PROJECTE FINAL DE CARRERA

APORTACIONS ALS SISTEMES DE RECONEIXEMENT DE LOGOTIPS EN ESDEVENIMENTS ESPORTIUS

CONTRIBUTION TO LOGO RECOGNITION SYSTEMS IN SPORT EVENTS

Estudis: Enginyeria de Telecomunicació

Autor: Pablo Beltran Sanchidrian

Director/a: Eugene Mbanya

Any: 2013

Resumen

Esta memoria presenta un sistema para detectar logotipos en eventos deportivos desarrollado en el Fraunhofer - Heinrich Hertz Insitute. El sistema se basa en los métodos de reconocimiento de objetos actualmente utilizados que consisten en la extracción de patrones que representan al objeto y búsqueda de estos en la escena. La implementación de uno de estos sistemas propuesto por David Lowe ofrece una buena precisión y una tasa de reconocimiento aceptable. Adicionalmente, en este trabajo se propone una mejora del tiempo de ejecución que consiste en la agrupación de los patrones utilizando grafos. Esta mejora es aplicable no sólo al reconocimiento de logotipos sino al de cualquier tipo de objeto. Finalmente, se estudian otros métodos para mejorar la tasa de reconocimiento y la respuesta del sistema a imágenes codificadas de forma entrelazada.

Resum

Aquesta memòria presenta un sistema per detectar logotips en esdeveniments esportius desenvolupat al Fraunhofer - Henrich Hertz Institute. El sistema es basa en els mètodes de reconeixement d'objectes utilitzats actualment que consisteixen en la extracció de patrons que representen al objecte i la cerca d'aquests a l'escena. La implementació d'un d'aquests sistemes proposat per David Lowe ofereix una bona precisió i una taxa de reconeixement acceptable. Adicionalment en aquest treball també es proposa una millora del temps d'execució que consisteix en la agrupació de patrons utilitzant grafs. Aquesta millora és aplicable no només al reconeixement de logotips sinó al de qualsevol tipus d'objecte. Finalment s'estudien altres mètodes per millorar la taxa de reconeixement i la resposta del sistema a imatges codificades de forma entrellaçada.

Abstract

This report presents a system to detect logotypes in sport events developed at the Fraunhofer – Heinrich Hertz Institute. It is based on the actual methods used to recognize objects, which look for patterns extracted from them. The implementation of David Lowe's version shows a good precision and recall for this task. Additionally, a new clustering technique that speeds up the execution time thanks to the use of graphs is proposed. This improvement can be applied not only to logos but to any kind of object. Finally, other methods aimed to improve the recall ratio are evaluated and the response of the system to interlaced images is tested.

Acknowledgements

I would like to thank Dr. –Ing. Patrick Ndjiki-Nya, head of the Content Aware Image Processing group at the Fraunhofer – Heinrich Hertz Institute, for giving me the opportunity to do my thesis in his group and for making me feel welcomed in his team. Also, I would like to thank my advisor, Eugene Mbanya, for his guidance and help during this work.

I am grateful to rest of the members of the group and the other researchers at the Fraunhofer - Heinrich Hertz Institute for their valuable discussions and the warm environment they created.

I thank my advisor at the Polytechnic University of Catalonia (UPC), Prof. Dr. –Ing. Ferran Marques, for his continued help and encouragement over the last year.

I am also thankful for the support I received from my family and friends throughout my degree program. Finally, I would also like to thank the people I have met these eight months in Berlin, for making me enjoy that great city.

Contents

Figure Index

Chapter I - Introduction

This project has been developed at the Content Aware Image Processing Group in the Fraunhofer - Heinrich Hertz Institute between October 2012 and May 2013. The Fraunhofer Society is a German organization founded in 1949 and it is composed by 60 research centres dedicated to different fields of applied science. The Heinrich Hertz Institute is located in Berlin and its research is focused in four fields of competence related to communications systems and digital media.

The Content Aware Image group belongs to the Image Processing Department in the electronic imaging field and its goal is to automatize methods to extract, create and deploy semantic information of audio-visual content. One of their research topics is automatic soccer video analysis, which aims at creating algorithms for the efficient annotation and retrieval of soccer contents, as well as using the generated metadata to create a more interactive experience when watching soccer games. This project is a part of European project TOSCA – MP (Task-Oriented Search and Content Annotation for Media Production) which develops user-centric content annotation and search tools for professionals in networked media production and archiving.

Figure 1 – Fraunhofer HHI and Tosca-MP logo

Among the several tasks contained in this research topic, a logo recognition and localization system offers the possibility of evaluating the brands appearing during a soccer game. Soccer games are one of the most viewed events in television so this information can be useful for the companies because it can assess the impact of an expensive advertising campaign. Hence, the main goal of this project is to recognize in each frame of a match if a certain company is advertised on the billboards by detecting their logo and what surface area of the frame is taken up by it.

1 - Problem statement

Many challenges can arise in this task of detecting and localizing a logo in videos of sport events. Firstly, the logo usually appears distorted by the camera viewpoint. This distortion can change its size and its shape, making it unrecognizable in some cases. Secondly, it can be partially occluded by the players, the referees or any other object between the camera and the logo. Thirdly, the frame containing the logo can contain some noise produced by the lenses or it may not be of sufficient resolution to identify smaller-sized logos. Also, some companies change their logos over time and it will be viable to find them only if the differences between them are minimal, otherwise it will not be possible (see Figure 2):

Figure 2 - Shell logo evolution

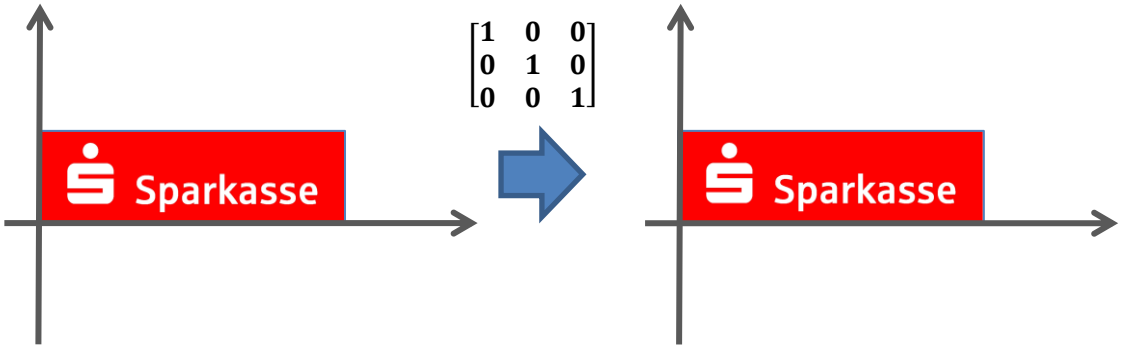
1.1 - 2D transformations

All the objects in a scene are perceived in a different way depending on their position and on the observer's viewpoint. In the case of 3D objects, the point of view determines not only which parts of the object are occluded by itself and which ones are visible but also how the appearance of the visible parts will be. If the viewpoint changes, the appearance perceived of the object is modified and it can be considered that the object has been subject to a transformation.

Some transformations preserve part of the characteristics of the original image, as the angles formed by two lines or the distances between points, but complex transformations can be decomposed into simpler ones. The representation of the transformations can be done with a 3x3 matrix (for planar objects) called the transformation matrix, that groups the functions which map the points from one viewpoint to the other. The simplest transformations are:

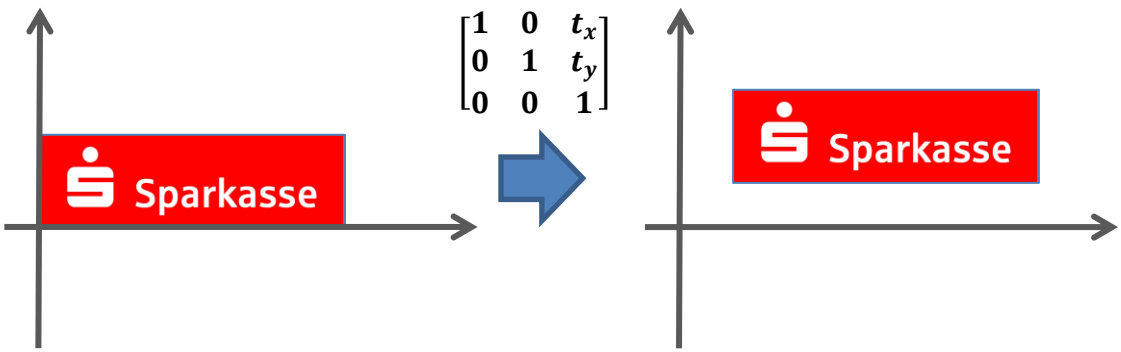
1.1.1 - Identity

The most basic transformation is the identity transformation, which maps all the pixels from an image to the other without making any change. Hence, the transformation matrix only contains ones in the diagonal (the identity matrix):



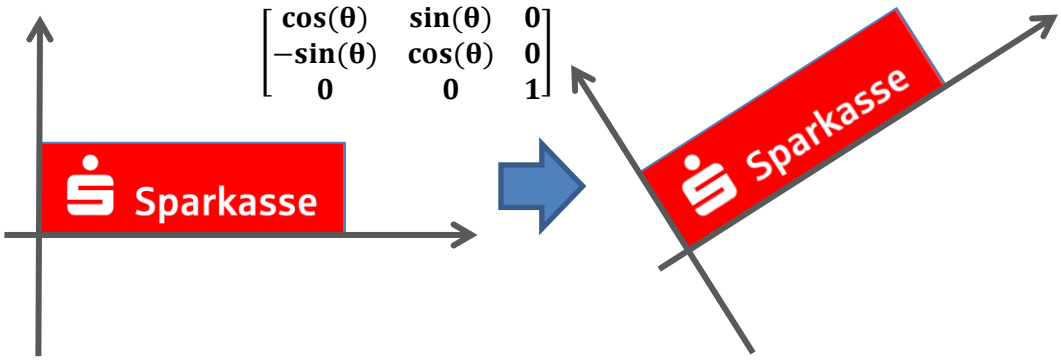
1.1.2 - Translation

All the logos that appear in a soccer game are subject to a translation (to the billboards in our case). A translation moves every point of the image the same distance in a specified direction. It adds an offset vector to each point, hence preserving all the distances between points in the object (it is an isometry).



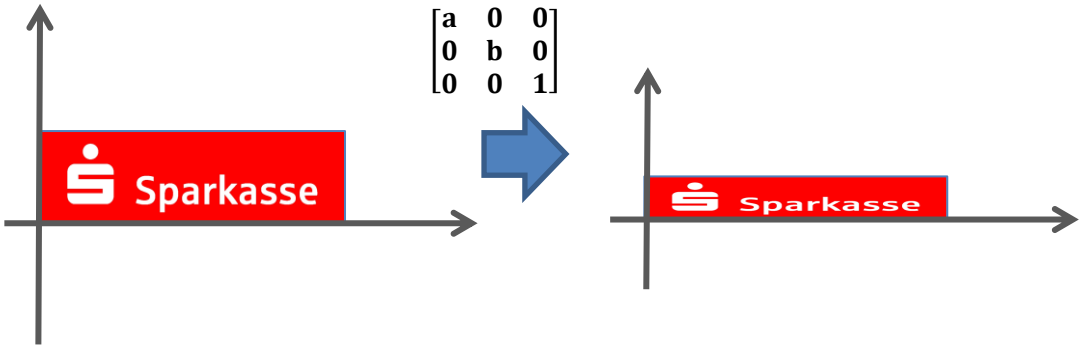
1.1.3 - Rotation

Some of the logos in the scene are rotated. A rotation moves all the points circularly at a given angle θ about the coordinate axis. It is also an isometry, because the distances and the angles between points are also preserved.



