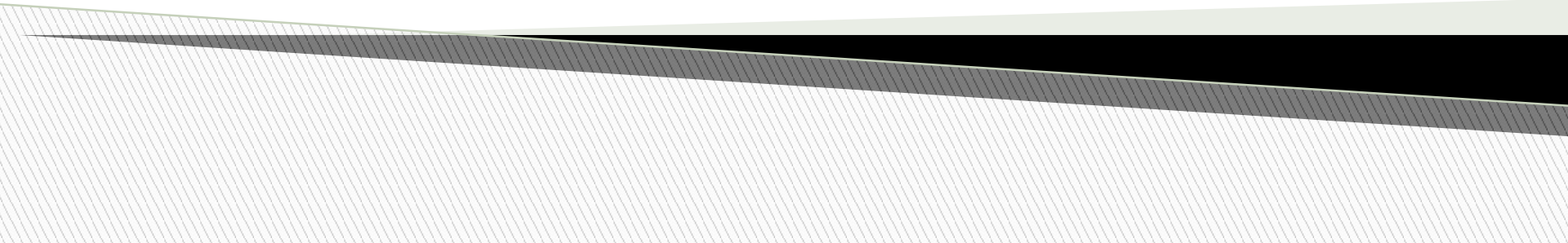


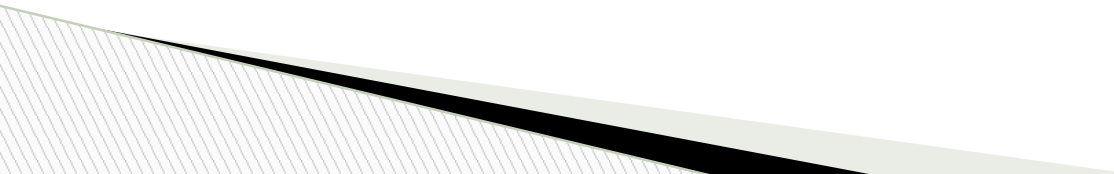
Week 3

Museum Painting Retrieval

Group 8: José Manuel López Camuñas, Marcos V. Conde, Alex Martin Martinez



INDEX

1. Image Denoising
 2. Text OCR
 3. Texture Descriptors
 4. Results
 5. Discussion and conclusions
- 

W3 - Task 1: Image Denoising

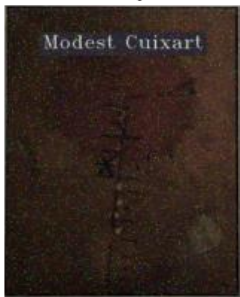
- We use **QSD1-W3** dataset, which consists of cropped noisy and noise-free images.
- **Image Denoising** sometimes leads to images' over-smoothing and the lost of textures and details, we consider this qualitative aspects besides the quantitative performance.
- We use the following techniques:
 - Linear: Low-pass filters, Gaussian filters, Average filter
 - Non-linear: Median filtering
 - and the Non-Local Means Denoising algorithm [1].
- We use cv2 implementations and default 3x3 filter size.

[1] Antoni Buades, Bartomeu Coll, Jean-Michel Morel , Non-Local Means Denoising, Image Processing On Line, 1 (2011),

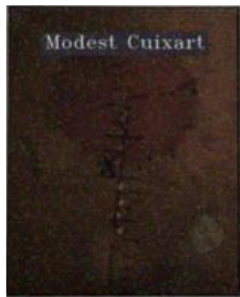
W3 - Task 1: Image Denoising Results

Qualitative Samples from the algorithms:

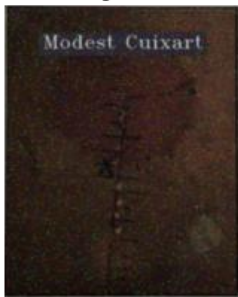
Noisy



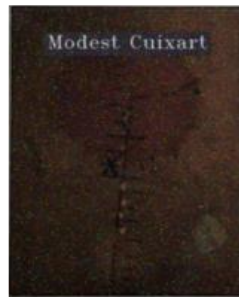
Low Pass



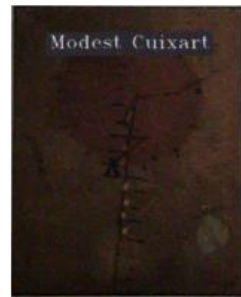
Avg. Filter



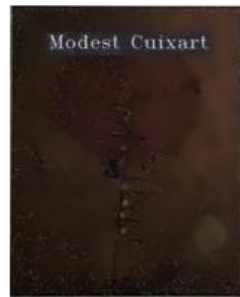
Gauss filter



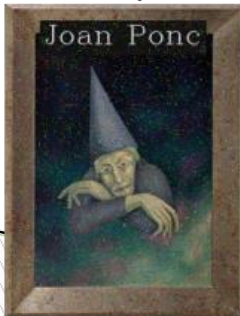
Median Filter



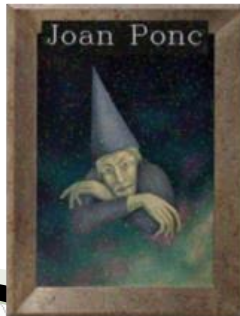
NL-Means



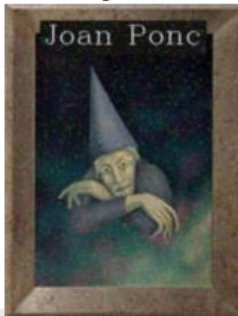
Noisy



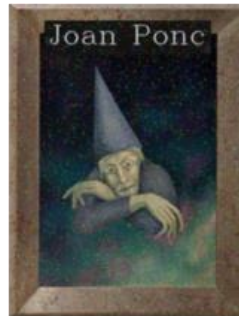
Low Pass



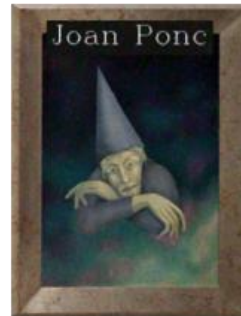
Avg. Filter



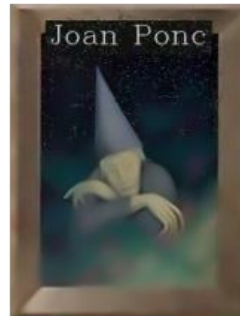
Gauss filter



Median Filter



NL-Means



W3 - Task 1: Image Denoising Results

Noisy



Low Pass



Avg. Filter



Gauss filter



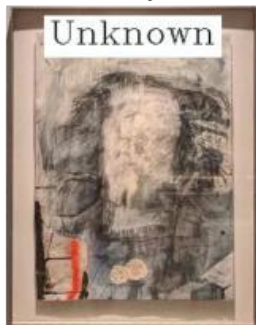
Median Filter



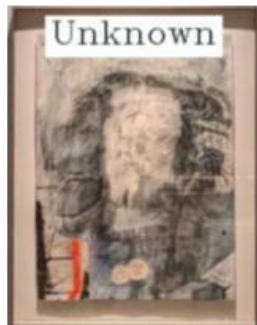
NL-Means



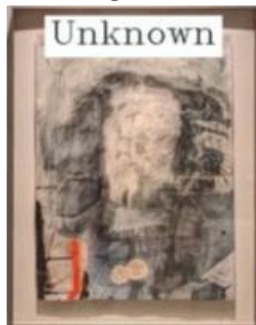
Noisy



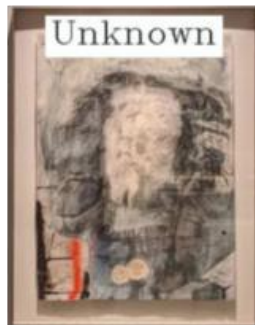
Low Pass



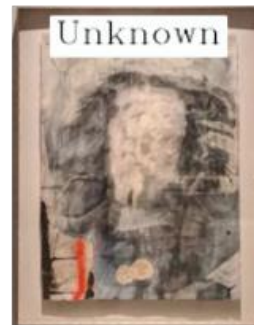
Avg. Filter



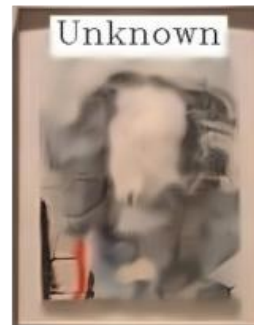
Gauss filter



Median Filter



NL-Means



W3 - Task 1: Image Denoising Results

Algorithm	QSD1-W3 MAP@10 Color Descriptor	QSD1-W3 MAP@10 Texture Descriptor
Base Noisy image	0.85 (25/30)	0.654 (19/30)
Low-pass Filter	0.867 (26/30)	0.775 (23/30)
Average Filter	0.867	0.775 (23/30)
Gaussian Filter	0.867	0.743 (22/30)
Median Filter	0.867	0.829 (24/30)
NL-Means	0.85	0.903 (27/30)

