Feature extraction and matching

Detection of multiple objects in the same scene

Project 8 – Image Analysis and Computer Vision

Barozzi Sara and Matteo Guidi

29-09-2018

Index

1. Introduction and outline of the project

Feature and object detection is one of the principle area of research in the field of computer vision. With this term, we refer to the process of finding point of interest of an image, such as corners and edges, that can be used to describe its content. One of the purposes of this technique is to find correspondence between images. This process is achieved through three main passages. First a feature detection and extrication of interest points. Then the neighbourhood of every interest point is represented by a feature vector, called descriptor, that needs to have specific property as be distinctive and robust to noise. Finally, the descriptors of the different images are matched.

These techniques can have lot of applications that goes from image alignment, stereo 3D scene reconstruction, motion tracking, object recognition, camera calibration techniques to access control to sensitive building, crowd and population statistical analysis, human detection and tracking, detecting of suspicious actions, traffic analysis, vehicular tracking, and detection of military targets.

Due to the huge amount of opportunities given by this method and the spread of cameras and systems that can collect images and videos, the number, type and quality of visual inputs for the detecting system have increased. Because of this, new, automated and sophisticated algorithms for the detection and extraction of features are studied and proposed, to satisfy the requirement of each user. Generally, these algorithms require intensive computation and huge quantities of memory availability, especially in case of high quality video, precise satellite images or real-time applications. Reducing the quality and resolution of the pictures is not the right solution, because it leads to poor performance in feature detection. The state of art in robotics, in fact, has proposed that to solve the technical problem is necessary to find more performing processing algorithms.

In our project, we propose an application of some of these techniques, for the robust detection of multiple similar planar object in a picture, taken with different angular perspectives. In particular, we will consider the following problem: given an image that contains multiple instances of the same object, how it is possible to distinguish the objects in the cases in which they differ from one to another only because of small variations or different colors? Our aim is to find and retrieve *all* the objects that are *exactly* the same.

For the human brain is easy to assess if two images are identical or not, but an automatic procedure to detect such differences is needed in order to make it applicable to different kind of autonomous applications.

More specifically, we will try to achieve this comparison by analyzing the differences between a reference image, from now on called template, and all the parts of the image that are retrieved with the method of feature extraction and matching, from now on called regions of interest (ROI). This method, that is one of the most used techniques, searches for similar features in both the template and the picture, and then match them.

This process is not enough to assess if the ROIs found are exactly identical to the template, since it focuses on matching some parts of the image which have identical features but does not compare the whole image. For example, if two different cereal boxes have the same identical brand, but completely different colors and other parts, they are matched due to the identical features in the

logo. Our project focused specifically on this problem: after the retrieval of the ROIs thanks to feature extraction and matching, the fundamental task is to assess if the two images are exactly identical in their whole parts.

This is indeed a challenging problem since this process is very susceptible to noise, alignment problem and shades in the images. Two images might be identical, but if one of the two there is darker than the other, they might result different. Furthermore, is difficult to understand how much of a change in the two images can determine the difference. Is a different small text enough to say that the two images are not the same? And if a such a small detail can determine the difference between them, how can we be sure that other insignificant details can impact this decision?

An automated method to determine these differences is really hard to create, and so we decided to use two different methods that combined together are giving acceptable results: in order to assess the structural differences in the two images, we used the Euclidean distance and SSIM index; in order to understand if their color is the same, an RGB histogram comparison between the two images, provided that for sure there are some errors due to illumination, occlusion, alignment and shading. For both these methods, it is of the uttermost importance to define a threshold that is able to discard the wrong matches and keep the right one. Furthermore, this value must be flexible and adaptable to the different kind of situation in which this method is applied, and this was a challenging task.
It is also to note that all these procedures were implemented with MATLAB. Other libraries provide already implemented scripts and routines that do some of the parts of the project, while in MATLAB there is no such feature previous to what we have done.

Figure 1 - Reference image and template to be found;

Our report is structured in this way: first a state of the art of the techniques and methods that were most used in our project; then the detailed implementation description and an algorithmic description of what was done; subsequently some cases studied and experimental results; in the end some limitations of our project and possible future works.

2. Overview and State of the art

In order to perform a robust detection by feature matching, the first thing needed is a way to extract and define the features of an image. This process is typically divided in two different parts: the feature detection and the feature description. There are multiple feature detectors and descriptor in the literature, however an ideal detector-descriptor combination that is superior to all the others does not exists. That is because, depending on the image and the application, one method could be better, or worse, with respect to another one.

Feature Detection

Local image features (also known as interest points, key points, and salient features) can be defined as specific structures in the image such as points, edges or objects, that must be unique, repeatable and invariant, in order to be semantically meaningful for the image. Ideal local features should typically have these important qualities:

(1) Distinctiveness: the intensity patterns underlying the detected features should be rich in variations that can be used for distinguishing features and matching them.

(2) Locality: features should be local so as to reduce the chances of getting occluded as well as to allow simple estimation of geometric and photometric deformations between two frames with different views.

(3) Quantity: the total number of detected features (i.e. features density) should be sufficiently (not excessively) large to reflect the frames content in a compact form.

(4) Accuracy: features detected should be located accurately with respect to different scales, shapes and pixels locations in a frame.

(5) Efficiency: features should be efficiently identified in a short time that makes them suitable for real-time (i.e. time- critical) applications.

(6) Repeatability: given two frames of the same object (or scene) with different viewing settings, a high percentage of the detected features from the overlapped visible part should be found in both frames. Repeatability is greatly affected by the following two qualities.

(7) Invariance: in scenarios where a large deformation is expected (scale, rotation, etc.), the detector algorithm should model this deformation mathematically as precisely as possible so that it minimizes its effect on the extracted features.

(8) Robustness: the feature detection algorithm should be able to detect the same feature locations independent of scaling, rotation, shifting, photometric deformations, compression artifacts, and noise.

Feature Descriptors

The goal of a descriptor is to provide a unique and robust description of an image feature, e.g., by describing the intensity distribution of the pixels within the neighbourhood of the point of interest. Most descriptors are thus computed in a local manner; hence a description is obtained for every point of interest identified previously.

The dimensionality of the descriptor has direct impact on both its computational complexity and point-matching robustness/accuracy. A short descriptor may be more robust against appearance variations but may not offer sufficient discrimination and thus give too many false positives.

Once a set of interest points has been detected from an image at a location $p(x, y)$, scale s, and orientation θ, their content or image structure in a neighbourhood of p needs to be encoded in a suitable descriptor for discriminative matching and insensitive to local image deformations. The descriptor should be aligned with θ and proportional to the scale s.

In our work, we decided to use the following main techniques, for the feature extraction and description, but also for the subsequent object detection and comparison:

SURF

Our project is based on SURF, the local invariant feature detection and description algorithm developed by Bay et al. in 2006 (Bay et al., 2006) which aim is to automatically examine an image and extract its feature, and to be able to find these features again in a different picture. This detection should be possible also when the image shows the object with different transformations, like scale and rotation, or when parts of the object are occluded. The process in SURF algorithm is based on 3 steps:

1) Find interest points: to detect interest points, SURF uses an integer approximation of the determinant of Hessian blob detector, which can be computed with 3 integer operations using a precomputed integral image.
2) Find descriptor: feature descriptor is based on the sum of the Haar wavelet response around the point of interest.
3) Matching: a match between two descriptors is a matter of testing if the distance between the two vectors is sufficiently small. The distance measure that is commonly used is the Euclidean distance.