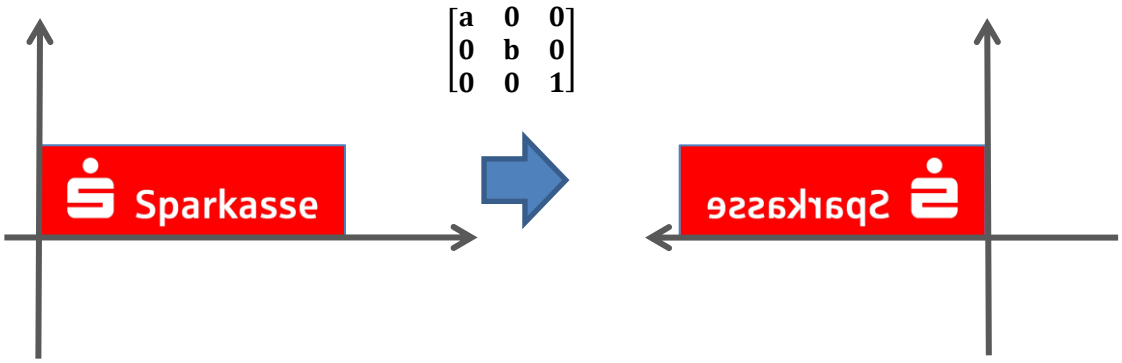
1.1.4 - Scaling

Usually the logos do not appear in the same size in all frames, most of them are subject to a scaling that enlarges or shrinks them. If the scaling factors ("a" and "b" in the example) are equal the transformation is called an isotropic scaling and it preserves angles and length ratios, otherwise it is called an anisotropic scaling and only the angles between parallel lines are maintained.



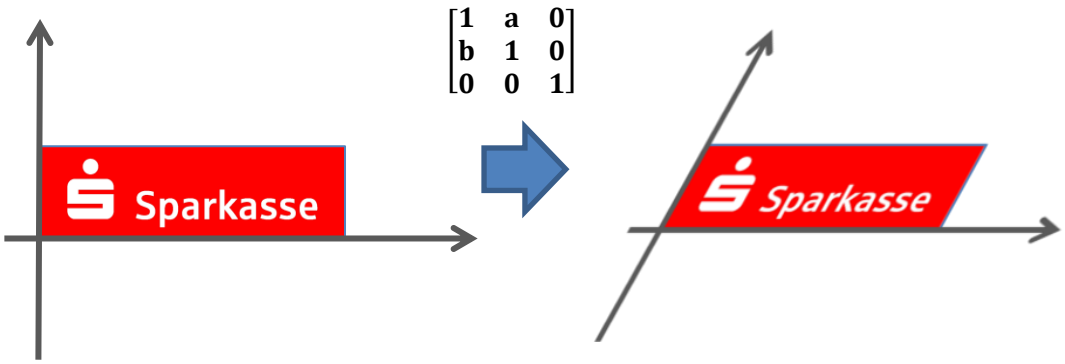
1.1.5 - Reflection

A reflection is an isometric transform that mirrors the logo around the coordinate axis, so it is not common in a football match. In its transformation matrix the "a" or "b" values have to be equal to -1 depending on whether the object is reflected around the vertical axis or the horizontal axis.



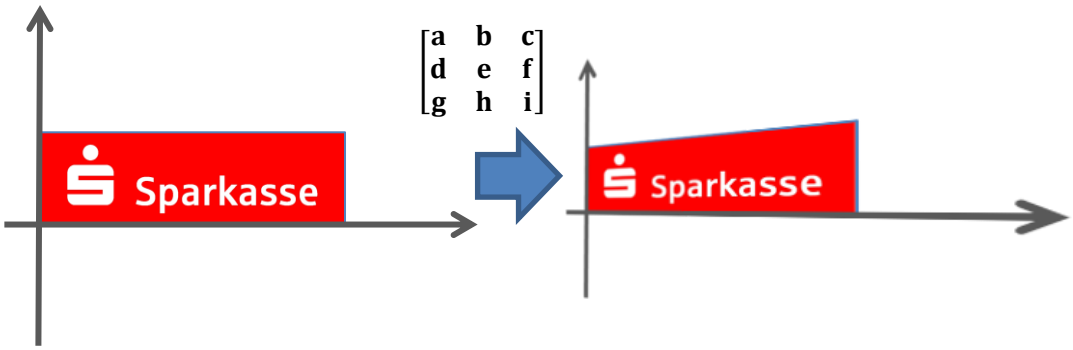
1.1.6 - Shear

Shearing can be considered as a translation along an axis that increases linearly with the other axis. It preserves neither the angles nor the distances and only the area of the logo is maintained. Usually all logos that are not placed in front of the camera viewpoint are subject to a shearing in the vertical axis or a perspective transform.



1.1.7 - Perspective

A perspective transform only preserves the lines of the object. It is the most common distortion that occurs in a sport video although it can usually be approximated by an affine transform. Its matrix has 8 degrees of freedom so it can account for any of the previous transformations.



1.1.8 - Compositions

Complex changes can be achieved through composition of the previous transformations. For example a translation can be done with several reflexions around different coordinate axis or a rotation can be performed with three shear mappings. All of them are grouped into families depending on their properties that preserve:

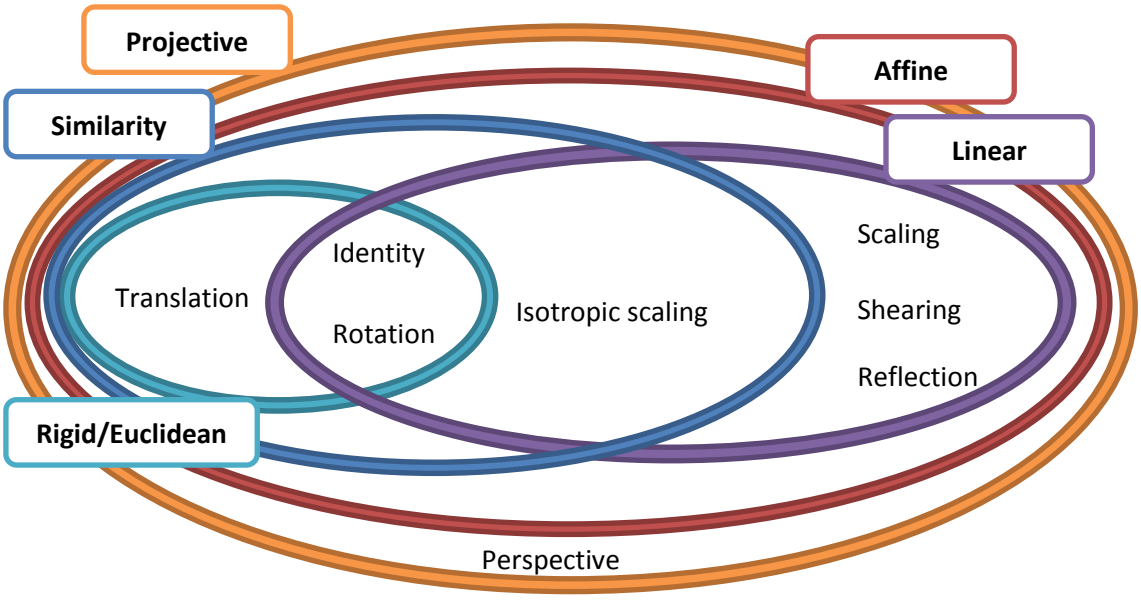


Figure 3 - Transformation families

The type of transformation determines the recognition of a logo. It is easier to deal with rigid and similarity transforms rather than linear, affine or perspective ones due to the conservation of the angle distribution.

1.2 - Occlusions, noise and blur

Some billboards might be partially occluded by people or objects. Those logos which are behind small obstacles should be recognized but it will not be possible to detect others with little visible parts. For example, in a football match the goal typically covers some of them; therefore, in order to handle these situations, local characterizations of the searched objects probably gives better results than a global feature.

Other distortions that can be expected from the camera are motion blur and noise. Motion blur usually occurs in sports events when the camera is used to follow the players or the ball and noise can be produced depending on the quality of the recording devices. This results in frames with varying degrees of sharpness and thus, different discriminative capabilities of logo detection methods.

Figure 4 - Examples of an affine transform, occlusions and motion blur

2 - State of the Art

Recognizing a logo into a frame is a specialization of a more general task: recognizing a pattern in a scene. This is useful not only for object detection but also for camera calibration or 3D reconstruction. A pattern extracted from an image usually describes a part of it in such a way that it is distinctive enough but, at the same time, robust to noise, geometric deformations and illumination changes. Hence, correctly describing the appropriate locations is essential to find correspondences between two images and detect objects without false alarms.

There has been a lot of research since Harris (Harris & Stephens, 1988) improved Moravec (Moravec, 1980) proposal to use the corners and the edges of a picture to track the objects in it. Schmid and Mor (Schmid & Mohr, 1997) addressed the problem of rotation variance and worked with grey-level invariance to characterize points. David Lowe perfected the interest point description with the Scale-Invariance Feature Transform (Lowe, 2004) that adds not only scale invariance and improves the robustness to 3D changes. There has been further research using the SIFT keypoints to develop PCA-SIFT, which aims to reduce the dimensionality but it is less distinctive, or SURF, which uses the Haar wavelet transform. All of them share the same stages for object recognition:

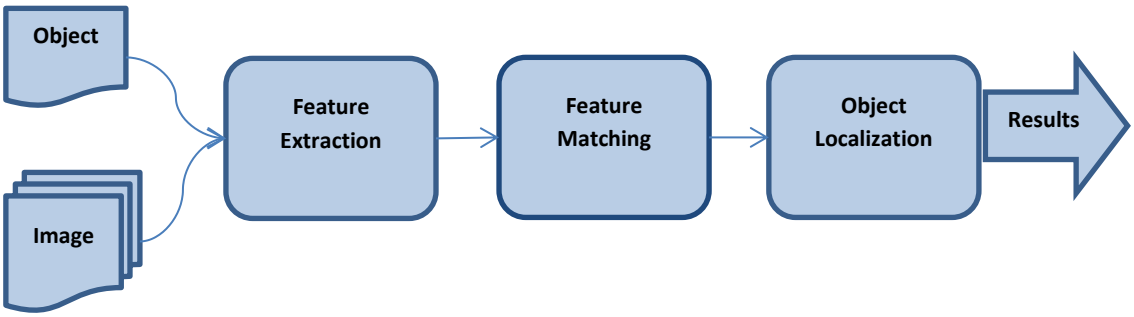


Figure 5 - State of the Art flow chart

In the first stage, the feature extraction, interest points are detected for a query logo and a frame in which the logo is to be recognized. After that, the correspondences are searched between the interest point descriptors. The way to compare two features and decide if they belong to the same object depends on the type of descriptor, for example, the SIFT features are compared using the Euclidean distance. Finally, a decision whether the object is present in a image or not and its localization is made depending on the number of matches found.

