Computer vision
Recognition based localization

Baert Olivier, Callaert Arne, Hernou Arno, Vereecken Ward, Waegemans Wolfgang

February 9, 2022

Abstract

In this project, a program was created which attempts to localize a person in the Museum of Fine Arts in Ghent through videos taken by this person. To achieve this, said video footage is processed frame by frame. Potential paintings are detected and extracted from each frame, which are then fed to a support vector machine classifier. Confirmed paintings are matched against a preinitialized database of known paintings using SIFT feature extraction. Calculated matching scores are then passed to a Hidden Markov Model, which then statistically determines the person's location in the museum. Painting detection performs adequately well on testing data sets, with 740 out of 801 paintings detected from the testing set, of which 103 were false positives. Matching also performed well, with a correct match percentage of 92.09%. Finally, the Hidden Markov Model processed these matching scores. The correct room does not always receive the highest score, but it is always guessed in the top five rooms.

1 Introduction

This paper presents an approach to localization of people inside a museum based on the artworks present. To achieve this, the paper was divided in four sections. The first section describes the extraction of paintings inside a single video frame using a combination of, among others, Otsu's thresholding method and Canny edge detection. By applying both methods, an intermediate result containing only the 'contours' of the input image is created. Followed by a list of logical filters (area analysis, corner count, etc.) applied to all potential polygon candidates, the candidate with the highest potential of being a painting is extracted. To finalize this section, the extracted painting undergoes a perspective transformation resulting in a standardized top view of the painting. The second section of the paper then describes a solution to match the extracted painting against the paintings in a preinitialized database. The key-points (points of interest) of the extracted painting are determined using SIFT and are compared to the key-points of the paintings in the preinitialized database. The found matches are then returned in a sorted order with the best match being the first. To further enhance the localization ability of the algorithm, section two is supported by a Hidden Markov Model, calculating the probabilities of being in a certain room, further described in section three. The final contribution to this paper described in section four starts with an approach to video frame processing and continues by describing how the algorithms in section one to three can be applied to a video fragment, both contributing to the end result of locating people on their way through a museum. This project can potentially become a great asset for the museum. It could be a great replacement for the classic hand-out floor-plan, and might even possibly be used to provide the visitor information about the painting he just has detected in the future. There already exists some image-based localization methods. One of them is 'Structure-from-Motion (SFM)'. Structure-from-Motion is

a prevalent strategy for 3D reconstruction from unordered image collections [1]. Another paper shares mainly the same goal of indoor localization through images. One of the main steps here work by loading extracted image database SIFT features into a kd-tree and perform fast approximate nearest neighbor search to find a database image with the most number of matching features to the query image, and basing the localization on this match [2]. This technique was a big inspiration for this project.

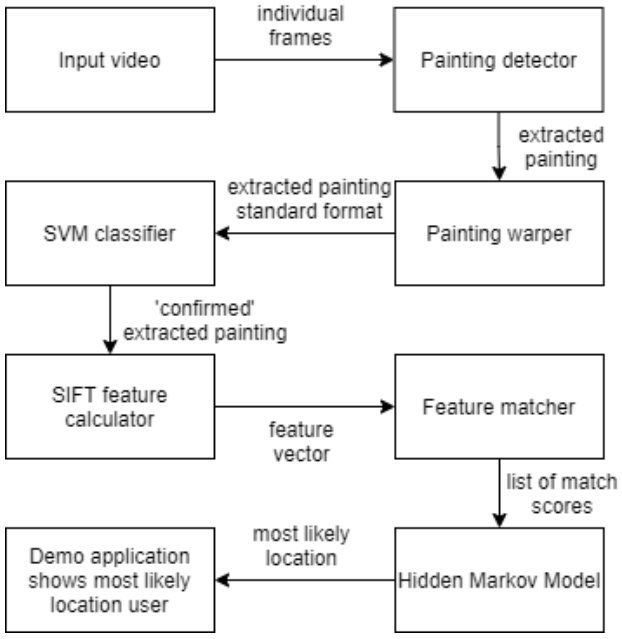
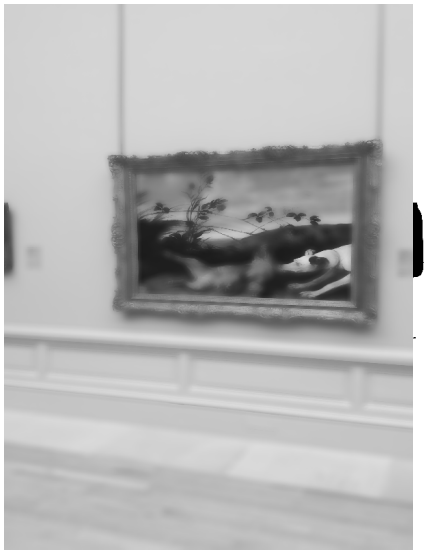