Image comparison

Comparing images is probably one of the most fundamental tasks in computer vision and image analysis. It is mostly used in high-level tasks such as object recognition and image retrieval and classification. Of course, the problem of deciding if two patches correspond to each other or not is quite challenging, as exists too many factors that affect the final appearance of an image, such as changes in viewpoint, variations in the overall illumination, occlusions, shading, and much more. There are multiple approaches in order to solve this problem.

The most basic approach consists in a simple subtraction between the two images. Since an image can be represented as a matrix with a value for each pixel, the difference of two pixel with equal values will be zero. The resulting difference will be a matrix where all the pixel with different values will be shown. This approach is very fast and easy, but is susceptible to noise, alignment problem and shades in the image.

Another method is based on the histogram matching. An histogram is a graphical representation of the value distribution of a digital image. This method is particular reliable when the color is a strong predictor of the object identity. In order to assess if two images are similar, the histograms of the two images are intersected, and a value ranging from 0 to 1 is calculated, based on how much of the histogram are overlying on each other. The closer to 1, the most similar the two images are. The histogram intersection does not require the accurate separation of the object from its background and it is robust to occluding objects in the foreground. Histograms are invariant to translation and they change slowly under different view angles, scales and in presence of occlusions.

The most precise and difficult method is the perceptual hashing method, that is an algorithm that can be used to compare images by calculating the Hamming distance (which basically means counting the number of different individual bits). The perceptual hash is calculated with 4 steps: reduce the size, reduce the color, calculate the average color, calculate the hash. Given two hash, the distance is just the number of different bits of the hash. The two main problems with this approach is it is not rotation-invariant, and it does not work well if the image is damaged.

In our project we decided to use two different methods that combined are giving acceptable results: in order to assess the structural differences in the two images, we used the Euclidean distance and SSIM index; in order to understand if their color is the same, an RGB histogram comparison was performed between the two images, provided that for sure there are some errors due to illumination, occlusion, alignment and shading.

3. Implementation

In the following section it is explained how we proceed for the detection of the occurrences of the template in the picture. We firstly describe the dataset we have chosen; then the main passages followed in the MATLAB implementation of the solution, starting from the extraction of the features to the definition of the thresholds in order to distinguish matching and non-matching boxes; finally, we present an algorithmic description of the whole process.

Dataset

The selected dataset is composed by some pictures that were taken in a supermarket. Each picture shows different shelfs containing mainly cereal boxes and coffee boxes. This subject has been chosen because this kind of boxes are characterized by very variable drawing, that allows different experiments. We could test the algorithm for boxes with same content and different color, or for boxes with similar drawings on them. The pictures are also taken from different perspectives, frontal and lateral, to test if the performances of the algorithm change.

Once the boxes for the different tests were chosen, we downloaded the corresponding templates from internet, trying to find the ones with the highest resolution.

Feature extraction and matching

In order to retrieve planar surfaces objects in a scene starting from a template is to choose the starting pictures. In this case, we took some photos of objects placed on shelves inside supermarkets. In particular we decided to work on 2 different samples: coffee boxes and cereal boxes. The photo taken presented multiple occurrences of the same object. In particular there were multiple coffee boxes with different colors, and multiple cereal boxes with small differences on the graphics. We purposely chose these examples to verify in a better way the behavior of our procedure. Then we downloaded the template of some of the objects from internet. The images below are two examples of the pictures used in the experiments.

Figure 2 - example of frontal picture

Figure 3 - Example of taken picture

The first step to be implemented is the extractions of the features from both the template and the image. The detector and descriptor chosen was the SURF algorithm, because of its robustness and speed. The extracted features of the image have to be matched with the features of the template, by minimizing the Euclidean distance. A linear search process is computationally expensive, knowing that the image's features are typically a high number. A more efficient solution is needed, and the one that is adopted is to organize the data in a KD Tree, to rapidly search for the nearest neighbor. The features of the template are adjusted in a KD Tree and then, for every feature in the image, the two closest features in the template are retrieved.

For every triplet of features (feature_image, closest_feature_template, 2nd_closest_feature_template) found in this way, a ratio test is applied in order to discard the ambiguous matches and the false matches arising from the background clutter.

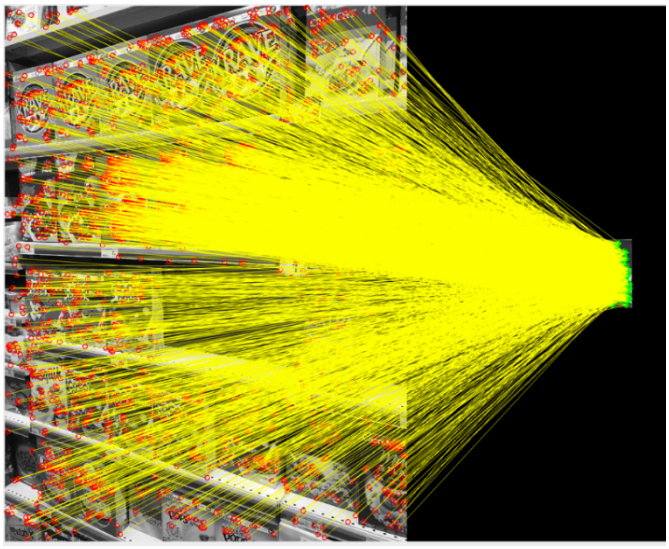


Figure 2 - Feature matching template-image

Retrieval of the regions of interest

We know that each of the occurrences of the template inside the image (from now on called region of interest or ROI) can be associated to the template through a different geometric transformation. In order to find the transformations, an algorithm based on MSAC (M-estimator SAmple and Consensus) method is used. MSAC is a modification of the most known RANSAC algorithm. The main idea is to evaluate the quality of the consensus set (i.e. the data that fit a model and a certain set of parameters) calculating its likelihood (whereas in the original formulation of the RANSAC algorithm, the rank was the cardinality of such set). At this stage both the template and the image are transformed in black and white to simplify the calculation and reduce the computational time. The objective is to find all the geometric transformations that map a minimum number of features (in our case 10) from the image to the template; The transformation to be found is a projective geometric transformation. In this way it is possible to find the region of interest that contains and then find the ROI associated with the template. In fact, from the features that agree with the found transformation, it is possible to find the reprojection of the template on the picture.

Starting from the matched features found before, an initial geometric transformation is estimated using the MSAC method, that is based on 4 steps:

1. A sample subset containing four pair of points is randomly selected from the features of the template.
2. A fitting model and the corresponding model parameters are computed using only the element of this sample subset.
3. All the other features in the image are then tested against the fitted model. The points that fit the estimated model are considered as part of the consensus set and they are called inlier, while the ones that do not fit are called outliers.
4. The model which has the largest number of inliers is selected.

In order to make a more precise transformation, another fitting of the geometric transformation is performed, using exclusively the inliers found before. In this way, the transformation is computed using a higher number of points, thus giving more precise results.

Note that since there are a lot of features, the MSAC method will randomly select four pairs of points, possibly giving different results each time is run. Furthermore, not all the possible combination of points can be testable since it would require too much time.

Once the final transformation has been estimated, is then applied to the contour of the template, which is projected onto the image. In this way we are able to find a resulting box that surrounds a part of the image that supposedly contains the template in the image. However, the transformation found might transform the box in unexpected ways. There are 3 possible cases:

1) The box found is **regular** and **inside** the picture. In this case, the transformation and the ROI inside the box is saved. All the features that are inside the box (inliers of the transformation) are removed from the set of the features, so that in the next step of the cycle they will not interfere with the detection of a new ROI.