In **red** color the best result, **blue** second best result. We use best color descriptor from Week 2 (Multiblock 3D Hist.)

W3 - Task 1: Image Denoising - NL Means

This algorithm extends the idea of “local mean” mean filters, or simple average filters. Instead of convolving the image, and take the mean value of the group of pixels surrounding the target pixel, this algorithm assumes that the most similar pixels to a given pixel have no reason to be close at all, therefore, the mean of the neighbour pixels is not always accurate.

The authors propose to scan more portions (patches) of the image in search of all the pixels that really resemble the pixel one wants to denoise.

B. Non-local Means [2-3]

$$u_i^{NL} = \sum_{j \in \Omega} \omega(i, j) g(j) \quad (1)$$

These weights are define as,

$$\omega(i, j) = \frac{1}{Z(i)} e^{-\frac{\|g(N_i) - g(N_j)\|_2^2}{h^2}}, \quad (2)$$

where $Z(i)$ is the normalizing constant

$$Z(i) = \sum_j e^{-\frac{\|g(N_i) - g(N_j)\|_2^2}{h^2}} \quad (3)$$



Fig. Scheme of NL-means strategy. Similar pixel neighborhoods give a large weight, $w(p,q1)$ and $w(p,q2)$, while much different neighborhoods give a small weight $w(p,q3)$.

[2] Buades, Antoni, Bartomeu Coll, and J-M. Morel. "A non-local algorithm for image denoising." *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE, 2005. (cited by 1792)

[3] Buades, Antoni, Bartomeu Coll, and Jean-Michel Morel. "A review of image denoising algorithms, with a new one." *Multiscale Modeling & Simulation* 4.2 (2005): 490-530. (cited by 1916)

W3 - Task 1: Image Denoising

NL Means Denoising sometimes produces blurry regions (too smooth) and we lose high-frequencies details and textures in such regions.

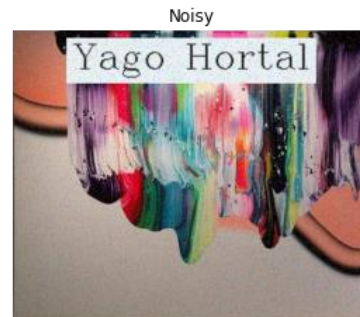
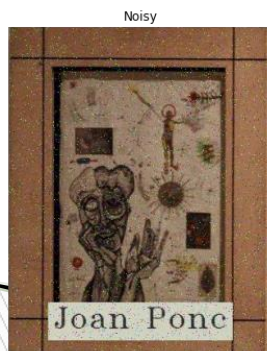
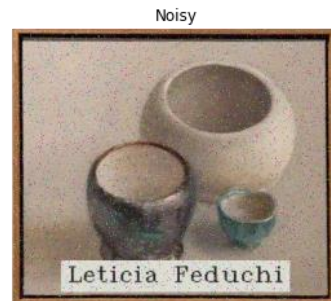
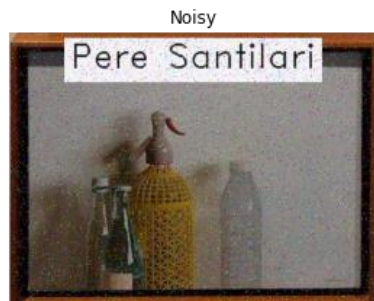
Overall quantitative and qualitative results are the best.

Finding the best hyperparameters is really time-consuming, we show results using the recommended setup. **Median filtering** is the alternative.



W3 - Task 1: Image Denoising

NL Means Denoising qualitative results:



W3 - Task 2: Text OCR

To obtain a region of interest to apply the OCR we used the detection of the text bounding box of Week 2.

