñas and Alex Martin-Martinez

***Abstract*— Computer Vision is arguably one of the most important areas in the Artificial Intelligence field. Machine Learning and Deep Learning revolution also affects this area notably, however, the fundamental geometric and algebraic Computer Vision techniques still undergoing in modern systems. This work covers such traditional Image Processing techniques, and their applications towards creating modern Image Retrieval and Recognition systems.**

I. INTRODUCTION

The Image Retrieval problem consists on finding the most similar images (in a database) to a given query image [1], [2]. This search is based only on images' features (*e.g.* edges, shapes, illumination, colors, contours, contrast). One common procedure is to extract the features [3], [4], [5] for each database image, then measure the similarity between the input image and the database images to extract the most similar one. We show modern Image Retrieval pipelines in Figures 1 and 2.



Google Image Retrieval E-commerce Image Retrieval

Fig. 1. Modern Image Retrieval systems.

Our main contributions are:

- A robust end-to-end pipeline for Painting Images Retrieval, suitable for real-time applications in museums, and based on pure image processing and optimization techniques, no machine learning is involved.
- Exhaustive ablation studies of different classical image descriptors and denoising techniques.

Universitat Politècnica de Catalunya (UPC), M1, Team 8
Project's page: https://github.com/MCV-2021-M1-Project/Team8
Contact:infomar97@gmail.com,
alexmartin299@gmail.com joseplcam@gmail.com

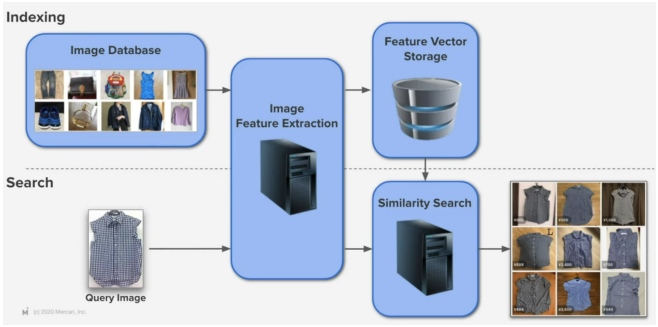


Fig. 2. Image Retrieval Pipeline from Mercari e-commerce.

II. RELATED WORK

A. Color Descriptors

In image processing, a color histogram is a representation of the distribution of colors in an image. Color histograms are flexible structures that can be used in different color spaces, whether RGB, HSV, YUV, CIELAB or even Grayscale images (then we refer intensity histogram).

A histogram of an image is produced by applying discretization to the colors in the image into a certain number of bins, and counting the number of pixels in each bin. Color histograms focus on the proportion of colors and statistical distribution, regardless of the spatial location of those colors.

B. Texture Descriptors

Texture descriptors characterize image textures, regions, edges and details (high-frequencies). They observe the region homogeneity and the histograms of these region borders. They can provide information that complements color descriptions, and thus, better characterize the image and retrieve similar images better.

The Histogram of Oriented Gradients (HOG) [6] is a feature descriptor used in Computer Vision and image processing for the purpose of objects and edges detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is similar to that of edge orientation histograms, scale-invariant feature transform (SIFT) [7] descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. We show qualitative results of HOG in Figure 3.

C. Keypoint Descriptors

Local features have many properties that make them suitable for matching different images of an object or scene.

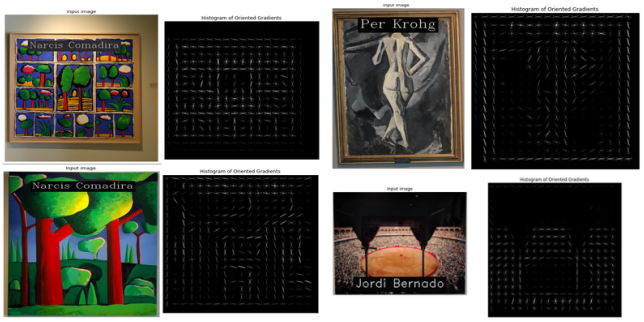


Fig. 3. Histogram of Oriented Gradients (HOG) Examples.

These features are invariant to image scaling, rotation, distortions or illumination changes. They are well localized in both spatial and frequency domains reducing the probability of disruption by noise, occlusion or other problems. Moreover, the features are highly distinctive, allowing proper matching in large database of features. We show keypoint extraction samples in Figure 6.