3 - Previously applied solution

The first solution applied in the Heinrich Hertz Institute by another student to detect and recognize logos was based on "Trademark Matching and Retrieval in Sports Video Databases" (Bagdanov et al., 2007). The authors from this publication extracted the SIFT features but their determination whether the logo was present or not was different from (Lowe, 2004). The next block diagram represents this approach (the broken line squares represents the blocks from "State of the art"):

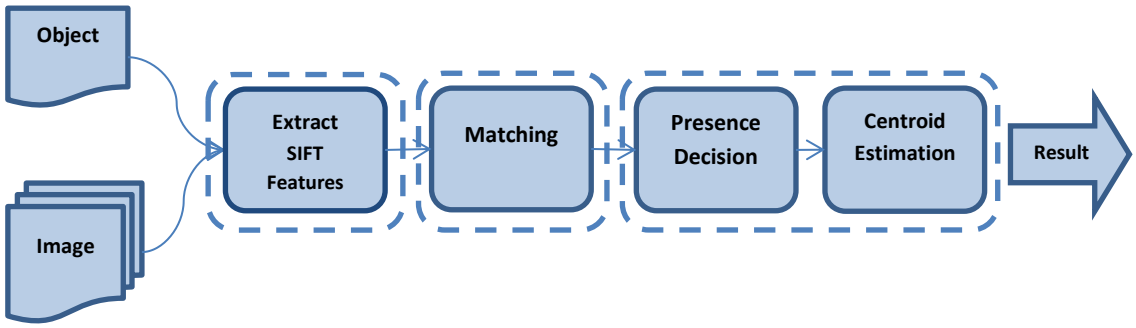


Figure 6 - Previously applied solution flow chart

3.1 - Feature extraction: Scale-Invariant Feature Transform

The local features used were the SIFT Keypoints. The SIFT Keypoints describe specific points of the image with their local gradients and its most important advantage is their high distinctiveness and their scale and rotation invariance as it has been explained. The extraction process can be summarized in four stages:

3.1.1 - Scale-Space extrema detection

The first phase searches for those locations in the image which are invariant to scale changes to be considered as candidates.

David Lowe proposed that the most efficient way to detect stable coordinates is looking for the extrema in the difference-of-Gaussians function convolved with the image. If the scale space of an image is defined as the convolution of the image with a variable-scaled Gaussian:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

where $I(x, y)$ is the image, * is the convolution and:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

The difference-of-Gaussians function convolved with the image can be computed from the difference of two nearby scales:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma)$$

In addition, Mikolajczyk (Mikolajczyk, 2002) found that the most stable keypoints are in the maxima and the minima of $\sigma^2\nabla^2 G$, which is related to the difference-of-Gaussians function according to the heat diffusion equation:

$$\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$$

Hence, in order to find the most stable keypoints, the author divides each octave into s intervals doubling σ each time, $k = 2^{1/s}$ (in the tests they show than the best sampling rate is 3 scales per octave) and computes the difference-of-Gaussians at each size (octave). Then, the image is resampled and this process is performed again as it can be seen in Figure 7:

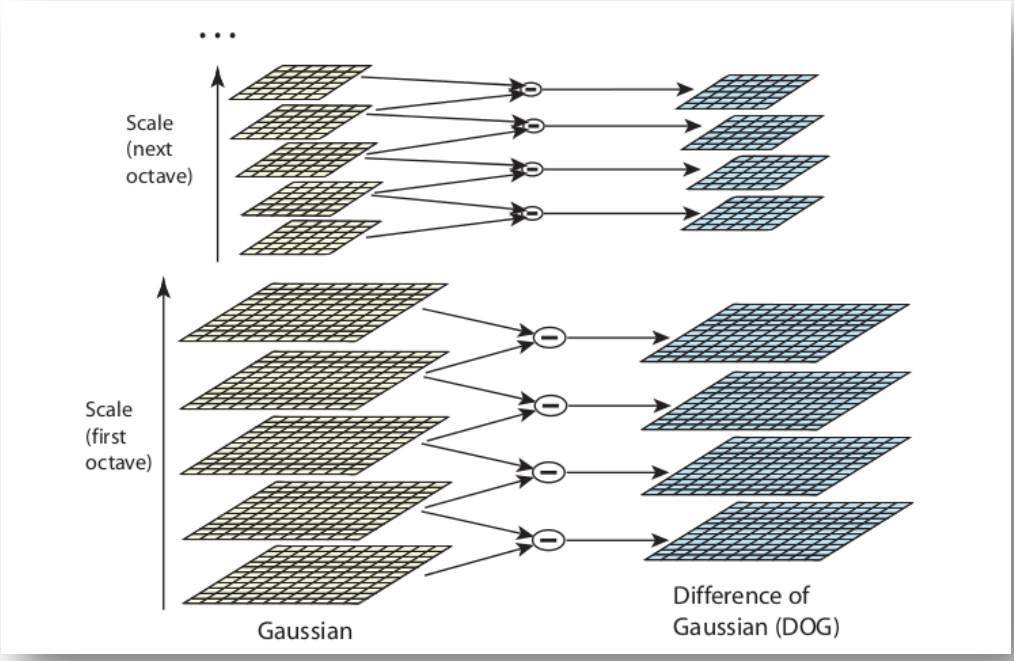


Figure 7 - Difference of Gaussian filtered images of different octaves

Finally, each sample point is compared with its neighbours in the same image of that scale and in the scale above and below and it is selected if it is larger or smaller than all of them as it is shown in Fig 8:

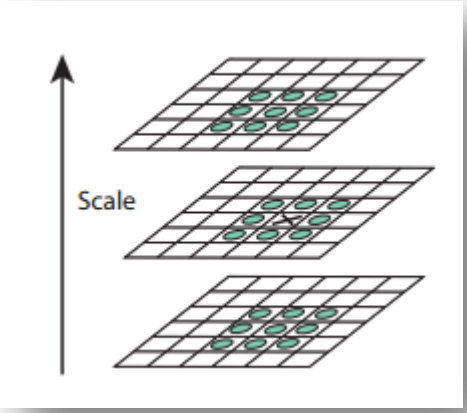


Figure 8 - Searching the maxima and minima

3.1.2 - *Keypoint Localization and stability criteria*

The next step is to improve the localization of the extrema. This process is done by fitting a quadratic function to every candidate:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathbf{T}}\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

Deriving it and setting it to zero. This gives:

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}}$$

where D and its derivatives are evaluated at the sample point. Here $\hat{\mathbf{x}} = (x, y, \sigma)$ is the final offset to be added to the point.

There are two stability criteria that the points must fulfil not to being rejected:

1. The point can't have a low contrast so:

$$|D(\hat{\mathbf{x}})| = \left| D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\hat{\mathbf{x}} \right| > 0.03$$

2. The points must have defined peaks. This condition is related to the ratio of the principal curvature along an edge to the curvature in the perpendicular direction. The curvatures are contained in the Hessian matrix:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix}$$

As the principal curvatures are proportional to the eigenvalues of **H,** and they can be easily computed using the trace and the determinant of the matrix, the condition to accept a point where α and β are the eigenvalues of **H** is:

$$\frac{Tr(H)^2}{Det(H)} = \frac{(\alpha+\beta)}{\alpha\beta} < \frac{(r+1)^2}{r}$$

for a certain *r*. David Lowe recommends setting $r = \frac{\alpha}{\beta} = 10$ as a correct threshold.

In the next two images from our database (Figures 9 and 10) the previous process was performed and the detected interest points are drawn in red:

Figure 9 - Logo Keypoint detect

Figure 10 - Keypoint detection in a frame from the Champions League 2010-2011

3.1.3 - Orientation assignment

In order to provide the descriptor with rotation invariance, an orientation must be assigned to each keypoint. For this task, a gradient orientation and a magnitude are computed within the region around a keypoint in the scale where it was found:

$$m(x,y) = \sqrt{\big(L(x+1,y) - L(x-1,y)\big)^2 + \big(L(x,y+1) - L(x,y-1)\big)^2}$$

$$\theta(x,y) = tan^{-1}\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}$$

All these gradient orientations are added to an orientation histogram after being weighted by its magnitude and by a Gaussian circular window with σ equals to 1.5 times that of the scale of the keypoint. The highest peak in the histogram is detected and a parabola is fitted to the closest values to find the final orientation angle. Other peaks in the histogram (with at least the 80% of height of the main one) may create other keypoints.

3.1.4 - *The image descriptor*

Once the keypoint locations are found a signature is created to describe it. In our case, this will allow us to find correspondences between two images so we can find an object (a logo) into a scene (a frame).

The interest point is described by their local gradients and this idea is based on a biological model of the primary visual cortex. More specifically, the descriptor is a vector of 128 elements formed by sixteen (4x4) histograms with 8 bins each one and this histograms group the gradients of the nearby regions rotated according to its main orientation. To avoid sudden changes in the descriptor with small changes in the position of the window, these gradients are weighted by their magnitude and by a Gaussian centred in the keypoint. This process can be observed in Figure 11.

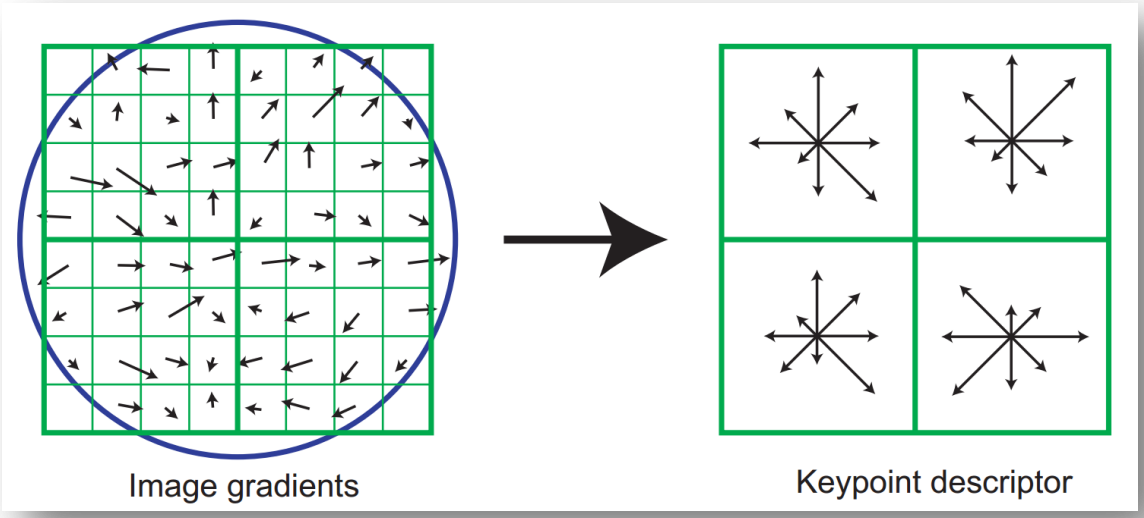


Figure 11 - Example of descriptor computation using only 2x2 instead of 4x4 histograms

Although this descriptor is robust to small affine changes it is not fully invariant to large ones, Pritchard and Heidrich suggested in (Pritchard & Heidrich, 2003) that the best approach to deal with these distortions is to generate SIFT features from a synthetic affine transformed object. This strategy and its results are explained in the optimizations section.

3.2 - Feature Matching

After the keypoints from the logo and the frame are found and their descriptors are extracted, the similarity between them is computed.