After the combination of morphological operators, the pytesseract OCR function was applied on the detected box, obtaining the detected string.

Finally, we postprocess the string to remove any character different to letters numbers or spaces (e.g. special characters).

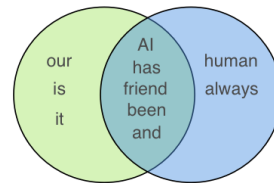


W3 - Task 2: Text OCR - Jaccard Similarity

The word-level Jaccard Similarity is defined as the size of the intersection divided by size of the union of 2 sets.

For example, we can consider the following sentences:

- Sentence 1: “AI is our friend and it has been friendly”.
- Sentence 2: “AI and humans have always been friendly”.



We split the sentences into a list of words and calculate the metric.

Usually lemmatization and other techniques to unify text are used, but we do not employ them.

The result Jaccard similarity for those 2 sentences is $5/(5+3+2) = 0.5$

W3 - Task 2: Text OCR - Jaccard Similarity

We provide the implementation details of the word-level Jaccard score for string similarity:

```
def jaccard(str1, str2):  
    a = set(str1.lower().split())  
    b = set(str2.lower().split())  
    c = a.intersection(b)  
    return float(len(c)) / (len(a) + len(b) - len(c))
```

$$score = \frac{1}{n} \sum_{i=1}^n jaccard(gt_i, dt_i)$$

n = number of documents

$jaccard$ = the function provided above

gt_i = the i th ground truth

dt_i = the i th prediction

W3 - Task 2: Text OCR

To compare the strings detected with the titles in the database two different similarity indicators were used.

- **Levenshtein (method1)** which is simple, works better with short strings as it doesn't take into account the semantics
- **Jaccar (method2)** which is token based it's more robust for computing text similarities as it takes semantics into account.

The MAP@K for the methods with the QSD1-W2:

	QSD1-W2 MAPK@10
Method 1	67%
Method 2	69.94%

W3 - Task 2: Text OCR

- As we can observe although Levenshtein being more suitable for short string jaccard had more precision in the text retrieval.
- The main problem of this method is that its performance highly depend on the produced bounding boxes, if they are too big (or not accurate), the OCR cannot generate a good string from the image.



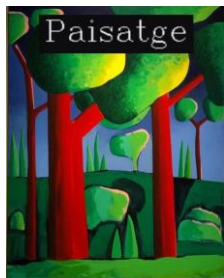
Generated string with
the whole image:
"Bc.)dlesne7)J l\l"



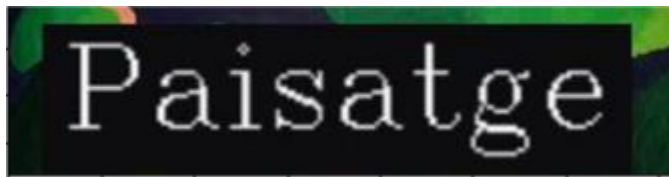
OCR generates no string
as all it detects are
special characters that
are removed afterwards.

W3 - Task 2: Text OCR

Painting

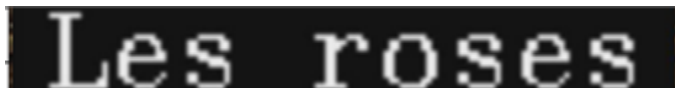


Detected box



Extracted string after
cleaning special characters
with re.sub() function

Paisatge



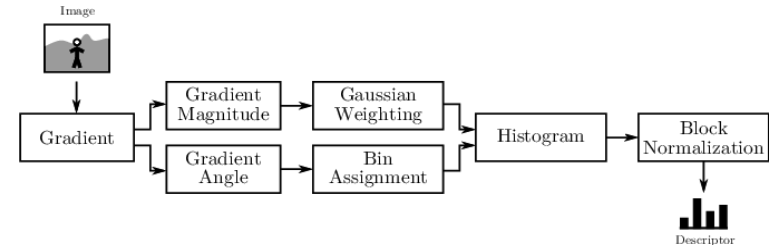
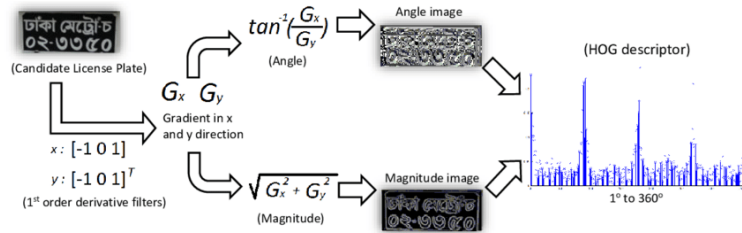
Les roses