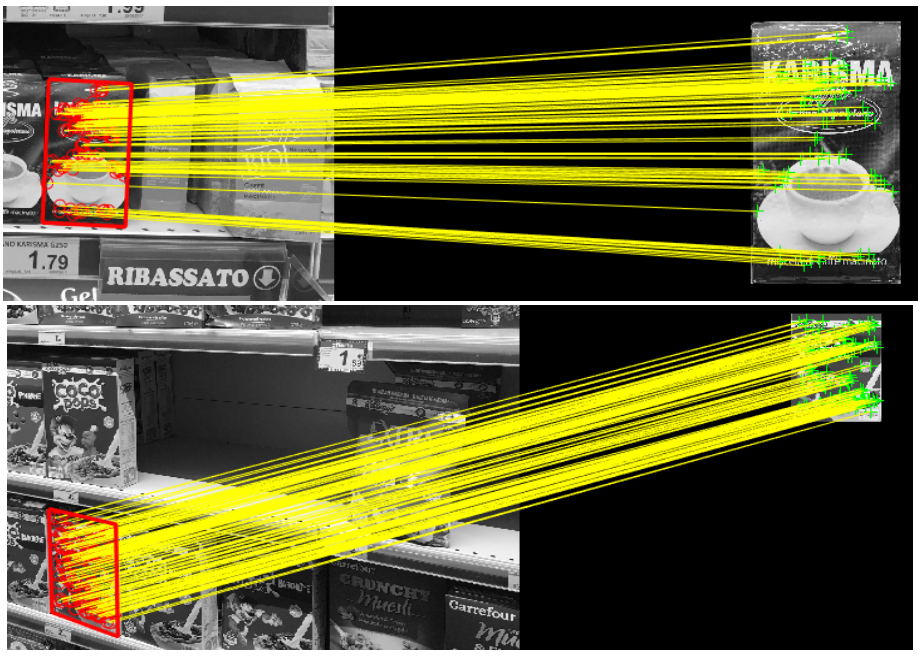


Figure 3 - Examples of perfect matching

2) The box found is **outside** the picture. In this case, the transformation is not saved, but all the features that are inside the box (inliers of the transformation) are removed from the set of features, so that in the next step of the cycle they will not interfere with the detection of other ROI.

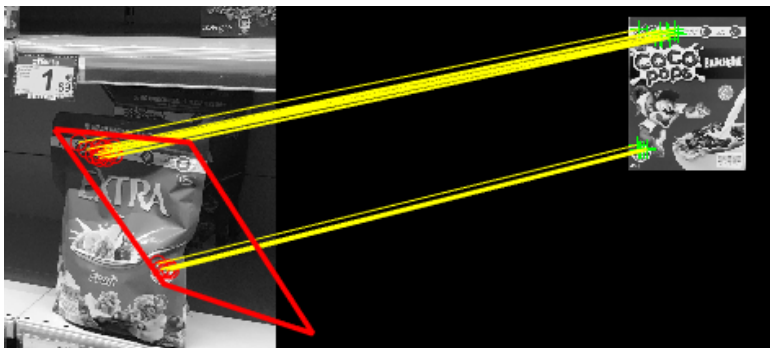


Figure 4 - Bounding box outside the image border

3) The box found is **degenerate** i.e. is concave and its center is outside of the bounding box. In this case nothing is done, so that in the next step of the cycle, the MSAC algorithm might find the regular box associated with the four pair of points.

Figure 5 - Degenerate polygon

This entire process is repeated until one of the two following conditions happens: either the number of degenerate polygons found is greater than 10 (this means that the process is iterating too many times on the incorrect combinations of points), or the number of inliers is smaller than a certain threshold (this condition is set because the geometric transformation cannot be computed with less than 4 points).

The feature extraction and matching technique can find all the repetition of the template in the picture for a variety of cases. Also, the presence of disturbing elements does not block the execution and finding of the good matches. In the reported cases the algorithm extracts more boxes with respect to the correct ones. This is due to the presence of similar parts between the ROI and template. These similar features confuse the matching algorithm, that creates a new box with the few features recognised.

Figure 6 - Detection of a wrong box because of similar features

The red lines (bounding box) surrounding the ROI are perfectly coincident with the border of the coffee boxes for the perfectly matching ROI. The cases in which the bounding box is not perfectly located are the one related to wrong matches. In fact, the presence of some slightly different features (in position and in content) makes more difficult to find a very precise border. The algorithm is also able to locate correctly boxes partially hidden and shaded with a very good precision, as it is possible to see in the reported image.

Figure 7 - Final result of box retrival

Figure 8 - Retrieval of wrong box (top right) and semi-hidden box (left)

Structural comparison ROI / Template

The region of interest retrieved might not show exactly the starting template. This is due to the fact that this whole process is finding occurrences of the template in the image even if there are only 4 joined features. Different object might have very similar, if not identical, parts, like logos and signs. The ROI extracted might be similar to the template or might have small parts that are identical. In these cases, a way to automatically discard the wrong matches is needed. This process is not an easy task, and in the literature, there are multiple ways to achieve this goal. One of the easiest and most used methods is by exploiting the Euclidean distance between two corresponding pixels in the two images.

Once the different ROIs have been found, a comparison with the template is made, in order to assess if they are structurally similar or not. To perform this operation, the two images have to be of the same size and to be comparable pixel by pixel. It is necessary to compare the template with the region of interest of the picture that is matching perfectly, while, at the moment, the various ROIs are still projected. For each ROI that has been found, the associated projective transformation is taken and inverted and then applied to the ROI.

The result obtained is a rectified image that is resized to be of the same dimensions of the template, to be able to proceed with the comparison. A black and white histogram equalization, that is a method to adjust the histogram of the two images in order to make a better pixel by pixel comparison, is performed between the template and each rectified ROI.

The Euclidean distance between template and ROI is computed for every pixel in the two images, and then the mean of these distances is taken. This value is compared with a threshold that is empirically computed by applying this process to different datasets. The value of the threshold has been fixed after several trials. It is impossible to fix a unique threshold value for every dataset. In fact, some photos can be characterized by more noise or less quality or very identical parts. Because of that, the chosen approach is to select the lowest mean distance found between ROI and template. We added to this value a 30% of the same distance. The obtained value is the threshold, that will be a variable number for every example. From the tests done on various pictures we saw that this value is a good limit since it discriminates picture that are structurally different, but keeps the ones that are similar even if affected by noise. The general formula for the threshold estimation is reported in the section: Experimental results. The minimum accepted distance must be lower than a chosen value (empirically chosen), to avoid the case in which the most similar ROI to the template is still very different.

If the value calculated is smaller, then the two images are similar enough. Note that the process of finding if two images are similar or not is a very challenging task, because if two objects are exactly identical except for a small text, they must be classified as different. Is quite easy for the human eye to assess if two objects are identical, but a computer method that is very precise and powerful has yet to be developed.

Another method used to confirm the results of the Euclidean distance method, is the computation of the structural similarity (SSIM) value, a quality assessment index that is based on the computation of three terms, namely the luminance term, the contrast term and the structural term. The overall

index is a multiplicative combination of the three terms and give a value that shows how much the two images are similar with each other. The results obtained further confirmed that the Euclidean distance is an accurate method to assess the similarity between two images, even with all the limitations explained before.