The Euclidean distance is used to evaluate the similarity between two descriptors. Some descriptors are more discriminative than others. Hence, a global threshold does not perform very well. In (Bagdanov et al., 2007) they use a ratio in order to accept or reject a keypoint; a feature is accepted if:

$$\frac{NN_1}{NN_2} = \frac{\min_q ||KP_l - F_f^q||}{\min_{q \neq NN_1} ||KP_l - F_f^q||} < matching\ threshold$$

where KP_l represents a logo feature descriptor and F_f^q any frame descriptor. This expression means than the ratio of distances to the first and the second nearest neighbours is less than a certain threshold (they used 0.8); *i.e.* for a correct match, the nearest feature in the frame must be significantly closer compared to the second nearest one. It has been demonstrated that this condition works very well even with large databases because it successfully rejects 90% of the incorrect features while it discards less than 5% of the correct ones.

The method used to find the nearest neighbours in (Lowe, 2004) is Best-Bin-First. This is an approximate nearest neighbour technique that provides a usual speedup of two orders of magnitude compared to a linear search thanks to a modified search version of kd-trees. Nevertheless, in the (Bagdanov et al., 2007) implementation, the FLANN (Lowe & Muja, 2009) library was used. This library has different ways to perform an approximate search (KD-trees and hierarchical k-means trees) and it chooses the best one depending on the case.

It can be seen in the next example the results of matching the logo features to the frame features, where the incorrect keypoints are drawn in red circles and the correct matches in green lines:

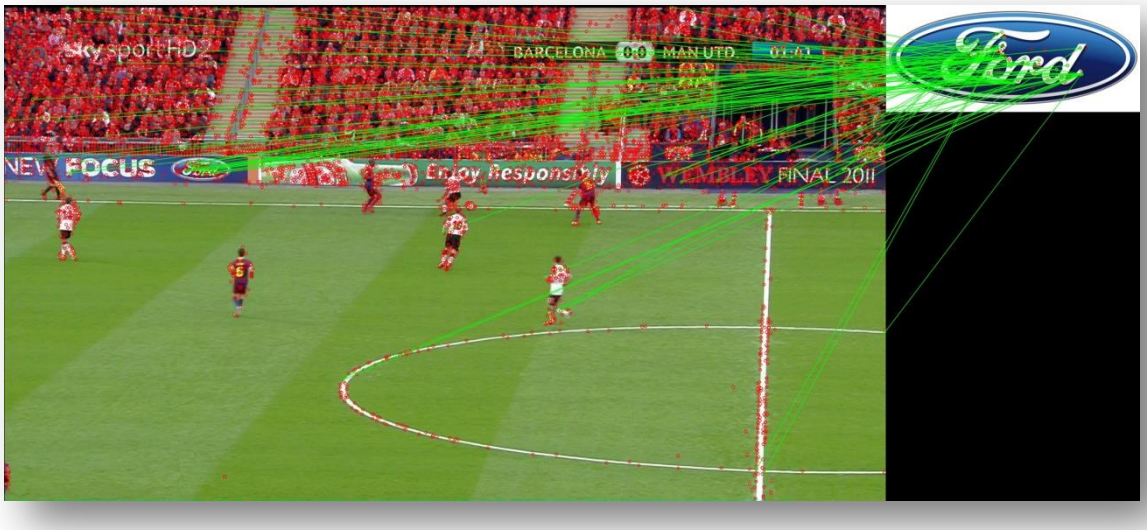


Figure 12 - Keypoint matching in the previous solution

3.3 - Logo Localization

Once the correspondences between SIFT keypoints are found, the last phase determines if the object is present and where it is located in the frame:

1. The presence decision is done regarding how many of the local features extracted from the logo are matched; if more than 20-25% of logo keypoints are found (depending on the trademark) it is present, otherwise it is not.
2. To find the logo position, the centroid of the cloud of features is calculated iteratively solving:

$$\sum_{t=1}^{n} \varphi(x_i; \mu_x) = 0 \qquad \sum_{t=1}^{n} \varphi(y_i; \mu_y) = 0$$

where x_i and y_i are the coordinates of the matched keypoints, μ_x and μ_y are the centroid coordinates and φ is the Tuckey biweight function:

$$\varphi(x; m) = \begin{cases} (x-m)\left(1 - \frac{(x-m)^2}{c^2}\right)^2 & if\ |x-m| < c \\ 0 & otherwise \end{cases}$$

$$c = median(|x_i - median(x_i)|)$$

After that, the limits of a bounding box are defined by the furthest matched keypoints from the centroid whose distance is smaller than *c,* as it can be appreciated in the next example:

Figure 13 - Correct logo localization using the Tuckey biweigh function

4 - Issues and challenges

The explained approach has several advantages: it is simple and it works correctly if there is only one instance of each logo per frame. Nevertheless some problems arise if the same logo appears multiple times.

One of the problems comes from the matching block. When the logo is present multiple times the same (or a very similar) descriptor is repeated in the frame, therefore, the distances to the first and the second nearest neighbors will be roughly equal. The ratio between those distances will be close to one, so none of these logo points will be matched despite having a correspondence in the scene and the presence decision will fail:

Figure 14 - Matching problem in the previous existing solution

Additionally, even when enough matched features are found to pass the presence test, the probability of being spread out among the instances is high and consequently the centroid and the boundaries won't be correct as it is shown in Figure 13. An example of this situation is shown below where most of the puma keypoints are not matched (in red) and the boundaries exclude the last appearance on the right:

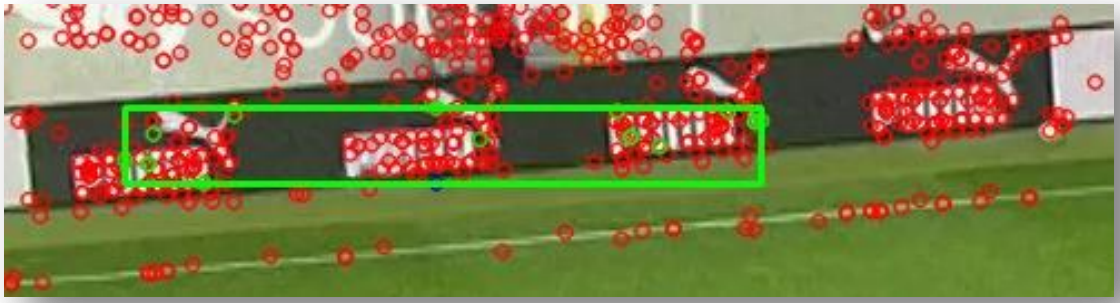


Figure 15 - Localization failure in the previous existing solution

Furthermore, this approach does not account for any change in the logo angles. All the bounding boxes are designed to be right-angled, so most of the recognized logos do not fit very well in them. It can be interesting to adapt the bounding boxes because that would make possible to know how many pixels of the scene belong to the logo, which may be interesting for commercial purposes and statistics.

The different methods to deal with these situations are explained in the next section.

Chapter II - Developed approaches

1 - Overview

In order to address the problems described in the previous section the feature matching block (in Figure 6) had to be changed In order to enable the matching of multiple instances of the same logo keypoint. In addition, the localization block had to also be modified to correctly group the features belonging to the same logo in cases where a logo occurs more than once in a frame.

The first problem is solved by changing the matching condition in I.3.2 and the second problem is solved by using the Hough Accumulator described in (Lowe, 2004). The original solution proposed by Lowe is also interesting because it can set a baseline to compare the performance of further approaches. The next image shows its flow chart:

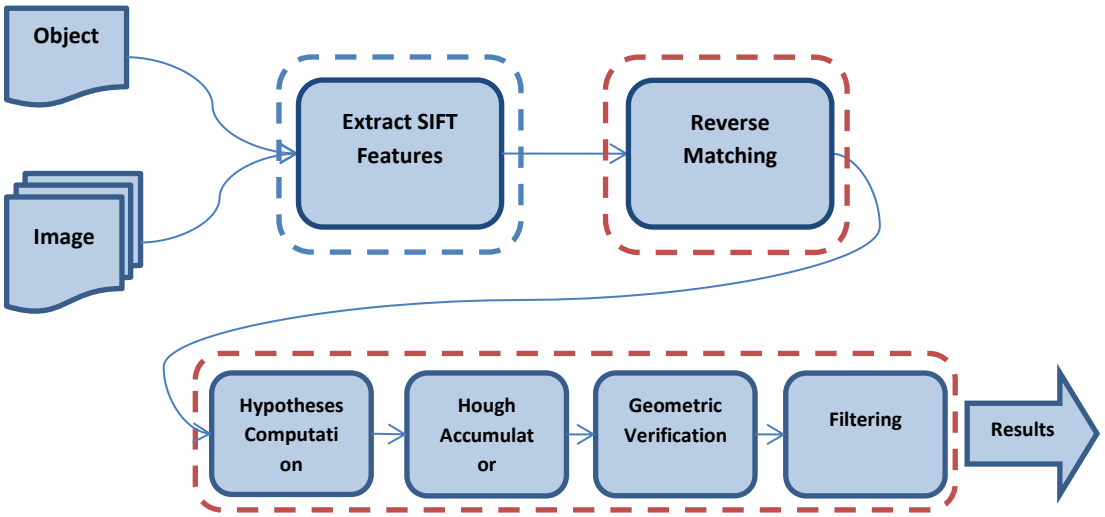


Figure 16 - Original approach flow chart

2 - Extract SIFT Features

There are several SIFT implementations available on the Internet with some differences between them. The main one is provided by David Lowe at his website (http://www.cs.ubc.ca/~lowe/keypoints/). It is however only released as a binary file, so other universities and companies have developed their own versions. The most popular implementations are found in the computer vision libraries *OpenCV* and *VLFeat*.

2.1 - SIFT implementations: OpenCV

OpenCV is a computer vision library developed by Intel in 1999 and is now supported by the company Willow Garage. It is a cross-platform project under the BSD license which provides a C++ framework to work with the most common algorithms used in computer vision and image processing.

Almost all tests in this project were performed using OpenCV version 2.3. The author of the SIFT implementation, Rob Hess, claims that the resulting keypoints and descriptors are very similar to David Lowe's version; for instance, trying to compute the matches in altered images from the Caltech-256 dataset ends up with more or less the same results (Hess, 2010): Rob Hess's version matched 858 of 3705 keypoints (23.2%) and David Lowe's version 1087 of 4635 (23.5%). However, it is slightly slower than the original version from Lowe.

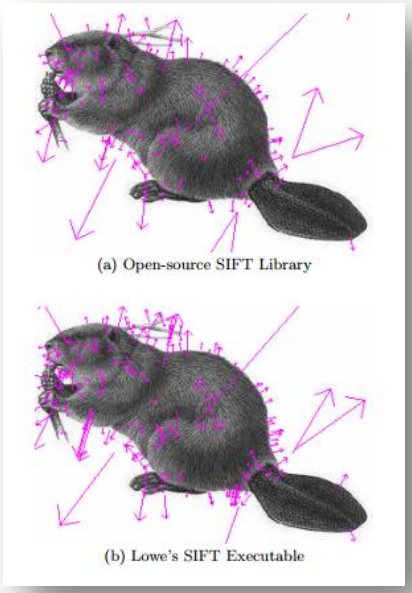


Figure 17 - Comparison between Lowe's and Hess's SIFT (Hess, 2010)

2.2 - SIFT implementations: VLFeat and SIFTGPU.