Figure 1: Block diagram showing the flow of the application between the different components.

2.1 Filtering

(a) After gray scaling



(b) After bilateral filter

(c) After Otsu's thresholding

Figure 2: Initial filtering of the image

2 Painting detection

In this Section, the focus lies on detecting any shape in a given image that is most likely to be a painting, completely **unsupervised** (without any human intervention). This problem was solved in multiple steps. Throughout this Section, one of the images from the provided testing images will be used to showcase the utilized methods and their respective results.

First, the image is converted to gray scale and a bilateral filter is applied. A bilateral filter is a non-linear, edge-preserving, and noise-reducing smoothing filter for images. As such, it preserves all pixels with more intense gradient changes and blurs the others. This leads to an image where the edges are still seen, but the rest is blurred out. This can be clearly seen by comparing 2a and 2b, where now all parts of the image are blurred except for the painting's borders, and some edges inside the painting.

Next, Otsu's thresholding is applied to the image. Otsu's thresholding can perform automatic image thresholding. In the simplest form, the algorithm returns a single intensity threshold that separates pixels into two classes: foreground and background. It determines these classes by classifying pixels as part of a class if its intra-class intensity variance is minimized. Thus, in the case of an image with a painting in it, it could divide the painting from the wall. And, as seen in Figure 2c, this is indeed what happens. The entire wall and floor get the value 255, and the painting, a small ridge and a small piece of another painting get the value 0. This is the result of calling cv2.threshold(image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU), where 0 and 255 indicate the new pixels values that are to be given to the respective classes. As a flag, we pass 'cv2.THRESH_BINARY + cv2.THRESH_OTSU', indicating we want to threshold to only two values, with the threshold itself determined by Otsu's method.

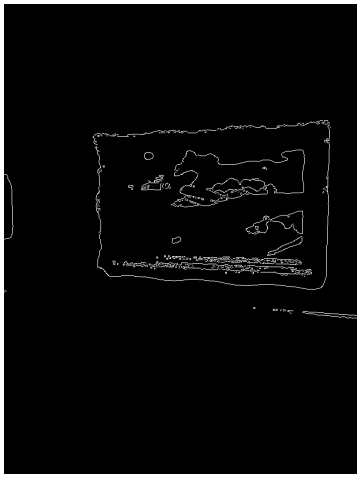
As seen in 2a, this painting (as well as other paintings) hangs on two ropes against the wall. This initially carried problems with itself: without Otsu's thresholding, edge detection on 2b would find an edge around the painting together with the ropes. This is illustrated in Figure 3a. To solve this problem, first, contours were detected in the image, and, after filling these contours, erosion was performed on the image to erode away this thin line. However, thanks to Otsu's thresholding, which classifies the ropes as background pixels, this is no longer necessary. Thus, this will not be explained in more detail.

2.2 Edge detection

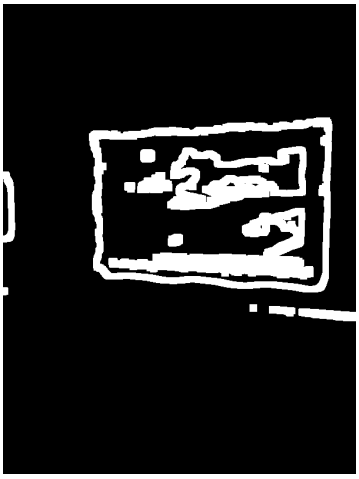
Now, we are ready to detect the painting's edges in the image. We do this using Canny edge detection. Canny edge detection works by using a Sobel to calculate the gradient (the degree of transition, so when a pixel suddenly changes significantly in intensity with its neighbor, the gradient will be higher here) of each pixel in the image [3]. Afterwards, these gradients are compare with the given thresholds:

- pixels with a gradient higher than the highest threshold get accepted as edges,
- pixels with a gradient lower than the lowest threshold get dropped,
- pixels with a gradient between the two thresholds only get accepted if they reside close to an accepted pixel.