By applying the two methods afore mentioned, only the ROI that are most similar to the template are saved.

Color comparison ROI / Template

Even if the Euclidean distance and the SSIM gives enough information on the similarity of two black and white images, there are some cases where two objects are structurally the same, but their color is completely different. Another comparison between the ROI found and the template is needed, and this time is based on the colors of the two images.

In order to remove brightness and shades problem, the ROIs found are equalized with respect to the template. An histogram equalization is used between the template (taken as reference) and the rectified regions of interest. This passage is important because it allows to reduce the differences between template and ROI due to illumination, brightness, shading variation. In fact, templates downloaded from internet can be characterized by different lights and slightly different colors. In this ways templates and ROI will result more similar. This technique transforms the ROI in such a way that its histogram is matched with the histogram of the template. The distances between the histograms of template and ROI can be computed. This value is then compared with an empirical threshold value, that states if the two images have the same colors or not: if the value is higher than the threshold, the ROI is discarded, else is taken. In this case the threshold has been fixed after different trials. Its value is the smaller distance ROI-template plus 2 times the distance value. This huge difference with the precedent case it is due to the different nature of the comparison that leads to very different distance values. In fact, for the Euclidean distance we need a small variability of the threshold, to be able to detect also small differences between template and ROI, for the color histogram distance we need more variability, to not to discard picture with similar color but shadows, that make the distance value increase. See Example Case 5, where one of the correct ROI retrieved is characterized by and high histogram color distance because it is partially covered by a shadow.

Using this method, the ROI that have a similar structure but different colors with respect to the template are discarded. Therefore, only the ROI that are structurally similar and have the same colors are found. All the occurrences of the template inside the image are correctly found.

The result of our process are all the rectified and correct occurrences of the template in the scene. This procedure can be applied to a vast set of combination template-image.

Another important result is the identification of the two thresholds. They vary every time and they have been identified through iterative and empirical procedure on a set of images. The formula and parameter values used allows to get a correct result in almost every case.

General algorithm

Our whole project can be summarized with an algorithmic process, where the steps are:

Feature extraction and matching

1. Select a picture and a template to be recognized in the picture;
2. Apply SURF algorithm to find:

   X_t=set of x_i feature in the template;

   X_im=set of x_j feature in the picture;
3. Use a KDTree searcher to find, for each template's feature, the two features that minimize the Euclidean distance:

$$x_j^i = \arg\min_{x_j \in X1} \left(\left|\left| x_i - x_j \right|\right|_2 \right)$$

4. Select the neighbour couples (x_i, x_j^i) and (x_i, x_k^i) and perform a ratio test to discard the ambiguous matches, using this formula:

$$\frac{\left|\left| x_i - x_k^i \right|\right|_2}{\left|\left| x_i - x_j^i \right|\right|_2} < 0.8$$

Retrieval of the regions of interest

5. Find all the occurrences of the template inside the given image:

   While: number of inlier > threshold:

   Use a MSAC algorithm to estimate the projective geometric transformation H∈ R^{3x3} between the template and the object located in the image:

$$H: \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

   Find the box surrounding the reproduction of the template in the picture;

   Eliminate all the degenerate and out of bounds boxes;
6. Rectify the found boxes and the part of image inside them:

   For each H in the set of found transformation:

   For each pixel x ∈ to the box found in the image and for each pixel X ∈ to the rectified box, find H^{-1} that satisfy:

$$X = H^{-1} * x$$

   Resize the box to the dimension of the template;

Structural comparison ROI / Template

7. Do an histogram equalization of the template;

8. For each image P_j in the set of rectified images:

   Transform the picture to BW;

   Do an histogram equalization of the picture;

   Calculate the Euclidian distance between the template and the image:

$$D_j = \frac{\sum_i \sqrt{h_1(i) - h_2(i)}}{n}$$

   Where *h1* and *h2* represent the values of the *i* pixel in the pictures and *n* is the total number of pixel;

9. For each distance Dj in the set of rectified images' distances:

   If $(\boldsymbol{D_j} > \mathbf{min}(\boldsymbol{D}) + \mathbf{0.3} * \mathbf{min}(\boldsymbol{D}))$ $\boldsymbol{where\ min(D) < maxEuDist}$**:**

   Where D is the set of all distances template-image

   Then: Discard the picture;

Color comparison ROI / Template

10. For each image P_j' in the set of correct images:

    Do an histogram equalization;

    Take the RGB corresponding image;

    Calculate the histogram distance between the coloured images and template;

11. For each distance D_j' in the set of correct images' distance from template:

    If $\boldsymbol{D'_j} > \mathbf{min}(\boldsymbol{D}) + \mathbf{2} * \mathbf{min}(\boldsymbol{D})\ \boldsymbol{where\ min(D) < maxHisDist}$:

    Then: The image found is an occurrence of the template, and is one of the final results;

4. Experimental results

The following section reports some result example. It shows in practice the limits and the capabilities of the tested algorithm. Case 1 shows a perfectly working case, Case 2 shows that the algorithm is able to distinguish color and little variations; the third example, called Case 5 because this is the name of this example in the MATLAB script, shows that the algorithm can distinguish boxes with very similar structures, also in the case of shadows in the picture. All the cases are presented following the structure: 1) Feature extraction, matching and retrieval of the region of interest, 2) Structural comparison, 3) Color comparison with equalized and non-equalized pictures.

In the third section we decided to show the results for both the equalized and non-equalized images, in order to better understand the differences that this process brings. In fact, in the first two example the histogram equalization might not be necessary, while in the third one is absolutely needed to retrieve the right boxes.

The following and other non-reported case studies have been used to find more accurate parameters to be used for the selection of the pictures.

In order to verify the correctness of the Euclidean distance method, used to compare two images and assess if they are different, another approach has been adopted. This approach uses SSIM, a structural similarity index between the pictures. The result of this trials confirmed that the Euclidean distance is a suitable method for image comparison. SSIM results are not reported in this document but this kind of procedure can be tried in the MATLAB script.

One of the main goals of the project was the estimation of threshold and parameters that allows to reach a very precise and correct results. These values cannot be fixed in an absolute way for all the case test but are calculated as a percentage increase of the smallest differences obtained in each case. The percentage variation values have been obtained empirically. The formula for the computation of the threshold is:

$$\boldsymbol{T = \min(D) + Pvar * \min(D)\ )}$$

Where T is the threshold, Pvar is the percentage variation and D is the set of distances template-ROI.

We have fixed also a value for the minimum distance D both for Euclidean and histogram comparison. This value is needed in case of huge differences between the template and the most similar region of interest. A case in which this parameter is important is reported in example case 2.

The final parameters selected are:

- Minimum number of inlier = 10;
- Maximum Euclidean distance = 0.25;
- Percent Euclidean= 30%;
- Minimum ssim value = 0.4;
- Percent SSIM =25%;
- Maximum histogram value = 0.1;
- Percent Histogram= 200%;

Example: case 1

This example is the basic one, it shows how the algorithm works in the easier case. The template is blue, while in the picture there are blue and green boxes. The green one have some slight variation also in the structure, because the name written inside is different. From this example we can see that our treshold on the euclidean distance it is not able to discriminate in all the cases between green and blue boxes, because the difference is very small. The correct boxes are finally detected with the histogram comparison. When the ROIs have been equalized, is possible to see that there is a huge Euclidean distance difference between the boxes with different colors.