VLFeat (Vedaldi & Fulkerson, 2007) is a cross-platform open-source collection of vision algorithms written in C and created by Andrea Vedaldi and Brian Fulkerson in 2007. Nowadays it is supported by a large number of computer vision groups and researchers like Krystian Mikolajczyk, Andrew Zisserman and Cordelia Schmid, who were also acknowledged in the SIFT research.

Among other advantages, the VLFeat SIFT implementation has become very popular due to its similarity to Lowe's version. For example, in Figure 18 89% of the VLFeat keypoints match to Lowe's keypoints with at least 0,01 pixels of precision and its descriptors are also similar: 37% of them match with less than 5% of difference, 35% with less than 10% of difference and 26% with less than 20% of difference:

Figure 18 - VLFeat keypoints in blue superposed to D. Lowe's keypoints in red. (Vedaldi & Fulkerson, 2007)

Although VLFeat was not directly used in this project, the GPU computation was developed linking to the library SIFTGPU. SIFTGPU (Wu, 2007) is inspired by VLFeat and (Sinha, Frahm, & Pollefeys, 2006). The usage of the graphic card processor with GLSL or CUDA (Compute Unified Device Architecture) speeds up the computation up to 0,1 seconds per frame (in a nVidia 8800GTX). The tests performed with SIFTGPU in the results section show how the runtime improves with this technology.

3 - Feature Matching – Reverse match

To be able to match each feature from the logo more than once, the conditions to accept a keypoint belonging to a frame are done in the opposite direction. Instead of looking for the nearest neighbor of the logo descriptors in the frame descriptors, the nearest neighbors from the frame descriptors are looked into the logo descriptors, and the ratio is calculated as follows:

$$\frac{NN_1}{NN_2} = \frac{\min_q ||KP_f - F_l^q||}{\min_{q \neq NN_1} ||KP_f - F_l^q||} < matching\ threshold$$

where KP_f represents the keypoints from the frame and F_l^q any logo keypoint. Incorrect positives can easily appear in high definition frames because there are more (between one and ten thousand) than in a logo (between fifty and four hundred), so the matching threshold is decreased from 0.8 in section I.3.2 to 0.7. Even with this new threshold some errors may happen but they will be removed in the next phase:

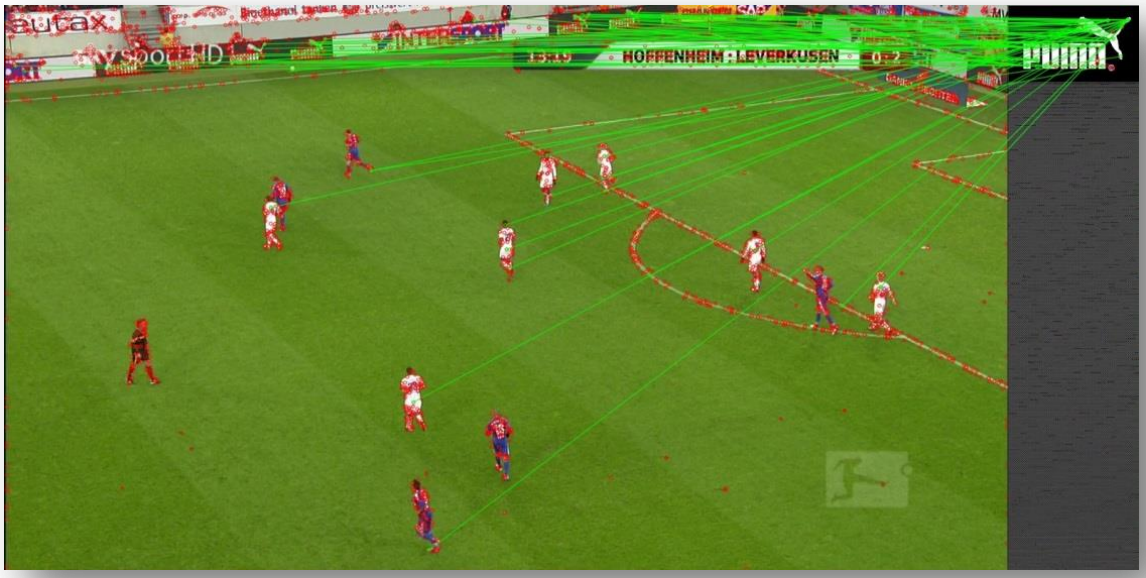


Figure 19 - Example of reverse matching

Among the different techniques to search nearest neighbors, the FLANN library was used. It has been demonstrated than other methods such as "Product quantization for nearest neighbor search" (Jégou, Douze, & Schmid, 2011) outperform the current approach regarding runtime for big databases if the data does not fit into memory, but that is not our case and the integration of FLANN into OpenCV was a big advantage.

4 - Object Localization

4.1 - Pose hypotheses computation

The first step in clustering the matched keypoints in the frame based on all their properties is to compute the pose hypothesis for each keypoint. These hypotheses, which can be computed

using the matched pair in the logo, define where and how the logo appears in the scene. The main goal here is not to fit a bounding box for the logo instance in the frame, but to find clusters of keypoints voting for the same pose. In such a case, the probability of interpreting an actual match as being a correct correspondence is much higher.

For a pose hypothesis, it is assumed that all the points from the logo have been subjected to an affine transformation that can be approximated by a similarity transform; *i.e.* a rotation, a scaling and a translation. An approximation to the rotation angle can be calculated using the main orientations: if the main orientation of the keypoint in the logo was α_l and the orientation of the corresponding matched point in the frame was α_f, the logo in the frame would have rotated by:

$$\theta = \alpha_f - \alpha_l$$

An approximation of the size of the logo in the frame is estimated using the octave where the keypoints were found and the original logo dimensions. Hence, if the logo keypoints in the λ_l octave have been extracted after reducing the logo image λ_l times, and their matches in the λ_f octave have been extracted after reducing the frame image λ_f times, the scale of the logo in the frame is:

$$2^{\lambda f} \cdot logo\ size\ in\ the\ frame = 2^{\lambda l} \cdot original\ logo\ size \rightarrow$$

$$\rightarrow logo\ size\ in\ the\ frame = \frac{2^{\lambda l}}{2^{\lambda f}} \cdot original\ logo\ size = 2^{\lambda l - \lambda f} \cdot original\ logo\ size$$
$$= 2^{s} \cdot original\ logo\ size$$

Finally, in order to get an approximation of the translation of the logo instance in the frame, the logo coordinates (y_{lk}, x_{lk}) are rotated and scaled and are subtracted from the keypoint coordinates (y_{fk}, x_{fk}) of the logo instance in the frame:

$$y_{fk} = (x_{lk} * \sin(\theta) + y_{lk} * \cos(\theta)) * 2^s + trans_y \rightarrow$$

$$trans_y = y_{fk} - (x_{lk} * \sin(\theta) + y_{lk} * \cos(\theta)) * 2^s$$

$$x_{fk} = trans_x + (x_{lk} * \cos(\theta) - y_{lk} * \sin(\theta)) * 2^s \rightarrow$$

$$trans_x = x_{fk} - (x_{lk} * \cos(\theta) - y_{lk} * \sin(\theta)) * 2^s$$

An example of this process can be seen in Figure 21 is drawn in the next frame. The matched features are drawn in green, the yellow lines represent the subtraction of the rotated and scaled logo coordinates and the purple circles the position of the translation of the hypotheses. As the coordinates in the original logo are referenced to their centers, the purple circles fall on the logo center in the frame. It should be also noted how the incorrect matches are spread around the scene while the correct matches get-together:

Figure 20 - Hypotheses Computation

4.2 - Feature Clustering: The Hough Accumulator

The goal of the Hough Accumulator is to detect the groups of features that vote for the same pose. This is achieved by accumulating the hypotheses in a four-dimensional histogram and retrieving all the poses which had received more than a certain number of votes. The process discards many of the incorrect features that have been wrongly matched in the previous phase because their hypotheses are scattered throughout the frame.

The greatest challenge in creating the accumulator is the mutual dependence between dimensions. The four bins for the keypoint parameters (scale, rotation angle and 2D translation) must be broad enough in order to deal with the large error bounds of the pose prediction. These errors in the translation dimension depend on the size of the object and the

scale dimension. Errors may appear because these four parameters only account for an Euclidean transformation and that is only an approximation either to a full 6 degree of freedom pose space of a 3D object or to a non-rigid alteration (for example a perspective transform). Therefore, in (Lowe, 2004) David Lowe suggests a bin size of 30 degrees for orientation, a factor of 2 for scale and 0.25 times the maximum projected training image dimension (using the predicted scale) for location.

In Figure 22 the location bin sizes have been drawn to illustrate their variations with the scale:

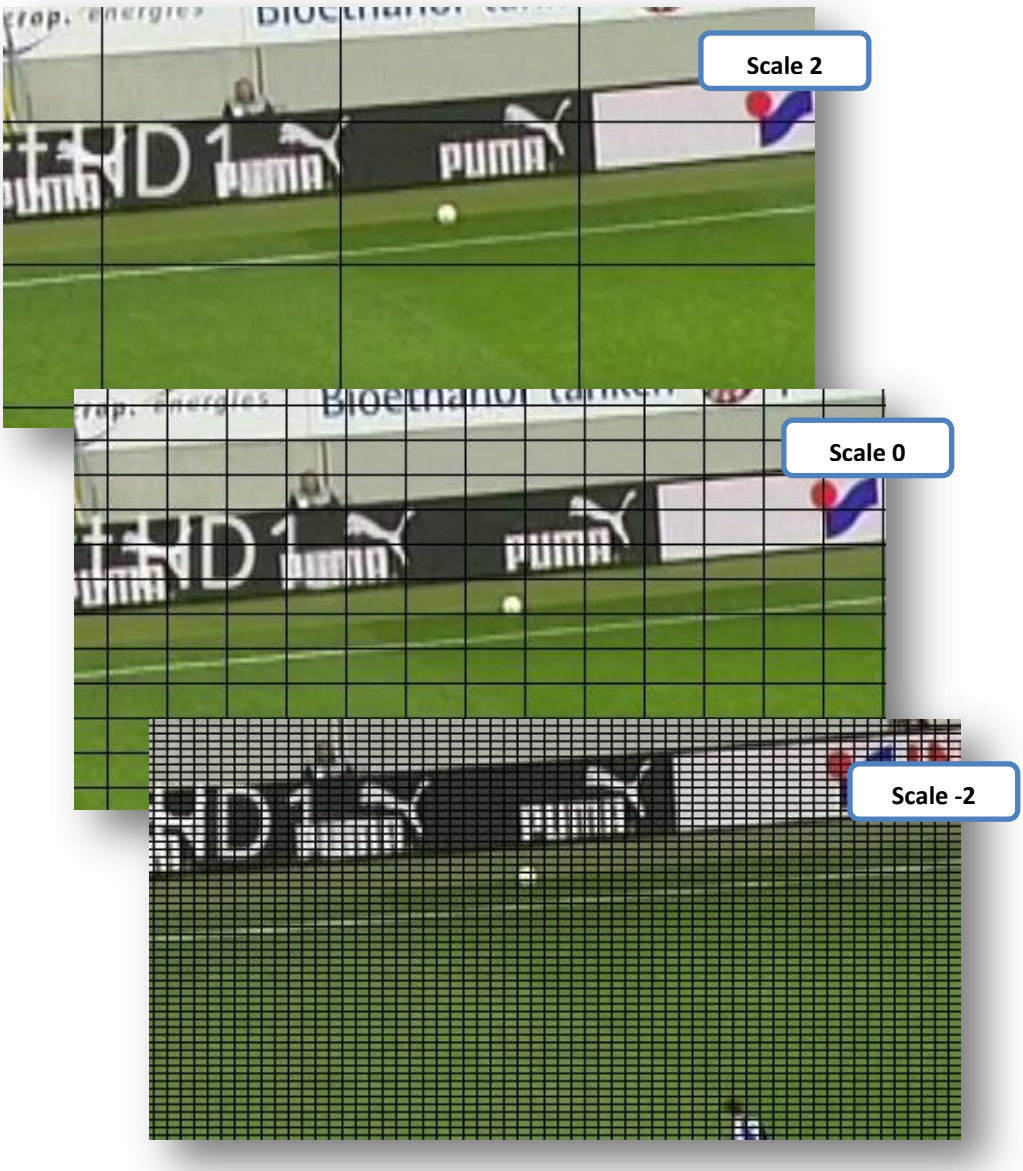


Figure 21 - Location bin sizes for different scale values

Once the accumulator is prepared, every matched keypoint votes for its pose hypothesis. To avoid the boundary effects, *i.e.* to split a group of features into two bins due to the position of the bin limits, each keypoint has to vote for the two closest bins in each dimension and that gives a total of 16 entries per feature. This multiple voting process for the location dimension can be observed in the Figure 23 where the bin sizes are drawn in black. The matches are drawn in green and the multiple votes in red (mainly accumulated over center of the logo):