(a) Without Otsu's thresholding



(b) After Otsu's thresholding



(c) After dilation

Figure 3: Detected edges visualized

Because, after performing Otsu's thresholding, the gradient between the painting and the wall should be very high, the given thresholds should not matter too much. After performing edge detection on 2c, we acquire the edges of our painting, as seen in Figure 3b.

Now, to identify the painting(s) in the image, we need to find the outline in these edges, or

the **contours**. This happens by looking for big changes in gradient in the image, just like canny edge detection. However, here, the algorithm will not just find edges or borders, but also trace this border along its path, and attempt to find a closed border ([4]).

However, after testing, it was found that sometimes, edges found with Canny edge detection do not always connect. In these cases, contours found would not be correct, as they would not go around the entire painting. Thus, to make sure all edges connect into a single contour around the painting, dilation was performed on Figure 3b. Dilation widens all light-colored areas of an image by replacing each pixel with the maximum of its surrounding pixels. The area of used surrounding pixels can be set as a parameter. In this context, it is best to set it higher than the standard, as it is ideal to connect as much disconnected contours as possible. The result of this dilation can be seen in Figure 3c. Now, edges are almost always fully connected, and contours may be found.

2.3 Actual painting detection

Finally, these contours need to be identified as paintings as well as possible, while also finding the coordinates of the corners correctly. Firstly, out of all the contours found, the one with the highest area is searched for, as this is the most likely to be a painting. Now, each contour could be tested as a painting candidate using e.g. a classifier (see further, Section 2.5). However, this could lead to many useless tests, where a first filtering of the contours might lead to more efficient detection. Instead, a comparison of each contours area against the max area of the image is made. If this are is equal to or higher than 20% of this maximum area, we consider the contour as a candidate. Next, the candidates are further analyzed by calculating the number of corners in them. This is done by this done by approximating the contour to a shape. This is possible by following the Ramer–Douglas–Peucker algorithm, which takes each point along the contour and finds the shape that lies closest to each of these points. This way, it takes a curve composed of line segments (a Polyline in this context), to find a similar curve with fewer points ([5]). Epsilon represents the maximum distance from the contour to the approximated contour, and is best chosen as 0.1 times the arc length of the contour. After approximating each contour to a shape, we acquire the corners, of which we can then retrieve the amount. If the amount of corners equals four, we are most probably dealing with a painting. Figure 10a shows the detected corners (as green dots) using this method, and Figure 10b shows these corners connected, revealing the detected painting.

(a) Detected corners

(b) Detected painting, drawn over

2.4 Painting extraction and warping

Finally, the painting needs to be extracted and warped to a state were classification can be performed as adequately as possible. This happens by performing a perspective transformation using the corners points that were detected in order to get a top view of the painting. The transformation is defined by a matrix multiplication that can be found when the translation of some points are known. After finding the destination corners by computing the maximum height and width of the painting, it becomes possible to derive the transformation matrix. After applying the transformation on the painting, key-point -and feature-extraction can follow.

2.5 Detector performance

To evaluate the performance of the painting extraction, a test database of photographs was provided, together with the manually annotated coordinates of each painting. By comparing the ground truth coordinates with the extracted coordinates, it's possible to quantitatively evaluate the performance of the painting detector. Namely, in terms of false negatives (paintings missed entirely), false positives (non-paintings extracted) and bounding box accuracy (average intersection-over-union or IoU). The IoU ratio describes the correctness of the segmentation with 100% representing a perfect match and 0% representing a missed match. Figure 5 shows the average IoU for each room. The rooms with a lower ratio are the ones where the walls are not plain white. Consequently, when the background is quite noisy, the extractions tend to perform worse. For example, when there is a shadow underneath the frame, the shadow and background wall differ in color intensity, resulting in an erroneous edge from the Canny edge detector. These edges extend the painting frames and, because the contour with the largest area is sought after, the segmentation step will include the shadow as part of the painting. In total, 740 of 801 paintings where found (61 false negatives). Unfortunately, 441 detections were false-positives, resulting in a rather high false-positive rate.