Picture: Template:

1) Feature extraction, matching and retrieval of the region of interest

Figure 9 - Box founded with feature matching and homography technique

Boxes retrieved and warped to re-align them with the template:

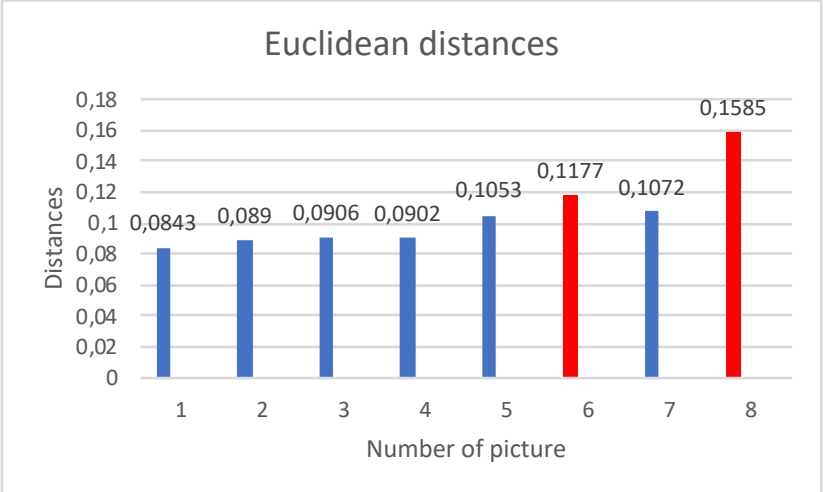
1 2 3 4 5 6 7 8

2) Structural comparison: euclidean distances with black and white equalized images

Minimum distance found and value of the treshold in this case:

- Minimum distance template – box = 0.0843
- Maximum accepted distance between pictures: 0.0843 + 0.0843*0.3 = 0.1096

In the graph below the correct pictures are shown in blue, the discarded one are in red.



3) Color comparison

Distances with non equalized histogram:

Minimum distance found and value of the treshold in this case:

- Minimum distance: 0.0472
- Maximum accepted distance: 0.1416

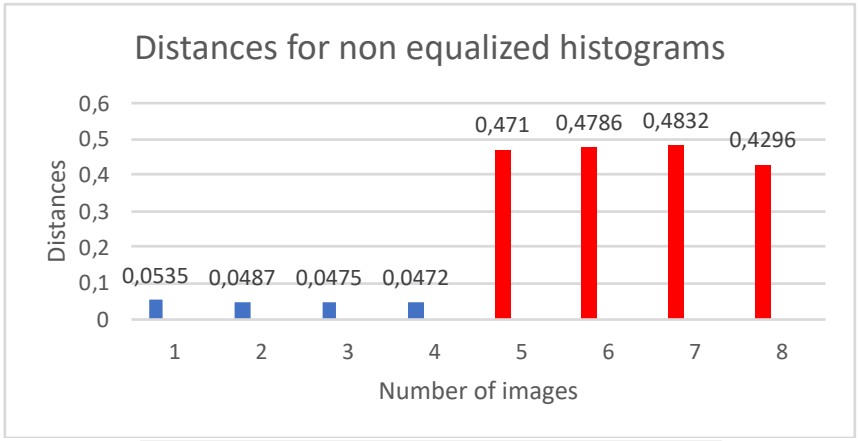


Figure 10 - Resulting non-equalized images

Distances with equalized histogram:

Minimum distance found and value of the treshold in this case:

- Minimum distance=0.0093
- Maximum distance accepted=0.0093+0.0093*(200/100) = 0.0279

In the graph below the correct pictures are shown in blue, the discarded one are in red.

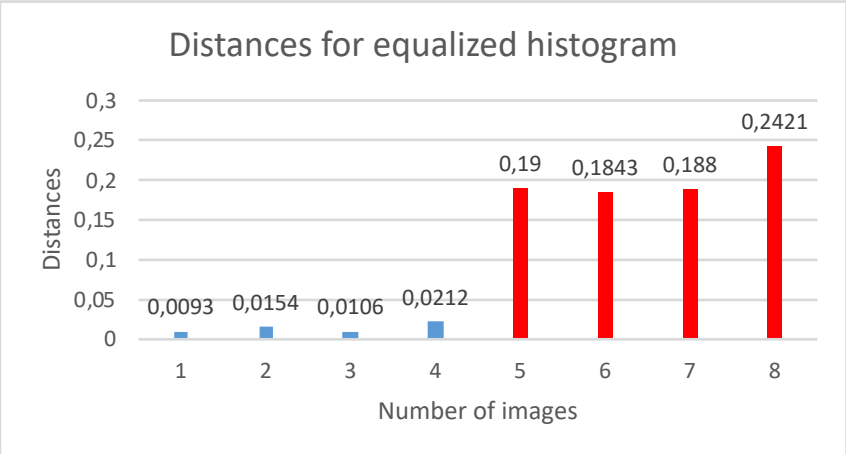


Figure 11 - Resulting equalized pictures

In this case the histogram matching with equalized or non-equalized images gives the same results because the colors are very different.

Example: case 2

This comparison has been done to verify if our technique is able to identify changes in color. In fact, during the feature detection, all the present coffee boxes are recognized, also if they have some small differences from the template. Because of this, as we can see from the following graphs, the Euclidean distances ROI-template are all quite high (but still, lower than the maximum distance value) and are all similar one to each other. It is impossible from this stage to understand that the template is different from the box retrieved in the image. To properly select the ROI, it is necessary to do an histogram comparison on equalized pictures. After that is possible to see that the color difference is high. In fact, all the difference values are higher that the threshold of maximum histogram distance.

Image:

Template:

1) Feature extraction, matching and retrieval of the region of interest

Figure 12 - Founded region of interest

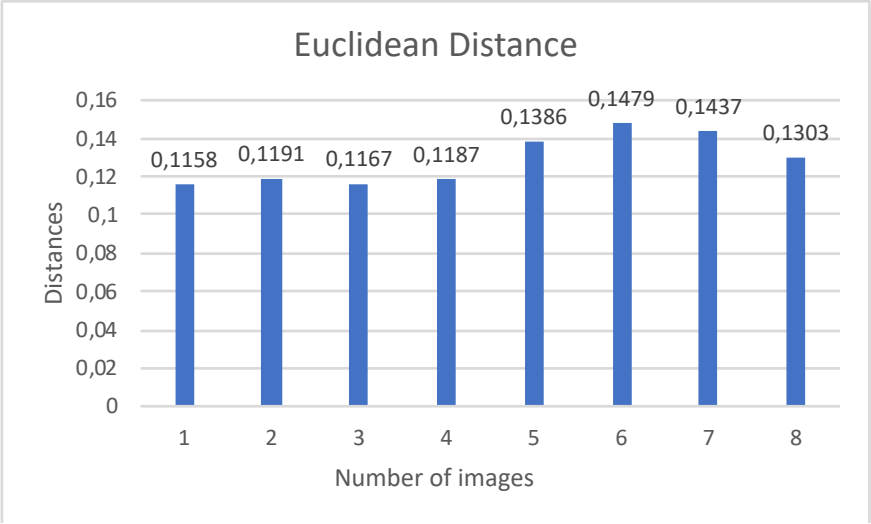
Rectified and reshaped ROI:

2) Structural Comparison: euclidean distances with black and white equalized images

Minimum distance found and value of the treshold in this case:

- Minimum Euclidean distance = 0,1158
- Maximum accepted distance = 0,150504

In the graph below the correct pictures are shown in blue, the discarded one are in red.