There are several efficient methods for local feature extraction [8], [7], [9].We integrate SIFT (Scale Invariant Feature Transform) in our pipeline as follows:

- **Scale-space extrema detection**: Search over different scales and image locations using DoG (Difference of Gaussian) to identify potential interest points which are invariant to scale and orientation.
- **Keypoint identification**: Each candidate must be evaluated based on their stability.
- **Orientation assignment**: Orientations are assigned to each keypoint-based on local image gradient directions. Operations will be performed relative to the assigned orientation, scale and locations to provide a robust method.
- **Keypoint Descriptor**: The local image gradients are measured at the selected scale in the region close to each keypoint. These are transformed so they can allow some distortion or illuminance changes.

D. Text Recognition

In the image there could also be other information than the present in the painting as it could be text with the author or the painting name. To extract information from the text present in the image to perform the retrieval, first the text has to be detected and to do so the usage of morphological operators comes in handy. Although it could perform a suitable text detection the kernels used to perform the different operation such as top hat, black hat or openings and closings, makes it a very difficult to optimize approach.

Despite this the zone where the text is placed in the image can be localized and with functions like the ones that pytesseract haves the text in the image is converted into strings. Once this type of data is acquired we can compare the detected strings with the ones present in the database in similar way than with the previous methods. As well as there's distance metrics for numerical data, there's also for text. In this case the distance between the detected text and the text that are in the database can be calculated with the Levenshtein [10] distance which is ideal for short strings.

Although this seems to be a good solutions to the retrieval problem the difficulty to detect the bounding boxes correctly and the presence of the same author in different paintings made this approach very inefficient which made us discard it.

III. Our Approach

First of all, we need to create a database of painting images captured in the same situations as we want our pipeline to work. Some examples of situations: photos properly focused, with the painting in the center of the image and taken from a close distance from the real picture in the museum. We define the database size (number of images in the database) as D.

We show a diagram explaining our pipeline in Figure 4. We proceed to describe each step into details:

1) We get the input query image, which can be any photo captured "in the wild" in the museum. The first step is to pre-process the image, this is: (i) get the image data from a *.png* or *.jpg* file, which will be a tensor $\mathcal{R}^{h\times w\times 3}$ where h is the height and w is the width of the image. (ii) we remove the noise from the image. (iii) rotate the image in case the image taken present rotation (in Section III-A we provide more details of steps(ii) and (iii)).
2) We apply a series of morphological transformations [11].
3) We crop the image based on the obtained mask from last step. Our method is constrained to extract up to 3 pictures from a single image.
4) For each extracted picture, we extract the following features: (i) Multi-block 3D Histograms or *color descriptor*, (ii) Histogram of oriented gradients (HoG) [6] or *texture descriptor*, (iii) local descriptor Keypoints using SIFT [7]. Therefore we have a color descriptor $\mathcal{C} \in \mathcal{R}^{512}$, a texture descriptor $\mathcal{T} \in \mathcal{R}^{300}$, and $\mathcal{K} \in \mathcal{R}^{N}$ keypoints (the number of keypoints N is variable).
5) For each picture, using the *texture descriptor* and *color descriptor*, we calculate the cosine similarity (Equation 1) between the query descriptors and each of the D descriptor in the database. We also calculate the number of keypoint-matches between the query keypoints and every image in the database.
6) We retrieve the top-k images from the database such that their distance to the query is the lowest, or the keypoint matches are higher.

$$similarity(A, B) = \frac{A \cdot B}{||A||_2 \times ||B||_2} \tag{1}$$

A. Results

The results are evaluated using the Mean Average Precision at a level $k = 10$, which is defined as follows:

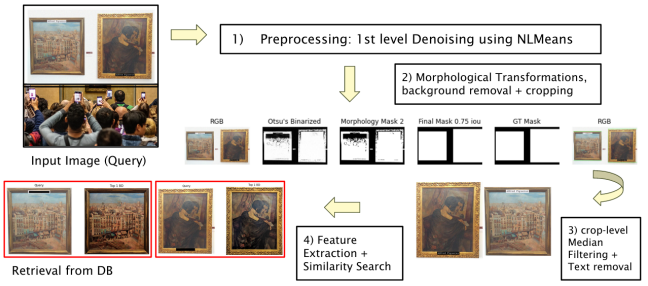


Fig. 4. Our proposed Image Retrieval pipeline for real-time applications in Museums.

$$mAP@10 = \frac{1}{Q}\sum_{q=1}^{U}\sum_{k=1}^{min(n,10)} P(k) \tag{2}$$

where Q is the number of test images, $P(k)$ is the precision at cutoff k and n is the number of predictions per image. In our experimental setup, given a query image, we predict the top-k closest (or more similar) images in our database, therefore $n = k = 10$. We then compare the retrieved images with the ground-truth and obtain the precision $P(k)$. Note that the top-k retrieved images are ordered by a score, this is explained in Section III. We also use 2 variants of this metric: mAP@5 and mAP@10 where we consider only the top-5 and top-10 predictions for scoring.