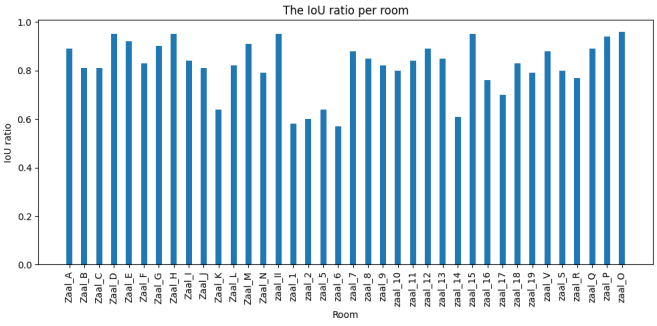


Figure 5: The average IoU per room

This observation let to the introduction of a Support Vector Machine (SVM) classifier that tries to distinguish painting pixels from environment pixels and hopefully lower the false-positive rate while keeping the false-negatives as low as possible. This way, it becomes possible to discard an extracted painting image when it exceeds a certain environment-over-painting pixel ratio. This, in turn, should reduce this rather high false-positive ratio. SVM is a supervised machine learning algorithm as its expects labeled training data, which can be derived from the provided test database. Also, the use of SVM in image classification has shown accurate results in previous research [6]. An SVM tries to find an optimal hyper-plane that creates a boundary between the types of data, as it plots each data item in the dataset in an N-dimensional space, where N is the number of features in the data. Figure 6 illustrates this idea, where the black line represents this optimal hyper-plane and the yellow and red data points each represent a different data type (environment vs painting pixels in this context). The classification was based on the surrounding color- and oriented gradients histogram as these are proven to be robust features [7]. Table 1 gives an overview of the accuracy and the average precision, recall and F1-score of the SVM model on the unseen test set. As a result the false positives are reduced to 103 but as an additional consequence the false negatives are now 197, because some painting images were erroneously classified as walls.