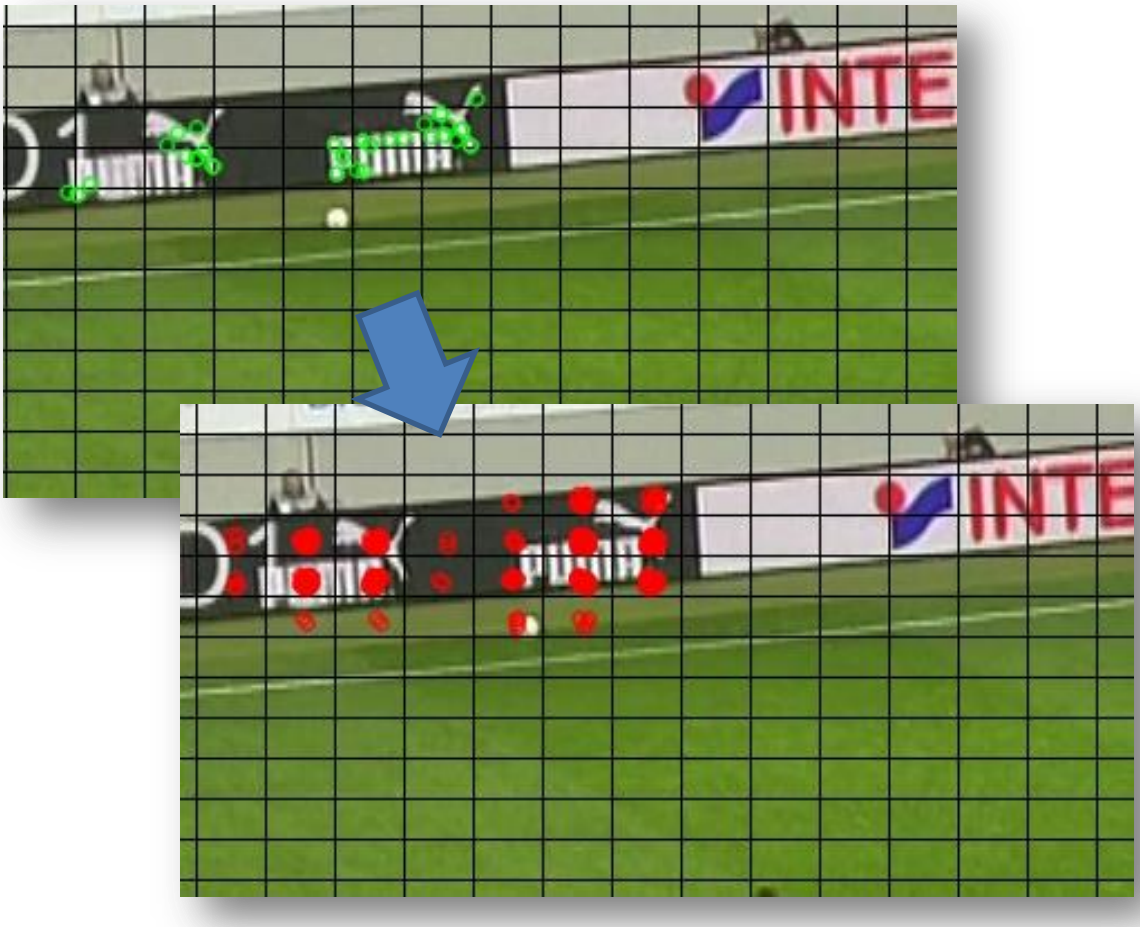


Figure 22 - Voting process in the Houg accumulator

Once the voting process is finished, those bins with more votes are retrieved. Due to the implementation of the multidimensional histogram in a one-dimensional array, all hypotheses with at least three votes are easily detected. After that, each bin is compared to their neighbours in the four dimensions and if it has more features voting for it than the others (it is a peak), it is marked as a candidate.

4.3 - Geometric Verification

A geometric verification is applied to all the candidate clusters to find the best affine parameters that convert the original logo into the one appearing in the frame. In our case modeling this distortion as an affine transform is enough because it can account for most of the transformations of a planar object. If no set of values exist, the hypothesis is discarded.

An affine transform from a point in the logo $[x_1\ y_1]^T$ to a point in the frame $[u_1\ v_1]^T$ can be written as:

$$\begin{bmatrix} u_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

where the m_i values represent a linear mapping and the t_i values model a translation. All the points that voted for that hypothesis can be inserted into a system to solve it. As there are at least three points, there is enough information to solve it:

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ & & \dots & \dots & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \\ \dots \end{bmatrix}$$

This linear system can be rewritten as:

$$\mathbf{A}\mathrm{x} = \mathrm{b}$$

Finally, in order to find the best parameters, a least-squares solution is performed:

$$\mathrm{x} = [\mathbf{A}^{\mathrm{T}}\mathbf{A}]^{-1}\mathbf{A}^{\mathrm{T}}\mathrm{b}$$

Once the system is solved, outliers can be discarded by their consistency with the system. If a point disagrees with more than half the error range used in the accumulator, which is the size of the bin, it is removed and the system is re-calculated. If fewer than 3 points remain after discarding outliers, the hypothesis is rejected; otherwise it will be subjected to a filtering step:

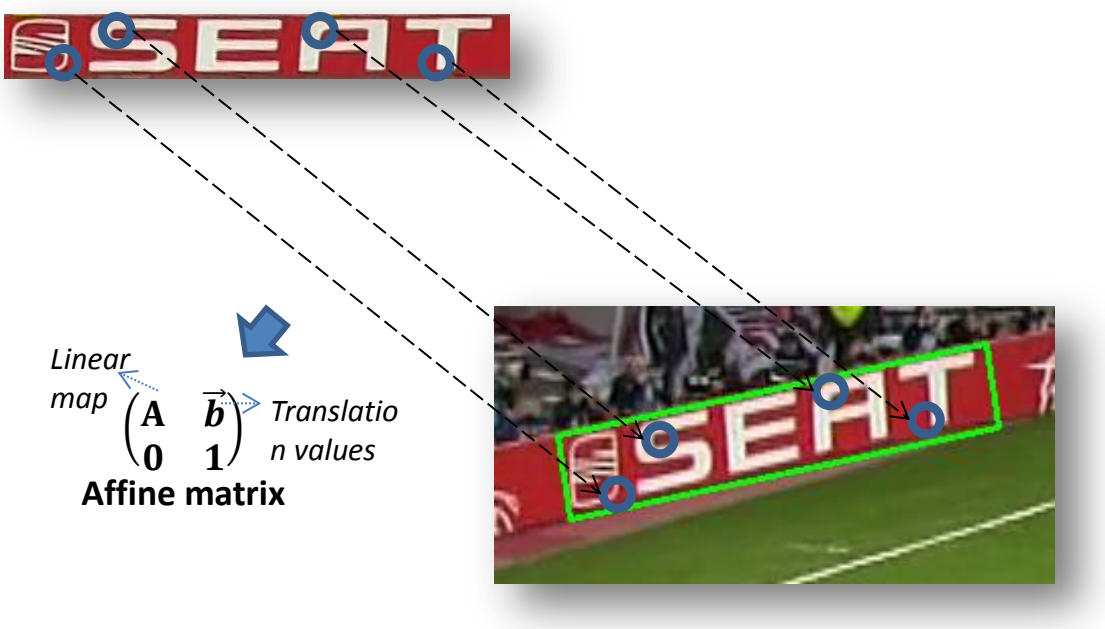


Figure 23 - The affine parameters map the original logo to the logo in the frame

4.4 - Filtering

The groups of features agreeing on a hypothesis with a feasible affine transform must possess a probability condition. The filtering process starts with a top-down matching to recover all the keypoints that agree with the affine parameters but were either not grouped or incorrectly discarded in the previous steps. Then, a probabilistic test is performed to check if there are enough matched features for the logo size. Finally a last algorithm checks if a logo was detected twice due to previous errors (and therefore the hypotheses are overlapped):

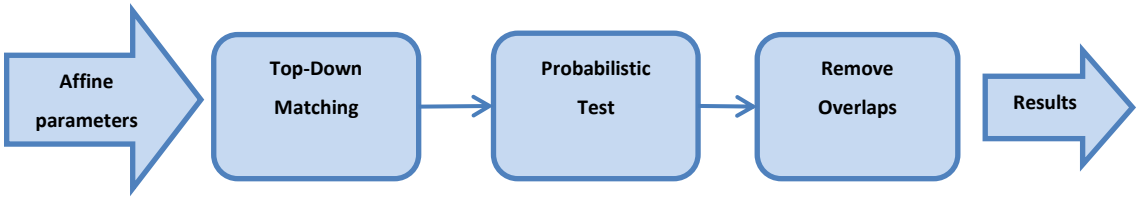


Figure 24 - Filtering flow chart

4.5 - Top-Down Matching

The top-down matching block gets back those matches that agreed with the hypothesis but which were lost or discarded in the clustering or in the geometric verification. This is done by projecting the logo keypoints in the frame and checking if the distance between them and the location of the matched keypoint is less than the bin size for that scale.

Although there are several reasons to miss a correct match, for instance a grouping error due to the similarity approximation, in our experiments this happened few times.

4.6 - Probabilistic test

The probability check makes a final decision to accept or reject a hypothesis regarding the number of matched features in the projected model size (the frame area covered by the logo).