W3 - Task 3: Texture Descriptors

- We use **QSD1-W2, QSD2-W2, QSD1-W3, QSD2-W3** datasets.
- **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 better.
- We use the following methods:
 - HOG (Histogram of Oriented Gradients)
 - LBP (Local Binary Pattern)
- We use skimage implementations.

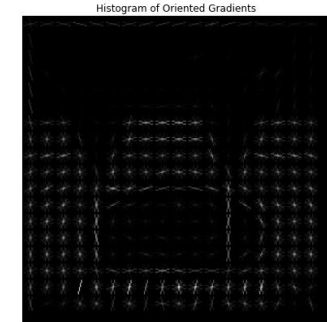
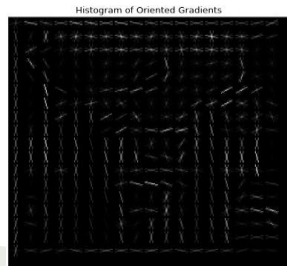
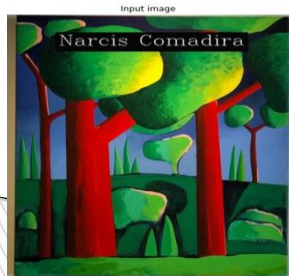
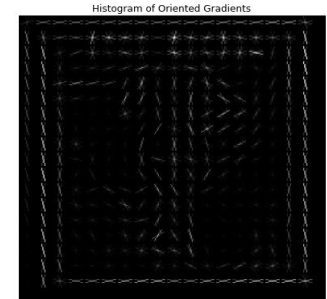
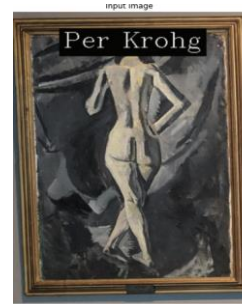
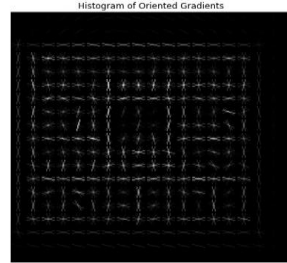
W3 - Task 3: Texture Descriptors

- HOG (Histogram of Oriented Gradients):
 - Counts occurrences of gradient orientation in the localized portion of an image.
 - Focuses on the structure or the shape of an object.
 - Generates histograms for the regions of the image using the magnitude and orientations of the gradient.
 - Generates a 300-dimensional descriptor.



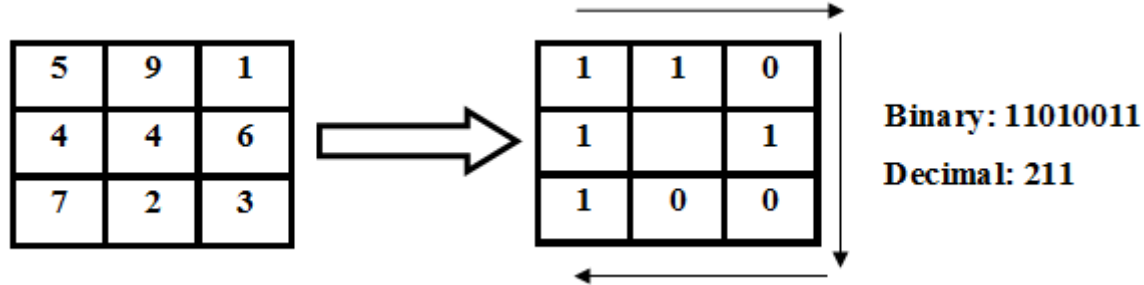
W3 - Task 3: Texture Descriptors

- HOG (Histogram of Oriented Gradients):



W3 - Task 3: Texture Descriptors

- LBP (Local Binary Pattern):
 - Divide image into blocks, for each pixel in the block compare to its 8 neighbors.
 - If the center pixel's value is greater than the neighbor's value, write 0, otherwise, write 1. The result is a 8-digit binary number. Compute histogram, over the block, of the frequency of each number.