As we show in Section III, we need to remove the noise from the image to obtain better descriptors, as we show in Table I, most noise removal methods help to improve the performance using color and texture descriptors. We consider the non-linear Median Filter method the most consistent and robust one, improving always the performance with respect the original "Base Noisy image", therefore, we use this method as our default denoising step in the pipeline. At Figure 5 we show a qualitative comparison of the denoising methods. As explained in section II many of the descriptors used are invariant to the rotation but in case that the taken images were rotated this rotation was corrected to get a better performance of other descriptors which may be affected by it so we could compare the methods fairly. To perform this angular correction the steps followed were:(i)convert image to grayscale representation. (ii) apply Canny edges[12] algorithm to extract the edges. (iii) use the Hough Line Transform[13] to detect straight lines form the edges previously obtained. (iv) calculate the angle of each detected line to calculate the mean avoiding angle lower than 30º. This mean is the detected rotation and the image are corrected consequently to have straight paintings in the images.

In Table II we show the performance using all the proposed descriptors and their combinations. As we expected, keypoint-based descriptors are really powerful and can extract easily important information to recognize the image, and thus, retrieve images with similar features. We find SIFT the most powerful descriptor, however we must note that is the most time consuming one. In Figure 6 we show

TABLE I

ABLATION STUDY OF DIFFERENT DENOISING METHODS AND THEIR IMPACT ON THE RETRIEVAL SYSTEM USING COLOR AND TEXTURE DESCRIPTORS ON THE QSD1-W3 DATASET. WE HIGHLIGHT USING BLUE COLOR THE BASELINE PERFORMANCE WITHOUT REMOVING THE NOISE FROM THE IMAGE, AND IN RED COLOR THE BEST DENOISING METHOD.

Algorithm	mAP@10 Color	mAP@10 Texture
Base Noisy image	0.850	0.654
Low-pass Filter	0.867	0.775
Average Filter	0.867	0.775
Gaussian Filter	0.867	0.743
Median Filter	0.867	0.829
NL-Means [14]	0.850	0.903

TABLE II

ABLATION STUDY OF THE RETRIEVAL PERFORMANCE USING DIFFERENT IMAGE DESCRIPTORS ON THE QSD1-W5 (FINAL) DATASET. OVERALL SIFT [7] IS THE BEST DESCRIPTOR DUE TO ITS SCALE AND ROTATION INVARIANCE PROPERTIES.

Method	mAP@10	mAP@5
HoG+Multi-histogram [6]	0.5333	0.4944
SIFT [7]	0.7333	0.7130
ORB [15]	0.3333	0.3333
SIFT+ORB [7], [15]	0.7333	0.7130
SIFT+ (HoG+Multi-histogram)	0.7333	0.7296

qualitative samples of keypoint extraction and matching, the process is:

1) Extract keypoints from every image in the database. This is a very time-consuming process.
2) Extract keypoints from the given input query image.
3) Compare and match keypoints between the query image and every image in the database.
4) Retrieve the images in the database that have more matches with the query.

We can combine descriptors by aggregating their similarity scores. Defining T_s as the Texture descriptor scores, C_s the Color descriptor scores and K_s the Keypoint scores, we can combine them as follows:

$$S = \alpha \times T_s + \beta \times C_s + \theta \times K_s \tag{3}$$

where α, β and θ parameters are set empirically, for example: $\alpha = 0.1$, $\beta = 0.1$ and $\theta = 0.8$. The final score S for an image is the result of such aggregation.

TABLE III

FINAL COMPETITION BENCHMARK USING THE QSD1-W5 DATASET. OUR RESULTS (IN BOLD) ARE COMPETITIVE IN EVERY METRIC.

Team	mAP@10	mean IoU	Mean Angular Error	Method
1	0,53	0,851	0,86	SIFT [7]
2	0,63	0,843	3,04	ORB [15]
3	0,40	0,87	0,64	ORB [15]
4	0,83	0,7162	0,9172	ORB [15]
5	0,73	0,924	0,548	3DHist + DCT + ORB [15], [16]
6	0,83	0,924	0,593	Akaze [9]
7	0,83	0,82	0,33	ORB [15]
8	**0,73**	**0,875**	**0,8969**	SIFT [7]

Fig. 5. Result samples using different denoising methods. We can appreciate that NLMeans [14] and the Median Filter are the most effective methods.