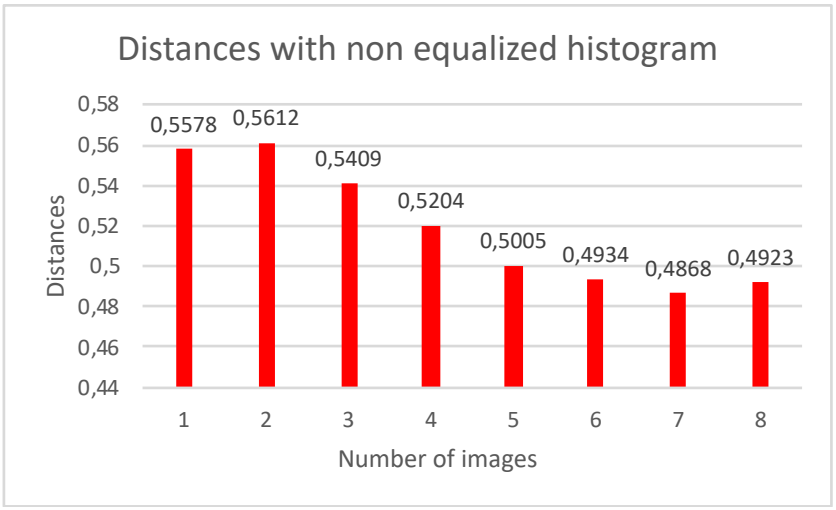


In this case, since all the ROIs found are structurally similar to the template, there is no ROI to be discarded.

3) Color comparison

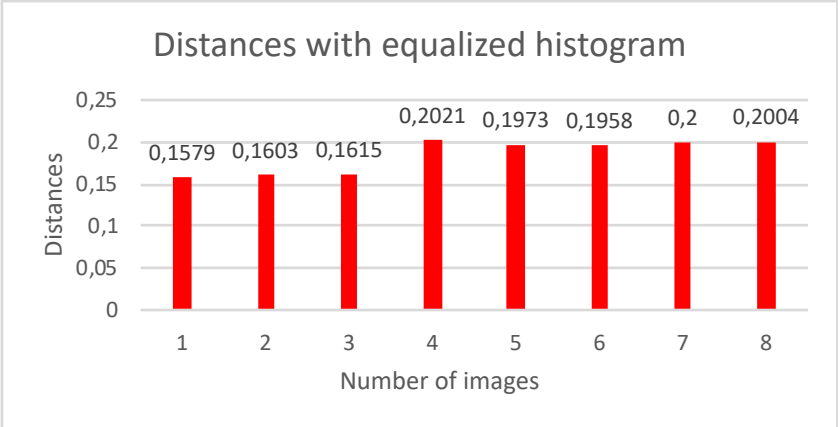
Distances with non equalized histogram:

- Maximum accepted histogram distance (working parameter): 0.1



Distances with equalized histogram:

- Maximum accepted histogram distance (working parameter): 0.1



In this case all the values found for the equalized histogram are over the value of the threshold (0.1), so we need to discard them all. This is coherent, since the template was red and there are no red coffee boxes in the image.

Example: case 5

In this example we try to detect a particular box. From the picture we can see that there are 4 boxes, and one of them is quite covered and shaded. During the first stage of ROI detection, the algorithm finds correspondences with some wrong box, because they are made by the same company (Kellogg's) or have the same name (Choco Pops). This example is particularly significant for the histogram equalization: in fact, without it, our algorithm is not able to find any right boxes; when it is performed, all the right boxes are retrieved.

Image:

Template:

1) Feature extraction, matching and retrieval of the region of interest

Figure 13 - Founded region of interest

Warped ROI:

1 2 3 4 5 6 7

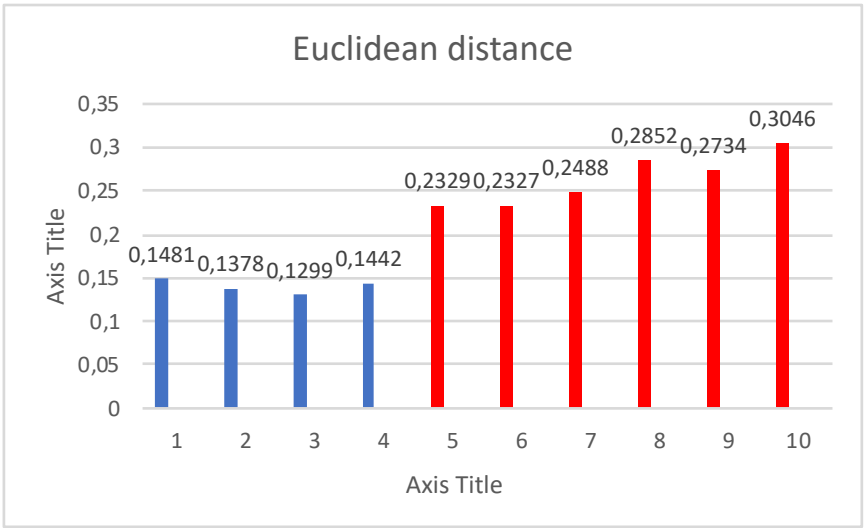
8 9 10

2) Structural comparison: euclidean distances with black and white equalized images

Minimum distance found and value of the treshold in this case:

- Minimum distance = 0.1299
- Maximum accepted distance= 0.1688

In the graph below the correct pictures are shown in blue, the discarded one are in red.



By discarding all the pictures that have an Euclidean distance grater than the threshold, all the picture that does not contain the template are discarded. The resulting correct ROI are show below;

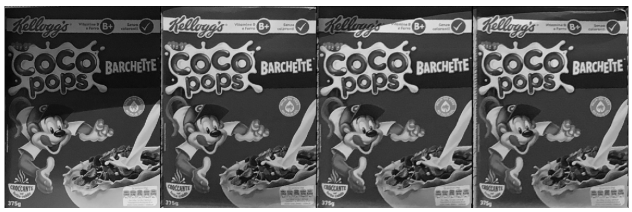


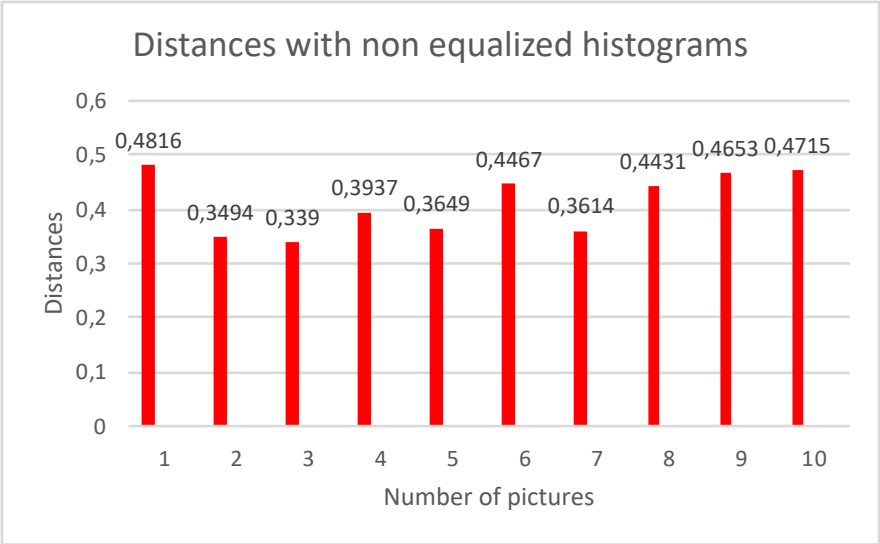
Figure 14 - Images with Euclidean distance lower than the threshold

We want to evaluate if our algorithm is able to detect the variations of colour in the pictures, so we tried to test on all the ROIs found, without discarding the images that have Euclidean distance lower than the threshold.

