Detection, Rectification and Localization in a Museum Environment

Aniello Panariello - 140195, Fabrizio Sorgente - 133855, and Emanuele Fenocchi - 148869

University of Modena and Reggio Emilia

June 9, 2020

1 Introduction

The aim of this work is to detect paintings and people inside a museum environment and then perform retrieval and rectification of the detected painting from a database of high quality images, we also detect statues. The detection of the three objects is performed with a custom trained YOLOv3 network, while the retrieval is done by ORB keypoints. For the rectification we exploit the keypoints obtained by the ORB to find the homography matrix. Once we have found the paintings and the people we can localize the latter by getting the localization of the painting. The direction in which the person is facing is computed by a face detection and assuming that if the person is not looking at the camera then he is facing a painting. We also process screenshots from a 3D model of the museum, replacing paintings with high quality images from a database, using an inverse approach w.r.t the rectification one.

2 Related Works

To perform the tasks of this work, we exploited some previous work in this field. We used YOLOv3 neural network [1] for the detection trained with darknet [2]. We also compare another pipeline, without neural network, in which we preprocess the image with adaptive threshold, median blur [3] and opening, to then apply the Connected Component Labeling (BBDT) [4]. Retrieval uses ORB [5] with Lowe's ratio test [6]. In the last part we detect faces with the Haar Cascade classifiers [7].

3 Approach

In this section we will describe the main steps of the pipeline with their evaluation.

3.1 Detection

The detection of paintings, statues and people is done through a custom YOLOv3 neural network [1]. Yolo is an architecture that provides a new approach to object detection, in which a single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation.

In order to use this neural network we have labeled 1343 images, 805 of them have been used to train the network, 269 for the validation set and 269 for the test set, providing a total number of 3733 labels. The images of paintings and statues were taken from frames extracted by some videos recorded in the Galleria Estense, for the person class was instead used a mix of images taken from the previous videos

and some images of people in another museum to increase generalization. Fig. 1 shows the graph for the training that has been performed with darknet [2] in which we took the mAP every 1000 iterations in order to have the best weights that weren't affected by overfitting.

In the end we got a powerful neural network with high performance and capable of a good generalization, a result of the detection can be seen in fig. 2 while the performance is shown in tab. 1.

An important point is the data management: the train set, the validation set and the test set have been balanced with a 60/20/20 split, this allowed the network to have a better performance and to predict classes in a correct way.

3.1.1 Comparison with previous technique

In a previous pipeline (showed in fig. 11) the detection was made without neural networks: the image was at first transformed in a gray level image, preprocessed with adaptive threshold, median blur [3] to remove the noise

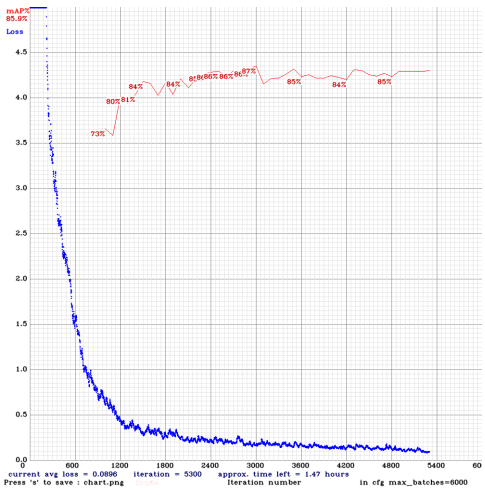


Figure 1: YOLOv3 trained on our custom dataset.

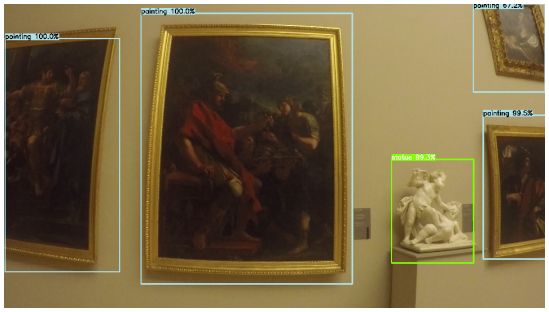


Figure 2: Detection with YOLOv3.