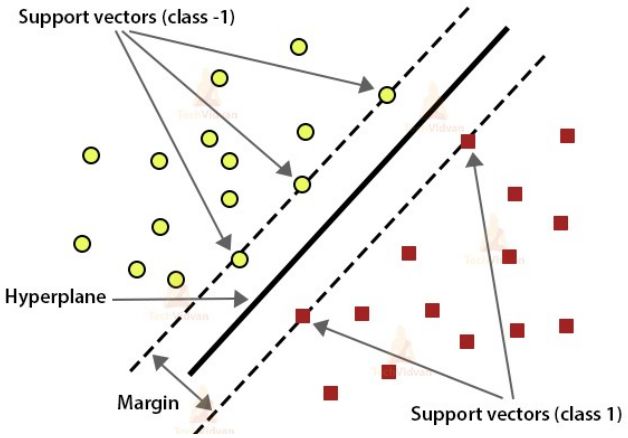


Figure 6: Illustration of the idea behind Support Vector Machines

Evaluation method	Score [%]
Accuracy	80
Precision	81
Recall	80
F1-score	80

Table 1: Evaluation of the SVM classifier

3 Painting matching

Once a painting is detected and extracted from the frames, the program attempts to match this with a painting from a pre-initialized database. The detection of key features (points of interest) and descriptors is handled by SIFT (Scale-Invariant Feature Transform). The features detected by SIFT are scale invariant, rotation invariant and are shown to have robust matching against a range of chance in illumination ([8]). These are beneficial factors for the given environment the matching should take place on (the museum). The paintings in the frames can be far away (lower scale) or the illumination can differ during the day. The matching of the paintings is performed by invoking a matching procedure between an extracted painting and the paintings of the database. Within this procedure, the descriptors of the key-points from the extracted painting are measured against the descriptors from the database paintings. A match between descriptors is defined by its distance metric. The lower the distance, the more likely the match will be valid. For the filtering of the matches, Lowe's ratio test is applied([8]). Each key-point of the extracted painting is matched with a number of key-points from a database painting. The two best matches for each key-point (the ones with the smallest distance) are kept. Lowe's test will check the difference in distance between both found matches. If the distance is not sufficiently different, the match will be discarded. Through Lowe's ratio test, the matches are filtered so that only significant matches are taken into account. The amount of significant matches is the measurement used to determine the strength of a match. The more significant matches are found, the more likely the match will be the extracted painting. After iterating over the database, the matches are sorted by the amount of matches found. The first entry in this collection of matches is the image that is estimated to be the match for the given source. Figure 7 shows an example of the matched key-points between a detected painting and a database painting.

Figure 7: SIFT key-point matching

3.1 Matching performance

The performance of the painting matching is evaluated on a data set of 451 images. The images in the data set are photos of the paintings from a fairly close distance and without a sharp angle. These images are used in order to prevent any problems that could arise with the extraction of the painting and to evaluate as independent as possible. The matching procedure is invoked on every image of the data set and the best match is compared to the original painting. The evaluator

counts the amount of times the original painting is the first entry of the matches (the best match) and the amount of times no matches were found on the extracted painting. This was tested with three different amounts of features to be extracted: 300, 500 and 700. Figure 8 shows the results of these tests in percentage of correct matches on firstly the entire database and secondly on the images on which at minimum one match was found. After evaluation of the paintings with no matches, it became clear that the majority of these images where failed extractions. The percentage *correct with matches* shows a more reliable result on images which are correctly extracted.

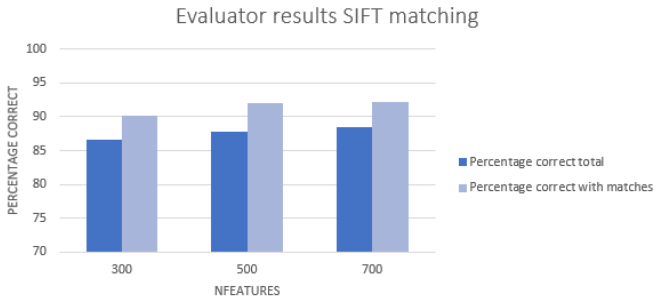


Figure 8: Evaluator results SIFT matching

All three configurations gave a high result for matching 90.09% (300), 92.09% (500), 92.15% (700). The configuration with 500 features was chosen on the grounds that its performance is nearly equal to a higher amount of features and that the amount of features is connected with the time it takes to determine a match. The higher the amount, the slower the matching will be.

3.2 Common difficulties

While the matching on the test set performed relatively well, with a percentage of 92.09% good matches, testing on extracted video frame sets showed increasingly worse results. This is likely a consequence of lower image quality, worse lighting and differences in scale. Initially, the matching algorithm used ORB, which is arguably one of the fastest image matching techniques (namely two orders of magnitude faster than SIFT, [9]). However, while the least computationally expensive, it also performs relatively worse than other, slower techniques. SIFT, its slower, more robust alternative, performs very strong even under scale in-variance ([10]), which is often the case in this use-case. For this reason, SIFT was chosen as matching technique.
Still, lighting and low quality issues significantly influence the performance of the matching. For example, a recurring problem is that extracted images with exceedingly low quality and dark lighting often get matched with paintings with few to no key-points, or do not even get matched at all. This might be explained by the combination of dark lighting and low quality leaving few actual points of interest in the image (e.g. a very dark, poor quality painting of a forest might just look like a dark "blob" instead of defined trees, leaving little features for the SIFT algorithm to detect). Because the painting than has very few key-points, it might get wrongly matched with another painting that truly has few key-points, e.g. a flat painting with little to no details. Consequently, a cascade of erroneous matches and room localizations may occur. For this reason, extracted images were filtered using a bilateral filter, of which the workings have already been discussed in Section 2.1. The purpose of this change is to enhance features that might be too unnoticeable for SIFT, and blur out other details, supposedly leaving stronger features to be matched on [11].
Another possible fix to this problem, might lie in taking the last 30 frames of a video and only matching using the frames with the lowest blurriness or best contrast. This technique will be further explained in Section 5.2.