W3 - Task 3: Texture Descriptors

- We select HOG (Histogram of Oriented Gradients) as our texture descriptor, overall results are better than the using only color information or LBP.

	QSD1-W3 MAP@10	QSD1-W3 MAP@5	QSD1-W3 MAP@1
Color Descriptor (Multi-Block 3D Hist)	0.85 (25/30)	0.85	0.833
Texture Descriptor (HOG)	0.877	0.867	0.833
Texture Descriptor (LBP)	0.62 (18/30)	0.561 (16/30)	0.267 (8/30)

W3 - Task 4: Descriptors Analysis QSD1-W3

- Each descriptor has different dimensionality and represent different spaces, therefore, we do not concatenate them.
- The Text descriptor can also be obtained by producing a N-dimensional embedding from the words, however, we did not find it useful.
- We **combine the similarity scores** in terms of “*Hellinger Distance*” from our Texture Descriptor (HOG) and Color Descriptor (Multi-Block 3D Histogram). The Texture (T_score) and Color (C_score) scores are combined into the final score (S) that we use for ranking the top-k using the equation below:

$$S = \alpha * T_{score} + \beta * C_{score} \quad \text{where} \quad \alpha, \beta > 0$$

- We do not use Text Descriptor, in our pipeline results did not improve by adding their score (Levenshtein score) into the equation above. Moreover, many images do not have text descriptor.

W3 - Task 4: Descriptors Analysis QSD1-W3

We provide results using different combinations of descriptors. Note that images were denoised as explained before.

	QSD1-W3 MAP@10	QSD1-W3 MAP@5	QSD1-W3 MAP@1
Color Descriptor (Multi-Block 3D Hist)	0.85 (25/30)	0.85	0.833
Texture Descriptor (HOG)	0.877	0.867	0.833
Text Descriptor (Levenstein)	0.4537(13/30)	0.43889	0.3
Color + Texture (alpha=0.5, beta=0.5)	0.9237	0.916	0.9
Color + Texture (alpha=0.3, beta=0.7)	0.9583	0.9583	0.933

W3 - Task 5: Descriptors Analysis QSD2-W3

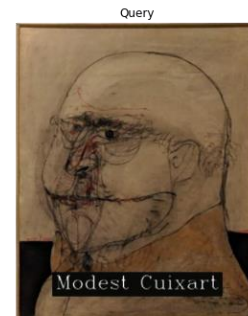
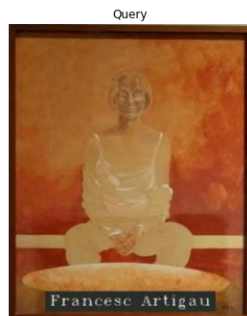
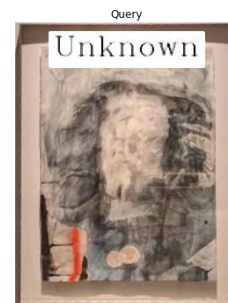
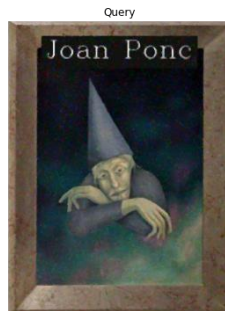
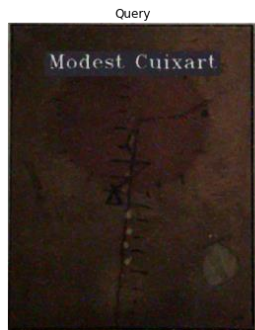
- We provide results using different combinations of descriptors. Note that images were not denoised, as specified in the slides.