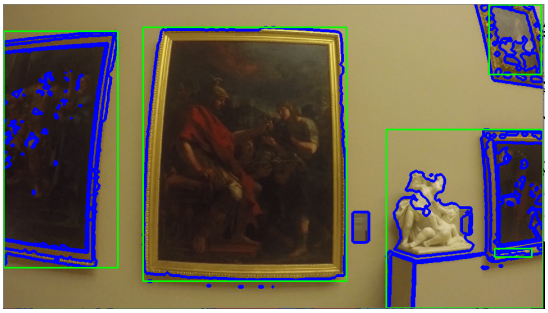
and opening, and then we computed the borders. After the preprocessing, the pipeline uses the result to compute the Connected Component Labeling [4] to label the background and the foreground objects. In order to label the foreground objects as paintings, the components needed to have an entropy grater than a threshold, and then the bounding boxes were drawn.

Despite this method worked well in some scenarios, it wasn't able to adapt to strong luminance variation, to manage the presence of shadows and to generalize with all the classes. As we can see in fig 3, the two big paintings are correctly detected, while the statue and the lower right painting are merged in a single bounding box since their borders overlap. Many times the shadow was recognized as part of the painting and for this reason the IoU was not precise enough, this led us to choose the neural network approach.

3.2 Painting Retrieval

Painting retrieval uses ORB [5] keypoints detector and descriptor to find matches between two images. ORB, is at two orders of magnitude faster than the old used SIFT [6] and this is the main reason why we have chosen to use this method, in order to achieve the painting retrieval with a good performance/results ratio, other than the fact that SIFT has been

Image with shadow



Not precise bounding box

Figure 3: Inaccurate detection.

Detection Performance				
	Painting	**Statue**	**Person**	**Overall**
TP	550	174	125	849
FP	107	19	32	158
Precision	83,71%	90,16%	79,62%	84,31%
Recall	-	-	-	94%
Average IoU	-	-	-	71%
AP	97,29%	98,61%	75,45%	-
mAP	-	-	-	90,45%

Table 1: Detection performance with YOLOv3.

patented in the last versions of OpenCV. An example of retrieval is shown in fig. 4.

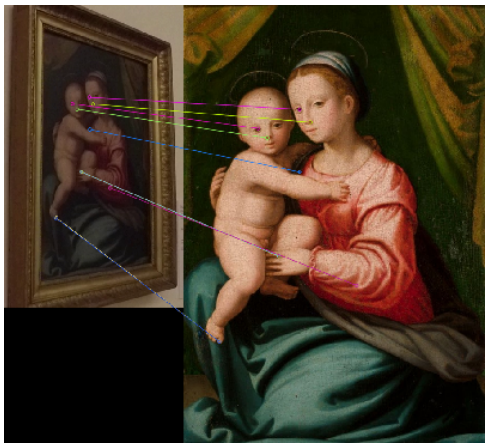


Figure 4: Example of painting retrieval.

3.2.1 Improvements

To improve the retrieval and to reduce false matches, we used the ratio test described in Lowe's paper [6], in order to get the best matches. Given the two best matches (the best match and the second best match) for a keypoint, we define a *threshold* and if the ratio between the distance of the two best matches, respectively d_1 and d_2, is above that threshold, $\frac{d_1}{d_2} > threshold$, we reject that keypoint, considering it equivalent to noise and because the best match is not so different from the second one.

Our goal was to keep the number of paintings correctly found in the database high but at the same time to not decrease the number of *true negatives* for the paintings that are

missing from the database. We have chosen $threshold = 0.6$ because increasing it even by just 0.1, despite the decrease in the number of false positive retrievals for paintings not listed in the database, this resulted in an increasing number of wrong matches for the paintings in the database. Decreasing the threshold led to an opposite situation and both of the cases didn't meet our goal.

Computing the same keypoints and descriptors for the same paintings in the database has resulted in a slow start, causing the retrieval to wait a couple of seconds or more. Computing the keypoints and descriptors offline and loading them at retrieval time, reduced the retrieval initialization by more than 40%.