In (Lowe, 2001) the authors propose to compute the probability of presence given the set of matched features, $P(m|f)$, using Bayes' Theorem and accept the model if it is above a threshold:

$$P(m|f) = \frac{P(f|m) \cdot P(m)}{P(f)} = \frac{p(f|m) \cdot P(m)}{P(f|m) \cdot P(m) + P(f|\neg m) \cdot P(\neg m)} > p.t.$$

Where p.t. is the probability threshold and $P(\neg m)$ can be approximated by 1 because there is low prior probability of a model appearing at a particular pose and $P(f|m)$ also by 1, as at least a certain number of features will be detected if the model is present in the frame. Therefore, the expression depends on $P(m)$ and the probability that the matched features would arise by accident if the model is not present, $P(f|\neg m)$:

$$P(m|f) \approx \frac{P(m)}{P(m) + P(f|\neg m)} > probability\ threshold$$

If *n* is the number of image keypoints within the projected logo and *p* is the probability of matching a single image feature to the current model pose, $P(f|\neg m)$ can be computed using a cumulative binomial distribution (the probability of an event with probability p occurring at least k times out of n trials) as:

$$P(f|\neg m) = \sum_{j=k}^{n} \binom{n}{j} p^j (1-p)^{n-j}$$

David Lowe in (Lowe, 2001) suggests assigning the probability that a single feature match is correct to $P(m)$, that is the ratio of correct features to all matched features, because it can be considered that one matching feature is used to determine the initial pose hypothesis and the others are used for verification. He also recommends setting the probability threshold to 0.95, which is confirmed in our cross-validation dataset too. Thanks to this test, some errors that arise from local mismatches are avoided as it can be seen in Figure 26:

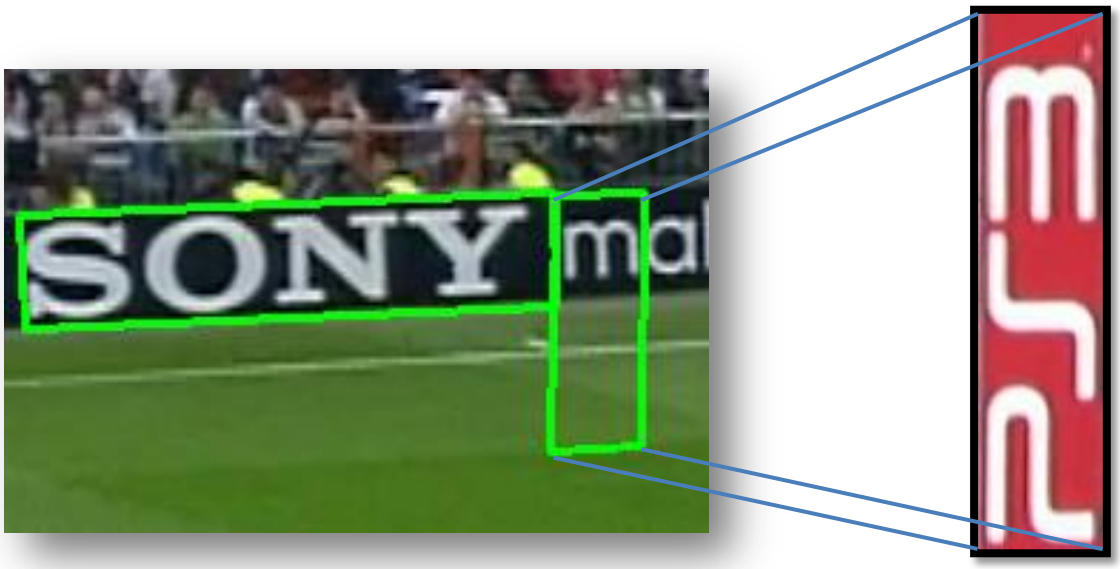


Figure 25 - Example of local coincidences

4.7 - Remove Overlaps

On few occasions, overlaps may happen due to the Hough Accumulator. In the case of finding two consecutive hypotheses (bins) with the same amount of votes, the peak detector retrieves both of them because there is not enough information to choose one over the other. This last step looks for these situations and removes the repeated detections (see Figure 27):

Figure 26 - Removing overlaps

This process reviews all the location hypotheses and removes those detections with more than a percentage of the overlapped pixels. In our experiments, detections were discarded if they had more than 30% the overlapped area. Small overlapping zones are allowed because in football matches multiple instances of some logos appear repeatedly aligned in the frame and little scale errors may occur as in Figure 28:

Figure 27 – Multiple logo instances aligned next to one another

5 - Researched approach: Graph Clustering

5.1 - Introduction

The graph clustering is a new approach developed during this thesis to group the matched features that outperforms the precision of the Hough accumulator. This is achieved by avoiding each feature to vote for several hypotheses as in the previous method. Hence, the system's block diagram is shown in Figure 28:

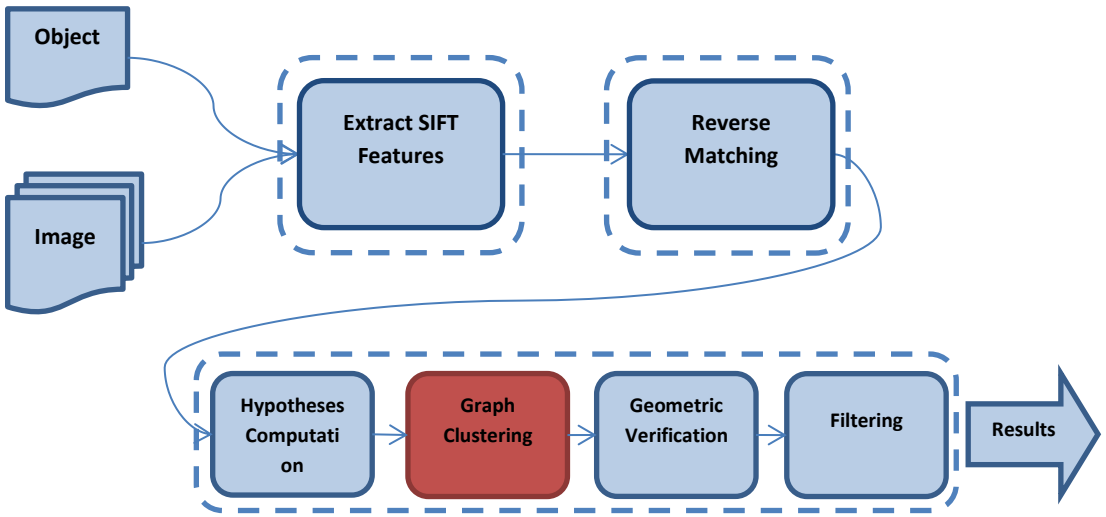


Figure 28 - Graph clustering block diagram

The graph clustering can be divided into three stages: the graph creation, where the matches that vote for the same hypothesis are linked, the searching stage, where the groups of linked components are found, and the pruning stage, where the incorrectly linked matches are discarded.

5.2 - Graph creation

After computing the pose hypothesis of each feature in the same way as in the Hough accumulator, an undirected graph is created making edges between them regarding their parameters. However, some errors can occur in the SIFT detection process so, similarly to the method described in section II.4.2; therefore, this edge would be created only if there is less than a difference of 15 degrees in the estimated rotations, less than one octave of difference in the estimated scale and less than a certain distance between the estimated translations between hypotheses.

The maximum allowed error for the estimated translation to link two hypotheses depends on the rotation error and the scale parameters:

$$Max\ linking\ distance = \sin\left(\frac{\alpha}{2}\right) * \sqrt{dim_x^2 + dim_y^2} * 2^s$$

where α is the maximum rotation error; dim_x and dim_y are the dimensions of the logo and s is the number of augmentations of the hypothesis (if it is negative it means that there have been reductions).

The maximum linking distance can be derived by computing the translation coordinates. As it has been shown in section II.4.1, these values are given by:

$$trans_y = y_{fk} - (x_{lk} * \sin(\theta) + y_{lk} * \cos(\theta)) * 2^s$$

$$trans_x = x_{fk} - (x_{lk} * \cos(\theta) - y_{lk} * \sin(\theta)) * 2^s$$

where y_{fk} and x_{fk} are the keypoint location in the frame (referred to upper left corner of the frame), y_{lk} and x_{ck} are the matched location in the original logo (referred to the upper left corner of the logo), θ is the rotation angle in degrees, and s is the scale. Usually the scale is correctly estimated and most of the errors come from the rotation hypothesis. If there was an error α in the angle estimation, the translation would be:

$$trans'_y = y_{fk} - (x_{lk} * \sin(\theta + \alpha) + y_{lk} * \cos(\theta + \alpha)) * 2^s$$

$$trans'_x = x_{fk} - (x_{lk} * \cos(\theta + \alpha) - y_{lk} * \sin(\theta + \alpha)) * 2^s$$

and therefore, the distance between the incorrect value and the original point is as follows:

$$error_y = trans_y - trans'_y =$$

$$(x_{lk} * \sin(\theta) + y_{lk} * \cos(\theta)) * 2^s - (x_{lk} * \sin(\theta + \alpha) + y_{lk} * \cos(\theta + \alpha)) * 2^s$$

$$\rightarrow \varphi = \theta + \frac{\alpha}{2} \rightarrow$$

$$\left(x_{lk} * \sin\left(\varphi - \frac{\alpha}{2}\right) + y_{lk} * \cos\left(\varphi - \frac{\alpha}{2}\right)\right) * 2^s - \left(x_{lk} * \sin\left(\varphi + \frac{\alpha}{2}\right) + y_{lk} * \cos\left(\varphi + \frac{\alpha}{2}\right)\right)$$

$$* 2^s = \left(-2 * x_{lk} * \cos(\varphi) * \sin\left(\frac{\alpha}{2}\right) + 2 * y_{lk} * \sin(\varphi) * \sin\left(\frac{\alpha}{2}\right)\right) * 2^s =$$

$$= 2 * \sin\left(\frac{\alpha}{2}\right) * (y_{lk} * \sin(\varphi) - x_{lk} * \cos(\varphi))$$

analogously:

$$error_x = trans_x - trans'_x = 2 * \sin\left(\frac{\alpha}{2}\right) * (x_{lk} * \sin(\varphi) + y_{lk} * \cos(\varphi))$$

Hence, the squared error is as follows:

$$Total\ Error = \sqrt{error_y^2 + error_x^2} = 2 * \sin\left(\frac{\alpha}{2}\right) * \sqrt{x_{lk}^2 + y_{lk}^2} * 2^s$$

For a given α, the maximum error occurs for a point located in the opposite corner of the reference, for instance, if the reference is set in the top left corner, the maximum occurs in a point in the bottom right corner. In this situation, x_{lk} and y_{lk} are the scaled dimensions of the object, dim_x and dim_y. If the reference is set in the middle of the image, the maximum error is divided by two, giving the previous *Max linking distance* equation.

Figure 29 visually describes this process; the hypotheses computation for each match with an arbitrary error and the graph creation between close matches:

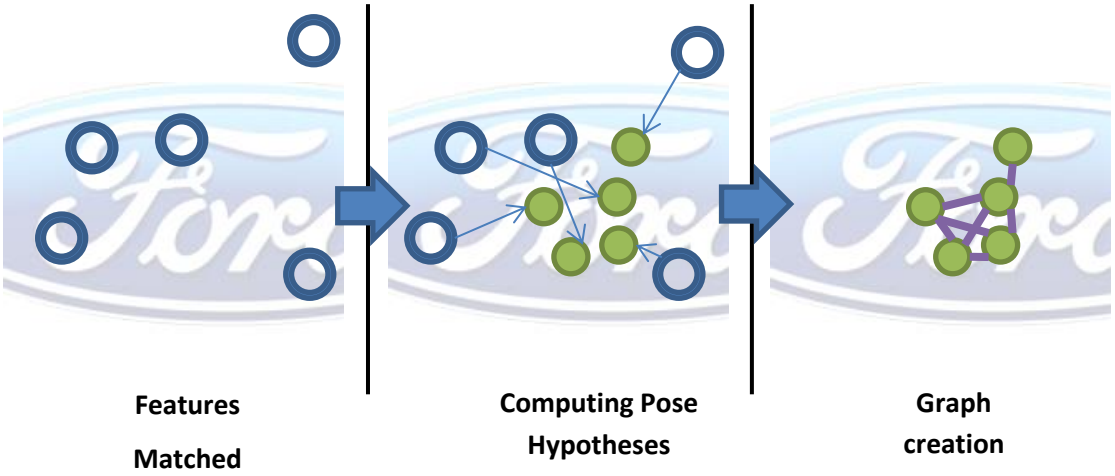


Figure 29 Linking (Graph version)

Figure 30 shows a real example of the linked hypotheses of a Ford logo in a frame.