	QSD2-W3 MAP@10	QSD2-W3 MAP@5	QSD2-W3 MAP@1
Color Descriptor	0.3083(9/30)	0.3083(9/30)	0.3667(11/30)
Texture Descriptor	0.35(10/30)	0.3667(11/30)	0.3667(11/30)
Color + Texture (alpha=0.5, beta=0.5)	0.55 (16/30)	0.55 (16/30)	0.567 (17/30)
Color + Texture (alpha=0.3, beta=0.7)	0.6000 (18/30)	0.6000 (18/30)	0.6333 (19/30)

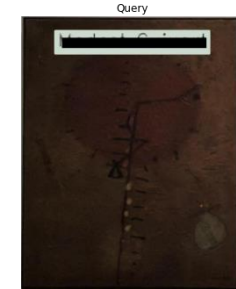
W3 - Task 5: More Results

QSD1 W2 - Text only			QSD1 W2 - Color only			QSD1 W2 - Texture only		
Team	Method	map@1	Team	Method	map@1	Team	Method	map@1
Team1	method1	0.4	Team1	method1	0.6	Team1	method1	1
Team1	method2		Team1	method2		Team1	method2	
Team2	method1	0.3	Team2	method1	0.6	Team2	method1	0.6
Team2	method2	0.3	Team2	method2	0.5667	Team2	method2	0.5
Team3	method1		Team3	method1		Team3	method1	
Team3	method2		Team3	method2		Team3	method2	
Team4	method1	0.7666	Team4	method1	0.5667	Team4	method1	
Team4	method2	0.7666	Team4	method2	0.5334	Team4	method2	0.6
Team5	method1	0.7583	Team5	method1	0.800	Team5	method1	0,667
Team5	method2	0.7583	Team5	method2	0,733	Team5	method2	0,5833
Team6	method1	0.19	Team6	method1	0.667	Team6	method1	1
Team6	method2	0.19	Team6	method2	0.7	Team6	method2	0.933
Team7	method1	0,05	Team7	method1	0,7	Team7	method1	0,9
Team7	method2	0,05	Team7	method2	0,7	Team7	method2	0,9
Team8	method1	0.67	Team8	method1	0.667	Team8	method1	1
Team8	method2	0.67	Team8	method2	0.667	Team8	method2	0.95

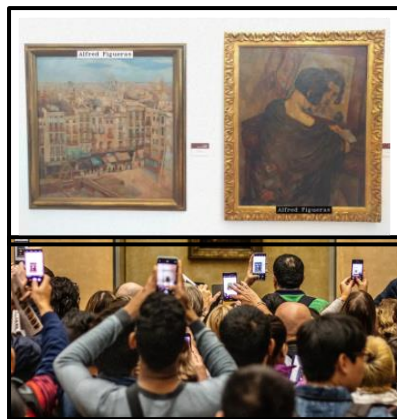
W3 - Task 5: QSD1-W3 Qualitative Results



W3 - Task 5: QST1-W3 Qualitative Results



W3 - Final Retrieval Pipeline



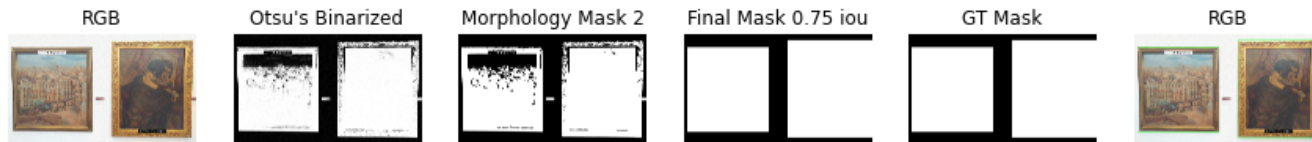
Input Image (Query)



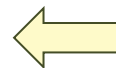
1) Preprocessing: 1st level Denoising using NLMeans



2) Morphological Transformations, background removal + cropping



3) crop-level Median Filtering + Text removal



4) Feature Extraction + Similarity Search



Retrieval from DB

W3 - Conclusions

1. Denoising can change the smoothness, details and textures of the image, it does not modify color information too much, for this reason, color descriptors do not change, but texture's ones are affected.
2. Extract “contextual” information from the image like texts using OCR methods and morphological detectors leads to a more complete and versatile retrieval system (multimodal data powered).
3. We find the texture descriptors more powerful than color ones, we assume this is because the edges and high-frequency details that characterize the painting are encoded in such descriptors.
4. The combination of these descriptors allow us to rank better the BD images related to a query image, and thus, boost our retrieval score.