3) Color comparison

Distances with not equalized histogram

- Maximum accepted histogram distance (working parameter): 0.1



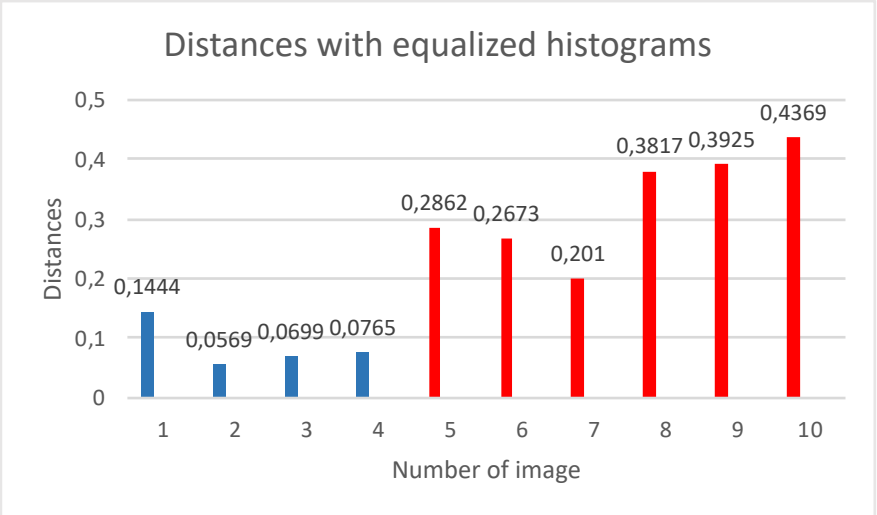
In this case we can see that all the values of the distances between the histograms non-equalized are above the minimum threshold, and it would be impossible to distinguish between correct and not correct pictures.

Distances with equalized histogram:

Minimum distance found and value of the treshold in this case:

- Minimum distance = 0.0569
- Maximum accepted distance= 0.1707

In the graph below the correct pictures are shown in blue, the discarded one are in red.



In this case, by evaluating the pictures with respect to the selected threshold, it is possible to select the following ROI.

Figure 15 - Selected equalized region of interest

Figure 16 - Selected area in the picture

5. Limits of this project

The focus of this study is on the detection of planar surfaces shaped as boxes. The detection of other kind of geometric frames has not been analyzed.

The main result of this project is the finding of an algorithm that can deal with the detection of multiple boxes. This type of detection is quite complicated, in fact when multiple occurrences of the object that needs to be detected are present in a picture, is easy to have some noise that can make the result of the algorithm unreliable. In fact, the thresholds that we have chosen can discard some correct results. An example of correct box that is discarded can be a damaged box: the threshold for the Euclidean distance is quite strict, to be able to detect small variations of the structure of the ROI selected when compared with the template; in this case the differences in the box will be considered as difference in the structure and the box will be wrongly discarded.

Another characteristic of the algorithm, that can lead to variations in the result is that the iterative procedure for the feature selection is random. This means that every time the process is run, it can give slightly different results and different ROIs. In particular, the highest variations are present for boxes that match with a low number of features.

Moreover, the search of the ROI is computationally complex and consequently quite slow.
Because of this for images with a considerable number of ROI the memory available needed by MATLAB to create some vectors may not be enough. The process, in this case, is blocked during the computation of the rectified images. Some improvement for the reduction of the occupied memory have been done in the code, but they are not enough to make it work on less powerful PCs.

Due to the long duration of the process we performed a time speed analysis of our code to find the most limiting steps. This procedure has shown that the two most time requiring operations are the *estimation of all the geometric transformations* and the *computation of the rectified boxes*.

The first one is an iterative procedure on the features that are found in the image. The MSAC algorithm cycle through all the possible combinations of features in order to find the 4 pairs that have the highest number of inliers. This procedure is computationally expensive, especially if there are a lot of occurrences of the template in the image.
We tried to speed up the process by *limiting the maximum number of trials* to 50000 and for each cycle we decided to *remove* all the features that have already been matched. This value can be changed: if increased, more boxes will be found with a higher precision, but the required time will increase; if decreased, the computational time will be much shorter, but a smaller number of boxes will be found and in a less precise way.

The second most time requiring operation is the *computation of the rectified boxes*, that is a transformation that requires lot of computational power, because it tries to find the inverse geometric transformation to bring the detected box and the part of image contained in it to a rectangular shape. The first boxes are detected fast, because when there are a lot of features that matches between the template and the ROI, the computation of the transformation is easy, precise and fast. After that, the boxes that match with a lower quantity of feature are detected. In this case,

the inliers detected are much less, and the computation of the transformation requires more time and is less precise.

The results of the time speed analysis and of the comparison of the time needed to complete the MSAC algorithm done for two different pictures is shown in the next paragraph.

1) For the first image we can see in the reported graph that while increasing the number of trials, the time to complete the step of the finding of the homography increases. At the same time, it increases also the number of boxes detected in the picture, starting from 9 with 5'000 trials to 13 with 150'000 trials. Because of that we chose as parameter for the number of trials 50'000, that is a good compromise between required computational time and quality of the result. In fact, there is no significant change in the quality of the detection passing from 50'000 to 150'000. The pictures represent the boxes found in the case of 5'000 and 150'000 trials.