4 Localization

Next, these matching scores need to be used to statistically determine or guess the location of the user. This will be attempted by using a Hidden Markov Model.

4.1 Hidden Markov Model

A Hidden Markov Model is a statistical model consisting of different states where the current situation can't be observed straight away. The

states are called hidden and are assumed to model a Markov Chain. A Markov Chain describes a sequence of states where the condition of the system depends on the previous situation. A Hidden Markov Model can determine the probabilities of the hidden states based on another process. That process is linked to the original process and consists of states which can be observed straight away [12]. In this project, the hidden states represent the rooms in the museum. The location of the visitor is called the state of the system. The current location depends on the previous location of the visitor. Other than that, the location can not be observed straight away. The paintings are linked with one of the rooms in the museum. The painting detection, as described in Section 1, can recognize paintings in the room. The painting matching, as described in Section 2, can match the detected painting with the paintings in the database. The location of the paintings in the database is known. Based on the matching score between the database painting and the detected painting, the probability of the current room can be determined.

4.2 Transition probability

The firs thing that needs to be done, is to determine the transition probabilities. The first step is to make a connectivity graph base on the floor plan. The graph describes an NxN matrix where N is the number of halls, and each element (i,j) in the matrix is 0 or 1 depending on whether you can move from hall i to hall j or not. The probability whether the user will go to another room or stay in the same room is determined by the number of paintings in the rooms. A function counts the paintings in every room and saves the numbers. Another function directly connects the rooms that are connected with a room without paintings. A last function determines the final transition probabilities in a transition matrix. For each '1' in the connectivity graph, the number of paintings in that room will be divided by the number of the total paintings accessible from the last detected/matched painting. Each element (i,j) in the transition matrix represent the chance that the next detected painting will be in room j, when the last detected painting was in room i.

4.3 Emission probability

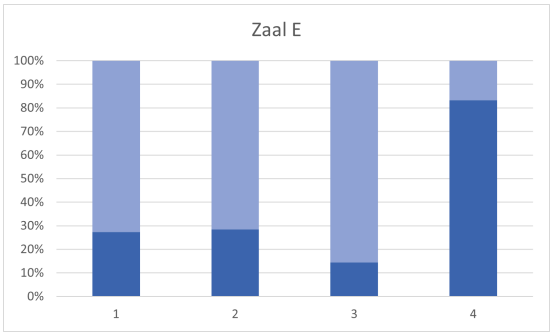
The second step is then to determine the emission probabilities. Those are the probabilities that a certain hidden state refers to one of the observable state. The two observable states will be the chance that the detected painting is in the room and the chance that the detected painting is not in the room. Each room that has a top ten matched image based on the matching scores of Section 3 will be represented by their best match in the top ten. To make those scores discrete, the soft matching score of the eleventh best match will be subtracted from each top ten score and divided by the score of the best matched image.

4.4 Finding current state

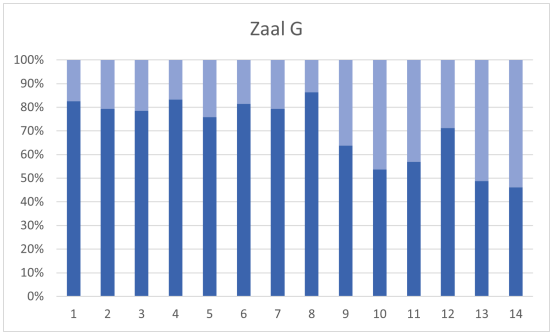
The probability of being in a given state (room), will be determined by the summation of every multiplication of the probability of each room being the previous state and the probability that the visitor travels from that room to the given room. The multiplication of that probability with the emission probability of that state will give the probability that the given state is the current state. The probabilities of the current location will be saved so that next time the probabilities needs to be determined, the previous state can be found.