(a) Picture from Per Krohg.

(b) Picture from Pablo Picasso.

Fig. 6. Keypoint matching examples using SIFT [7]. We can plot up to 200 keypoints that describe the image. We show 2 samples (a) and (b), for each sample, (left) the original image, (right) the image with most common keypoints retrieved from database.

B. Clustering

Although the main part of this work is image retrieval, image clustering was also performed. In this work, art and engineering were mixed to group the images from the database based on the visual perception and their content. To perform this clustering we decided to use the median μ and standard deviation σ of the Laplacian and each channel from the HSV color space as they represent: the edges, the saturation, the lightness and the palette of color being used, respectively. To this 8 features (calculated for each image), a principal component analysis (PCA) [17] was applied to represent the feature space in 2 dimensions. The clustering method used was the known as Gaussian Mixture Models (GMM) [18] as the contribution of each of the points to the clustering of the images makes it a better option than other clustering methods, in our opinion. In Figure 7 we present the clustering of the paintings in the database where each cluster is represented with one color.

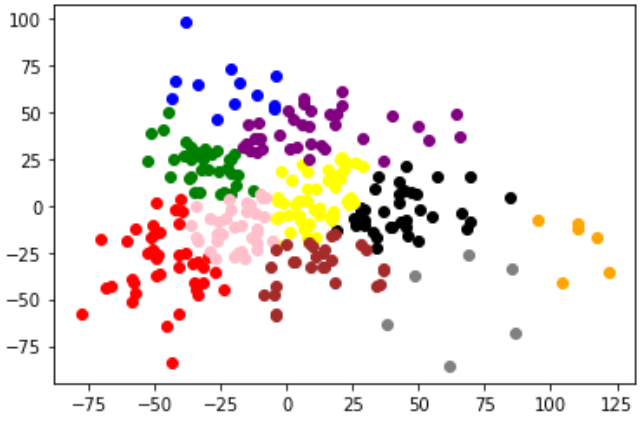


Fig. 7. Plot of the data points in the feature space after performing with Gaussian mixture models after 10000 iteration.

C. Limitations

The proposed retrieval model presents the following limitations:

1) The model highly depends on the background removal and cropping of the pictures, this is the extraction of the pictures from the original photo.
2) Despite we do not need to ”train“ any model, our pipeline requires fine-tuning of multiple hyper-parameters such as: background removal morphological operations, keypoints filters, number of blocks and histogram dimension. Most of these parameters are set empirically after trial-error, to find the optimal parameters is very time-consuming.
3) Combine multiple descriptors might increase the inference time (i.e. the time the system needs to process and images and retrieve the top-k from the database), and therefore, the pipeline would not be suitable for real-time applications.

IV. CONCLUSIONS

Classical Image Processing formulations allow us to solve Computer Vision problems without Machine Learning limitations (*e.g.* huge amount of labeled data, design and select a model, train complex models). The proposed Image Retrieval pipeline allows to retrieve the closest images from our database, for a given input photo captured “in the wild” in a Museum. Moreover, the pipeline is robust against illumination changes, noise, rotations and camera pose, this is because we use SIFT invariant keypoints as our main descriptor of the image. We will explore other techniques such as Query Expansion or Database-side Augmentation (DBA) to improve the retrieval performance without modifying the descriptors.

REFERENCES

[1] O. Chum, A. Mikulík, M. Perdoch, and J. Matas, "Total recall ii: Query expansion revisited," in *CVPR 2011*, 2011, pp. 889–896.

[2] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," 2018.

[3] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," 2016.

[4] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," 2018.

[5] A. Gordo, F. Radenovic, and T. Berg, "Attention-based query expansion learning," 2020.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 886–893 vol. 1.

[7] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1150–1157 vol.2.

[8] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 404–417.

[9] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, 2018, pp. 1–10.

[10] V. I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb. 1966.

[11] K. Sreedhar, "Enhancement of images using morphological transformations," *International Journal of Computer Science and Information Technology*, vol. 4, no. 1, p. 33–50, Feb 2012. [Online]. Available: http://dx.doi.org/10.5121/ijcsit.2012.4103

[12] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[13] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, p. 11–15, jan 1972. [Online]. Available: https://doi.org/10.1145/361237.361242

[14] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, 2005, pp. 60–65 vol. 2.

[15] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[16] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, 1974.

[17] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[18] D. A. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometrics*, 2009.