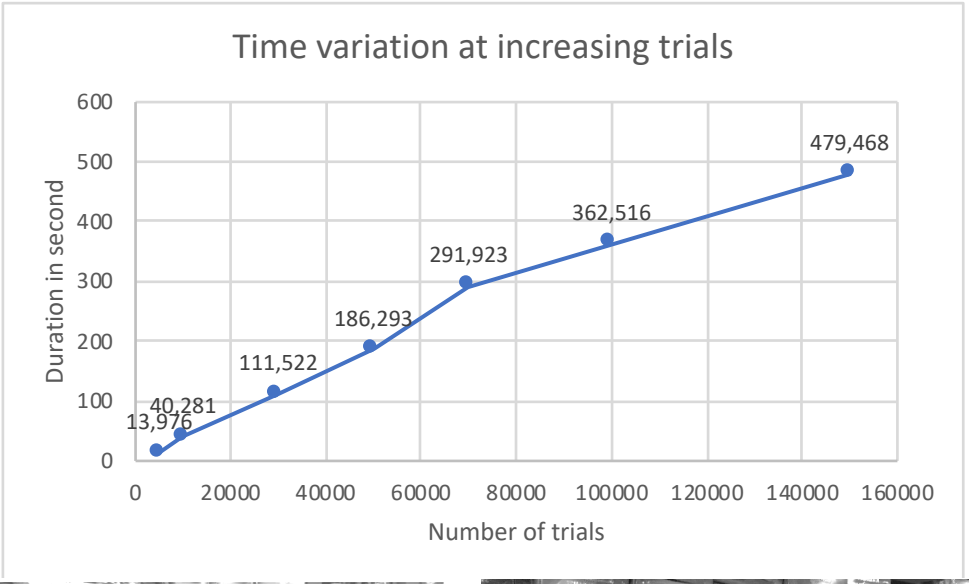
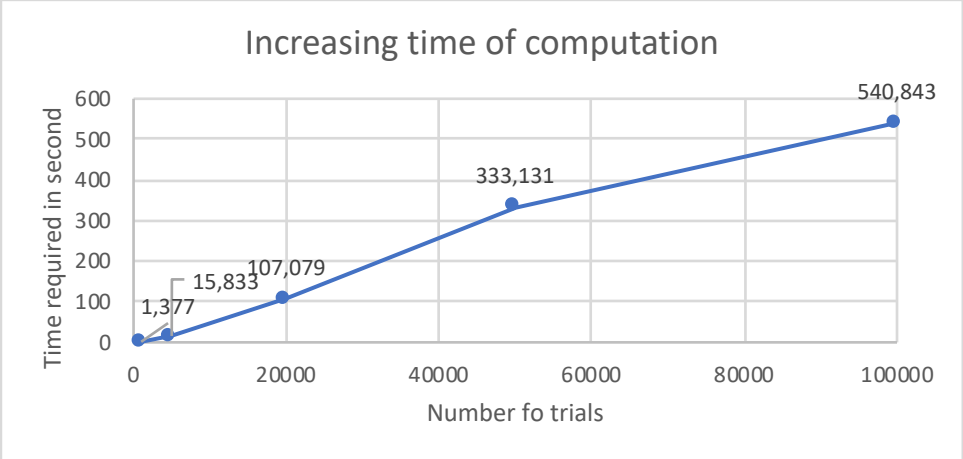


Figure 17 – Difference of retrieved boxes between MSAC with 5000 trials and 150'000

2) In the second example, we can see the increasing time of computation and the variation of detected boxes with the increasing number of trials in MSAC algorithm. From the picture it is possible to see that there is not a significant increasing in the quality of the outputs passing from 50'000 to 100'000, while at 30'000 some wrong boxes are found.



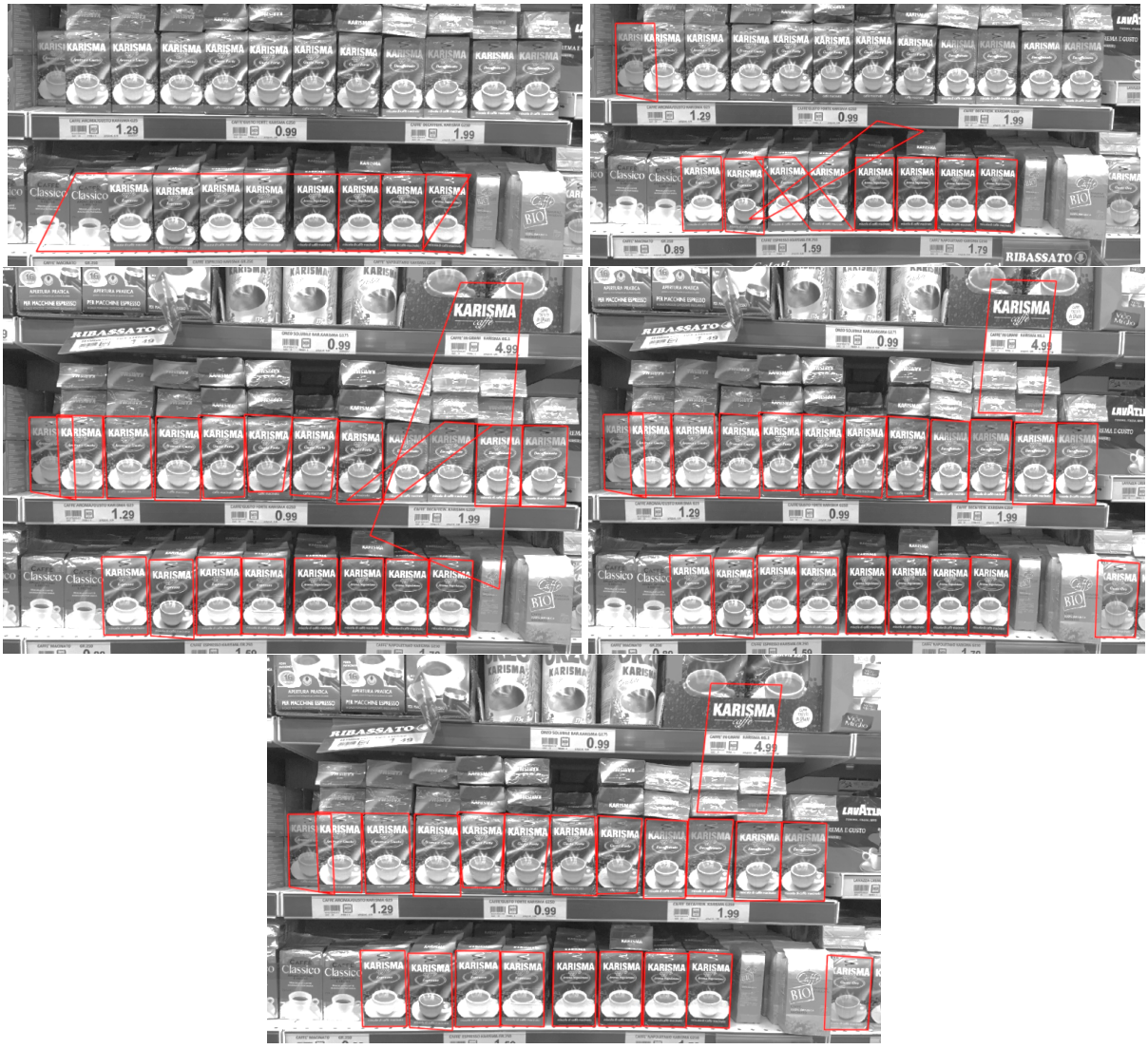


Figure 18 - Boxes detected in pictures with 1000, 5000, 30000, 50000 trials

6. Conclusion

The presented algorithm and examples show the operational feasibility of our technique. The procedure can detect multiple object in an image, both in the case of picture taken with frontal or accidental perspective and in case of picture containing a high number of disturbing elements. The proposed steps to eliminate the non-correct ROI works in the simple case of very different color and structural differences, but also in case of small variations. The improvement done in the code, like the histogram equalization, allow to recognize as similar two images also in presence of shades that make the image darker.

References

- Class notes
- '*SURF: Speeded Up Robust Features*' Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, 2006
- '*Recent Advances in Features Extraction and Description Algorithms: A Comprehensive Survey*' Ehab Salahat, 2017
- *'Euclidean distance mapping'*, Per-ErikDanielsson,1980
- *'Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography'*, Martin A. Fischler, Menlo Park, Robert C. Bolles, Menlo Park
- *'Overview of the RANSAC Algorithm'*, Konstantinos G. Derpanis,2010
- *"MLESAC: A new robust estimator with application to estimating image geometry"*, P. H. S. Torr, A. Zisserman, Computer Vision and Image Understanding, vol 78 Issue 1, April 2000, pp 138 - 156.
- *"Implementation and Benchmarking of Perceptual Image Hash Functions"*, Zauner, Christoph, 2010