3.2.2 Evaluation

To evaluate the retrieval, we tested it with a sample of 3 random frames per video, with a total of 100 randomly selected videos out of 208. Some of these frames either did not contain any frames or the region of interest of the paintings exceeded the frame size, therefore 100 frames out of the total of 300 were discarded. 266 are the total paintings detected using our trained model and 45 of these were discarded because they were unrecognizable, due to their small size and/or their luminance. For the remaining 221 paintings, we manually counted every time we saw a wrong or a right answer. More precisely, we checked if the painting was in the database and the correctness of the retrieval answer, building the matrix in table 2.

The result is that 58 out of 221 paintings were in the database and they were correctly retrieved from it, 56 out of 221 were not in the database but the retrieval correctly gave us a *no match found*. The remaining paintings were divided in wrongly retrieved from the database, and the ones that had not an instance in the database but a wrong match was uncorrectly found.

		Retrieval answer	
Paintings		**Found**	**Wrong or not found**
	In DB	58	60
	Not in DB	47	56

Table 2: Painting retrieval evaluation results.

Accuracy is the measurement used to get information on how good our configuration is:

$$Accuracy = \frac{58 + 56}{221} \approx 0.52$$

We think that this result is pretty good, based on the fact that we could increase the real matched paintings, reducing the uncorrectly found paintings that were not actually in the database, adding them manually to it. Moreover this configuration was the one that optimized the next steps in our pipeline.

3.3 Painting Rectification

We manage to perform a very accurate rectification when the painting retrieval is able to correctly fetch the painting entry from our database. In this case, not only we have the correct correspondence of paintings, but we can also exploit the descriptors obtained from ORB to compute an extremely accurate homography matrix, thanks to the large number of keypoints, to then compute the projective transformation. We see an example in fig. 5.

When the retrieval fails, for the lack of descriptors or when the painting is missing from the database, we use Harris Corner Detector [8] to find the corners of the painting and use them as source points for our homography matrix. This method has a worse performance than the descriptors method, since the corners are not always precise.

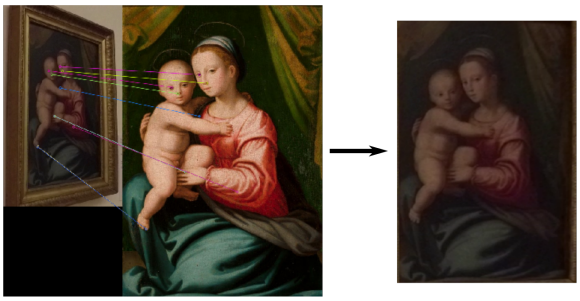


Figure 5: example of painting rectification

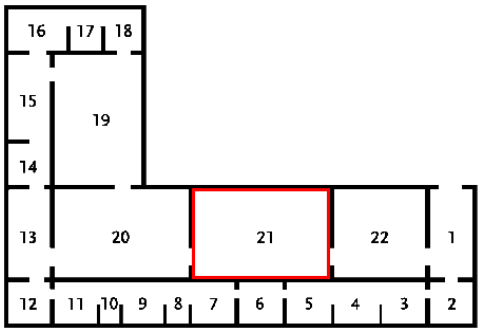


Figure 7: Person localization.

3.4 People Localization

Our approach to achieve this task is based on the quality of the detection and painting retrieval. In fact, if our trained model correctly detects a person, in order to localize that person, we use information about the paintings detected by the model and retrieved from the database.

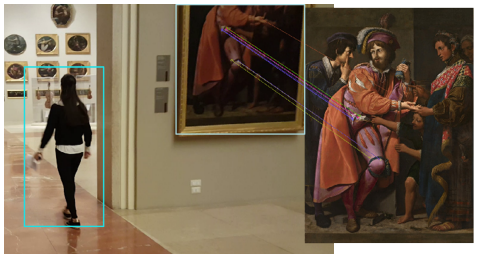


Figure 6: Using detection and retrieval to localize a person.