4.5 Testing results

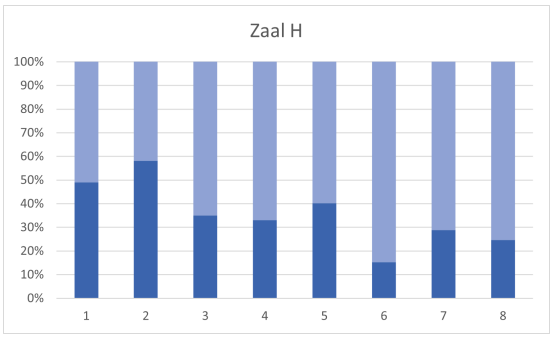
The localization was tested on the first four rooms (Hall E, G, H and M) of the video "MSK10.mp4". The results can be seen on Figure 9. The dark blue represents the probability that the visitor is in the correct room, the light blue the probability that the room is another room. The correct room doesn't always have the highest score, but it's always in the top 5 rooms. The success rate of the localization also strongly depends on the room. Mismatching and a wrong detection strongly affects the success-rate.



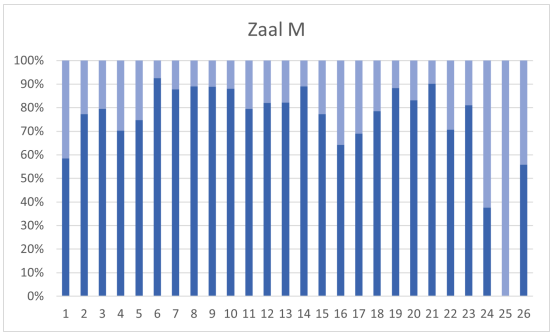
(a) Results Hall E



(b) Results hall G



(c) Results Hall H



(d) Results hall M

Figure 9: Hidden Markov Model testing results

4.6 Optimization

The algorithm behind this localization is the forward algorithm. It computes the total probability of being in the current state regardless of the path taken. The Viterbi algorithm on the other hand, can also evaluate the probability of each path that will lead to the current state. In future work, instead of the forward algorithm, the Viterbi algorithm can be used, to determine more details of the visitors path.

5 Demo

In this Section, the focus lies on running the algorithms created in Sections 1 to 3 against a set of prerecorded video fragments. At first, an attempt will be made to increase the performance of the overall algorithm by subjecting all video frames to analysis and correcting frames if necessary before these are passed on to the painting extraction algorithm. To end this section, a conclusion is built up on the basis of a visual presentation.

5.1 Frame preprocessing

Since the painting extraction algorithm is highly susceptible to deviations within the supplied images, these should be checked first to avoid incorrect results. The provided video fragments are divided in two categories, namely recordings made using a mobile phone and recordings made using a GoPro. The lather uses an ultra wide-angle lens that produces strong visual distortion intended to create wide panoramic images.

5.1.1 FishEye distortion

In order to reduce incorrect painting extractions, this visual distortion needs to be corrected. Typical properties of fish eye lenses are that objects along the optical axis of the lens occupy disproportionately large areas of the image. On the other hand, objects near the periphery occupy a smaller area of the image. In brief, the representation of distance relations in the real world is not the same as in the camera image. This problem can be solved by using a calibration model to estimate the lens parameters which on their turn are capable of unwarping the GoPro footage. The camera matrix as well as the vector containing the distortion coefficients can be obtained by using

OpenCVs camera calibration toolbox. OpenCV determines the constants in these two matrices by performing basic geometrical equations on several camera snapshots of calibration objects. The most commonly used form of calibration object in this case is a chessboard. Figure 10 shows the correction of a fish eye video frame.

(a) Before fisheye correction

(b) After fisheye correction

Figure 10: Fisheye correction

5.2 Frame Analysis

To further improve the frame extraction, a buffer will be used. Every frame will be added to this buffer and will undergo a small analysis that calculates the blurriness and contrast levels of the image. Rather simple methods are used for both of these calculations to reduce the CPU load required for this process. In order to distinguish blurry and non blurry frames, the algorithm makes use of the Laplacian variance. The Laplacian highlights regions of an image containing rapid intensity changes, much like the Sobel and Scharr operators. And, just like these operators, the Laplacian is often used for edge detection. The assumption here is that if an image contains high variance then there is a wide spread of responses, both edge-like and non-edge like, representative of a normal, in-focus image. Contrast level calculation on the other hand is calculated using the standard deviation of the gray scale representation of the frame.