If our trained model detects a person and there is at least one painting, we retrieve that painting from the database, like the example in fig. 6, and localize the person searching for an instance in the *data.csv*, that gives us the room localization as well as the painting info. When we have correctly localized a person, we show the Galleria Estense's map with the correct room highlighted, using a red rectangle, as shown in fig. 7.

3.4.1 Evaluation

We have access to the *paintings_db*, with information about the room where each painting is located, but there are two main problems that worsen our evaluation:

a) The painting retrieval finds a match only in 50% of the cases.

b) We do not take into account the scenario in which the camera and the person are in a room, while a detected painting is in another room, visible through a door.

The case a) is the only reason why our localization is not perfect, since in some instances we cannot find a correspondence. In our evaluation we discarded all cases that match b) and the results are all depending on the painting retrieval, which are shown in table 2.

3.5 Face detection

We used Haar Cascade classifiers [7] proposed by Viola and Jones in order to perform this task. First, we use our trained model to detect a person, then the ROI of that person is passed to the face detector. The face detector performs the algorithm to check if a person is facing a painting, based on these following scenarios:

a) Face found

(i) Eyes found (at least one)

(ii) Eyes not found

b) Face not found

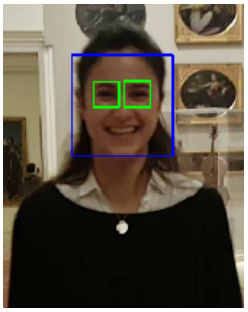


Figure 8: The case in which the face detector correctly found a face from the person ROI.

In case (i), if the face is found with its eyes, we assume that the person is facing the camera therefore he is not facing a painting, like in fig. 8. In case (ii), if the face is found but the eyes

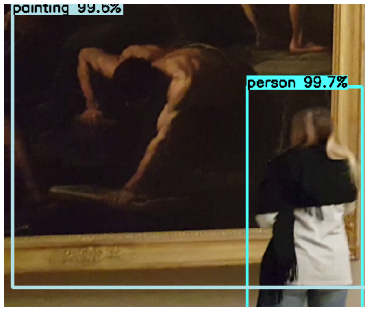


Figure 9: The case where the face detector can't find the face.

are not detected, we assume that the person is facing a painting, this case is a particular scenario where the person could be in profile. The case b) has the same assumptions of the case (ii) because if the face is not detected, this could mean that the person is turning his back to the camera and, therefore, is possibly facing a painting, like in fig 9. In these last scenarios, we take into account the paintings ROIs and we check if the person ROI overlaps a painting ROI, in this case the person is in front of a painting.

3.5.1 Evaluation

The assumptions we've made, have allowed us to model most of the possible cases, but since the videos in which there is a clearly visible person are only few, the test evaluation may not represent the real accuracy of our approach. As a result, the test has been done using 2 frames per second for each videos where there is at least one person for more than 3 seconds, with a total of 8 videos. 466 are the total frames used and only 121 are the optimal candidates for the test, where a person is clearly visible, and was not too far from the camera. Since the face detector takes in input the person and paintings ROIs, our test is also affected by the quality of our trained model, therefore, not detecting some of the people and paintings, led us to discard another 42 frames.

After all of these rejections, we analyzed a total of 84 frames with 94 people inside it. The evaluation results are shown in table 3 and gave us an accuracy of:

$$Accuracy = \frac{14 + 57}{94} \approx 0.85$$

		Facing painting	
		True	**False**
Face detector answer	**True**	14	10
	False	13	57

Table 3: Facing painting results.

3.6 Painting rectification of the 3D model

For the painting rectification of the 3D model, we manage to replace the low quality painting images with a rectified version of the same paintings in higher quality, retrieved from the

database. Given a screenshot from the 3D model, it goes through the first part of our pipeline until the correspondent painting is found. The retrieved painting is subject to a projective transformation, with an inverse process w.r.t our main pipeline, in order for it to have the same projection as the painting in the screenshot. It is, then, replaced in the screenshot with the use of a mask. If the painting is not found, the warping is not performed. An example can be seen in fig. 10.

Original 3D model painting Replaced painting

Figure 10: Painting rectification from the 3D view.

4 Results

In this work we were able to train a neural network with high performance to detect paintings, people and statues with a mAP of 90.45%. This is a great result for a custom network and a not so large dataset. The neural network detection lays the foundation for all the subsequent steps of the pipeline, having a good performance in the first step is the key to have a good performance also in other steps.

The painting retrieval is our bottleneck since it is able to correctly retrieve paintings just in the 52% of the cases, this is also due to the fact that not every painting is in the database and the bad luminance and projective condition of some frames.

Regarding the painting rectification and people localization, these performance are strictly tied to the previous step since, when the retrieval is correct, this step works without flaws.

The face detection, which is performed in order to determine if a person is facing a painting or not, exploits the detection of a person, this means that it has an AP of 75.45% as we can see in tab. 1. On these detections we are able to verify if a person is facing a painting with an accuracy of 85%, as we can see in tab. 3, using a classic computer vision algorithm without the use of neural networks.

In the last step, the rectification of paintings of the 3D model is performed, using the same pipeline as before with a change in the rectification step. The performance of this step are the same as the rectification in 2D frames.

5 Discussion

In this project we compared the performance of classic computer vision algorithms and neural networks. The neural network we have trained, greatly outperforms our implementation of CCL [4] or thresholding algorithms. While these algorithms could have been optimized with some parameters tweaking, the deep learning approach is faster to implement, once a dataset is available, and is able to generalize much better.

In all the steps, except the detection step, classical algorithms were used, such as descriptors, projective transformation, face detector and so on. Not using neural networks in these steps, allowed us to have better computing performance and a faster pipeline, this would not

have been possible if we only used deep learning.

Moreover, in steps such as painting retrieval, the use of a neural network would have been impractical, since it would have been impossible to obtain an exhaustive dataset of paintings, we only had one image per painting.

In conclusion, the combined use of both classical and deep learning algorithms is inevitable to achieve optimal performance and to solve some tasks that otherwise would have been impossible.

Bibliography

[1] Joseph Redmon and Ali Farhadi. *YOLOv3: An Incremental Improvement*. 2018. arXiv: 1804.02767 [cs.CV].

[2] Joseph Redmon. *Darknet: Open Source Neural Networks in C*. http://pjreddie.com/darknet/. 2013–2016.

[3] T. Huang, G. Yang, and G. Tang. "A fast two-dimensional median filtering algorithm". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 27.1 (1979), pp. 13–18.

[4] Costantino Grana, Daniele Borghesani, and Rita Cucchiara. "Optimized Block-Based Connected Components Labeling With Decision Trees". In: *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* 19 (Mar. 2010), pp. 1596–609. DOI: 10.1109/TIP.2010.2044963.

[5] Ethan Rublee et al. "ORB: an efficient alternative to SIFT or SURF". In: *Proceedings of the IEEE International Conference on Computer Vision* (Nov. 2011), pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[6] David Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60 (Nov. 2004), pp. 91–. DOI: 10.1023/B:VISI.0000029664.99615.94.

[7] Paul Viola and Michael Jones. "Rapid Object Detection using a Boosted Cascade of Simple Features". In: *IEEE Conf Comput Vis Pattern Recognit* 1 (Feb. 2001), pp. I–511. DOI: 10.1109/CVPR.2001.990517.

[8] Christopher G Harris, Mike Stephens, et al. "A combined corner and edge detector." In: 15.50 (1988), pp. 10–5244.

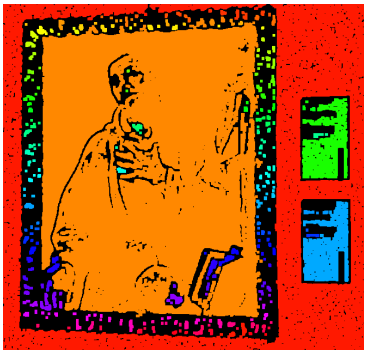
Additional Resources

Frame from video

Adaptive threshold

Median blur

Opening



Connected Component Labeling

Painting Detection

Figure 11: Detection pipeline without neural network.