The earlier mentioned buffer will build up to a total of 30 frames, a common frequency rate used in the video industry, before running the painting detection. Only frames that meet both the blurriness and contrast thresholds will be provided to the painting extraction algorithm. Once 30 frames have passed, the frames in the buffer will be ordered from lowest to the highest level of blurriness and the least blurry image will be provided to the matching algorithm which in turn will trigger the Hidden Markov Model.

6 Future work

As previously discussed, one of the main difficulties in this project, lies in the matching between high-quality, well-lit database image and inevitably lower quality frames of a video. To gain better results, the matching could be more refined by basing it off not only a key-point matching technique, but also color histograms, to give an example. The two could then be combined to gain better matching results ([13]). Another difficulty lies in the speed of the program. Because SIFT was used as a matching technique, and each extracted image gets matched against each database image, the program runs rather slow. This could be solved by adding more efficient threading to the program, or only matching the extracted images with a subset of database images of certain rooms, based on a statistical assumption.

7 Conclusion

The goal of this project was to create a program that attempts to localize a person in the Museum of Fine Arts in Ghent through videos taken by this person. With many difficulties faced, this goal was still achieved relatively well. Painting detection performs adequately well, with 740 out of 801 paintings detected from the testing set, of which 103 were false positives. The introduced SVM

classifier lowered the false-positive detections from 441 to 103. Matching also performed well, with a correct match percentage of 92.09%. While due to lighting and quality issues in individual video frames, this matching performed worse, adjustments such as adding a bilateral filter and only using the best frame out of the last 30 frames attempts to solve this partially. The algorithm is able to determine the probabilities of the correct room due to the addition of the Hidden Markov Model. It is possible to view the probabilities of the current room by using a colored map in the Demo. After testing, it can be stated that the correct room does not always receive the highest score. However, it is always at least in the top five most possible rooms, even when there is a mismatch.

Thus, while many enhancements are still possible, the program achieves what it set out to achieve to a certain degree. If this is possible, who knows what the future of computer vision might bring.

References

[1] Jan-Michael Frahm1 Johannes L. Schonberger. Structure-from-motion revisited. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016.

[2] Jason Zhi Liang, Nicholas Corso, Eric Turner, and Avideh Zakhor. Image based localization in indoor environments. In *2013 Fourth International Conference on Computing for Geospatial Research and Application*, pages 70–75, 2013.

[3] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[4] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[5] David H. Douglas and T. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10:112–122, 1973.

[6] Saurabh Agrawal, Nishchal K Verma, Prateek Tamrakar, and Pradip Sircar. Content based color image classification using svm. In *2011 Eighth International Conference on Information Technology: New Generations*, pages 1090–1094. IEEE, 2011.

[7] Michael Villamizar, Jorge Scandaliaris, Alberto Sanfeliu, and Juan Andrade-Cetto. Combining color-based invariant gradient detector with hog descriptors for robust image detection in scenes under cast shadows. In *2009 IEEE International Conference on Robotics and Automation*, pages 1997–2002. IEEE, 2009.

[8] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[9] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011.

[10] Ebrahim Karami, Siva Prasad, and Mohamed S. Shehata. Image matching using sift, surf, BRIEF and ORB: performance comparison for distorted images. *CoRR*, abs/1710.02726, 2017.

[11] Tomoaki Yamazaki, Tetsuya Fujikawa, and Jiro Katto. Improving the performance of sift using bilateral filter and its application to generic object recognition. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 945–948, 2012.

[12] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *PROCEEDINGS OF THE IEEE*, pages 257–286, 1989.

[13] P. S. Suhasini, K. Sri Rama Krishna, and I. V. Murali Krishna. Combining sift and invariant color histogram in hsv space for deformation and viewpoint invariant image retrieval. In *2012 IEEE International Conference on Computational Intelligence and Computing Research*, pages 1–4, 2012.