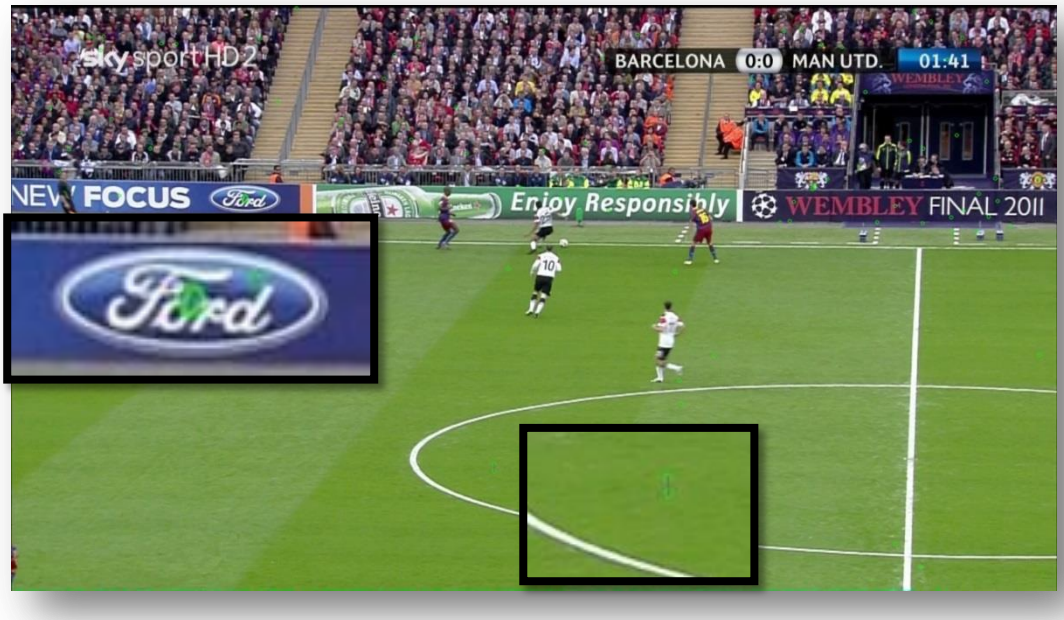


Figure 30 - Linking step example (graph version)

5.3 - Searching groups: Connected components and DFS

The groups of features voting for the similar hypothesis have to be identified taking into consideration the described errors. In other words, the clusters with linked vertices have to be retrieved.

According to David Lowe in (Lowe, 2004), a reliable recognition is possible with as few as three features, thus groups of at least three nodes are searched. Clusters of features voting for the same pose will probably be completely connected vertices, *i.e.* all its nodes have edges between them and that is called *clique*. It is necessary to find them because other edges can be created with incorrect matches and they must be excluded. The main problem is that finding the *cliques* in a graph is one of the Karp's 21 NP-complete problems. A possible solution to tackle this problem is given by the Bron-Kerbosch algorithm, but the worst case still runs in exponential time. After some more research, we found that creating a directed graph, finding the connected components and trying to remove the outliers lead to similar results without a high computation cost.

5.3.1 - *Clique versus Connected Components*

The main advantage of a *clique* is that there won't be any pair of matches whose hypotheses differ by more than the allowed error. This condition may not be fulfilled if just the connected components are retrieved, as can be observed in Fig 31.

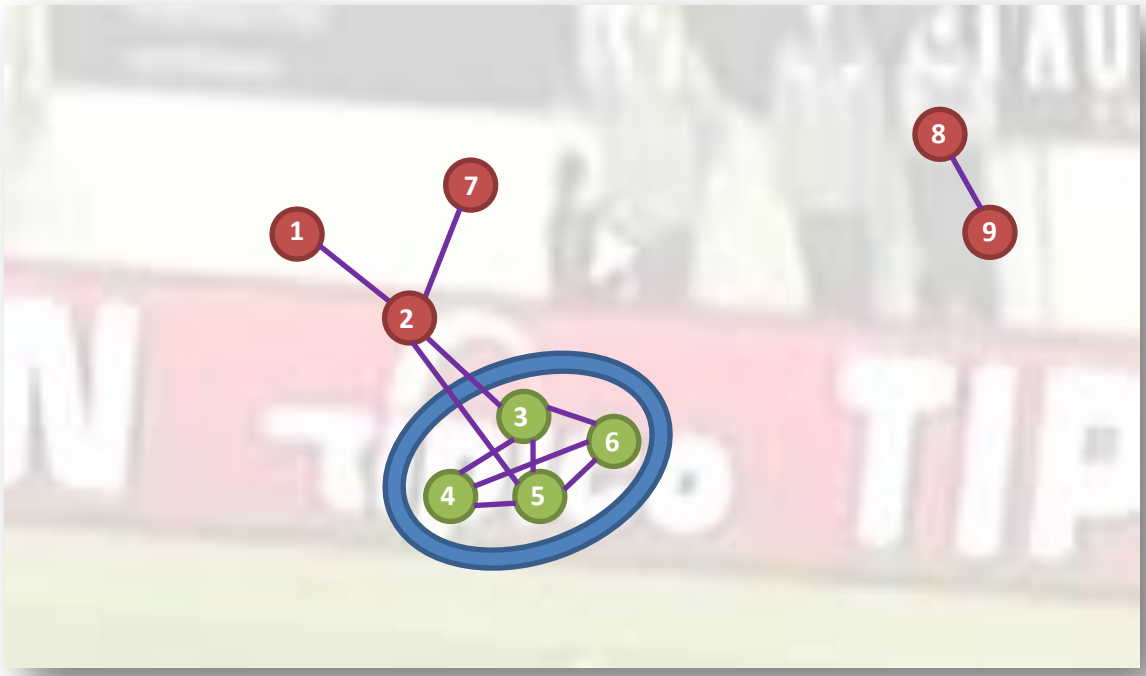


Figure 31 - Differences between cliques and connected components

Here, the green vertices (numbers 3, 4, 5 and 6) that represent features voting for similar hypothesis create a *clique*, but computing the connected component also includes the red vertices (number 1, 2 and 7), which can come from either correct or incorrect matches. These situation may occur if, for instance, the number 3 is linked to number 2 because their parameters don't differ too much but the difference between scales, rotations or locations of 2 and 4 or 6 is too large and exceed the error bounds.

Using not only the links but also the parameter values it is possible to get something close to the *clique* by considering all the connected nodes with Depth First Search (DFS) and eliminating the most distant ones.

5.3.2 - *Depth-first search*

Depth-first search is an algorithm used to explore a graph investigated in the 19th century to solve mazes. This algorithm is implemented in the Boost C++ Library and its complexity is linear with the number of edges.

To discover a graph it starts in a randomly chosen node and propagates down until it reaches a vertex previously visited (or a vertex without edges). Then it returns to the last recent node it hasn't finished exploring and continues. Another popular algorithm to solve the same problem is called BFS, Breadth First Search, but it explores the graph visiting the nodes level-by-level:

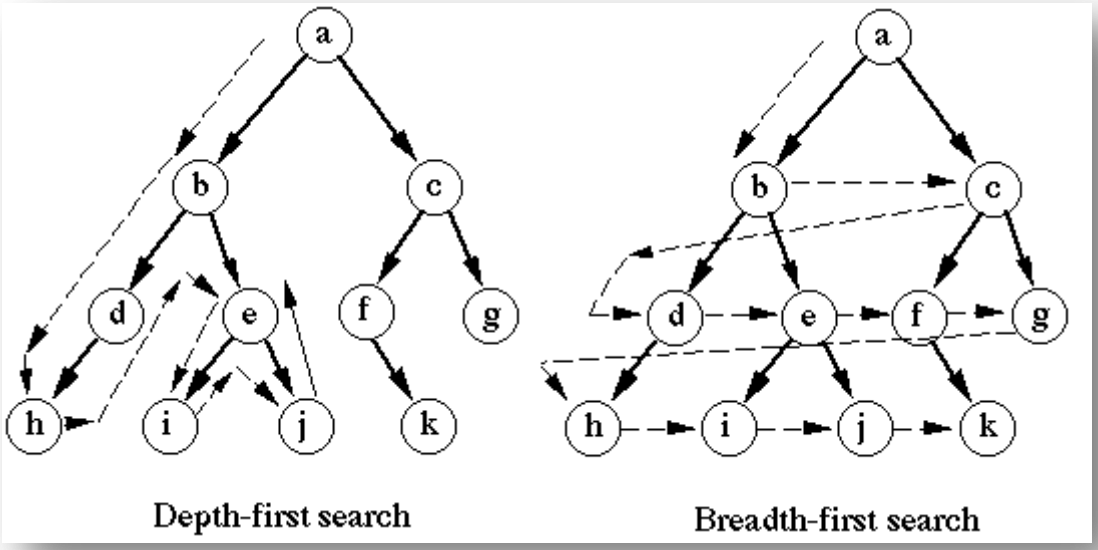


Figure 32 - Differences between DFS and BFS (Wilson)

After all the graph has been explored, all the connected components are retrieved and those with less than three nodes are discarded.

5.4 - Graph pruning

Lastly, the outliers that have been wrongly matched and linked are removed. This is done by computing iteratively the centroid of the parameters and checking if any of them differ by more than twice the error boundary. If the previous limits are considered, all keypoints must agree by less than 30 degrees for the estimated rotation, less than 2 octaves

for the scale and less than twice the *Max linking distance* from section II.5.2 for the estimated translation.

$$Max\ linking\ distance = \sin\left(\frac{\alpha}{2}\right) * \sqrt{dim_x^2 + dim_y^2} * 2^s = \sin\left(\frac{30}{2}\right) * \sqrt{dim_x^2 + dim_y^2} * 2^s$$

$$= 0{,}258 * \sqrt{dim_x^2 + dim_y^2} * 2^s$$

It should be noted that using the same error ranges as in the Hough Accumulator for rotation and scale, the *Max linking distance* coincides approximately with the diagonal of the translation bins suggested by David Lowe: 0.25 times the maximum projected training image dimension.

6 - Optimizations steps

6.1 - Introduction

The performance of the previous systems was good enough but some pre-processing blocks were added in order to improve the recall and the recognition speed. Querying multiple transformed versions of the same logo image was suggested by David Lowe and it was aimed at increasing the detection rate. The changes made in the frame were done either to speed-up the SIFT computation by removing the match field or to test the execution of the system with interlaced images.

All these improvements could be applied independently and modified only the feature extraction block. A flow chart showing their location is presented below:

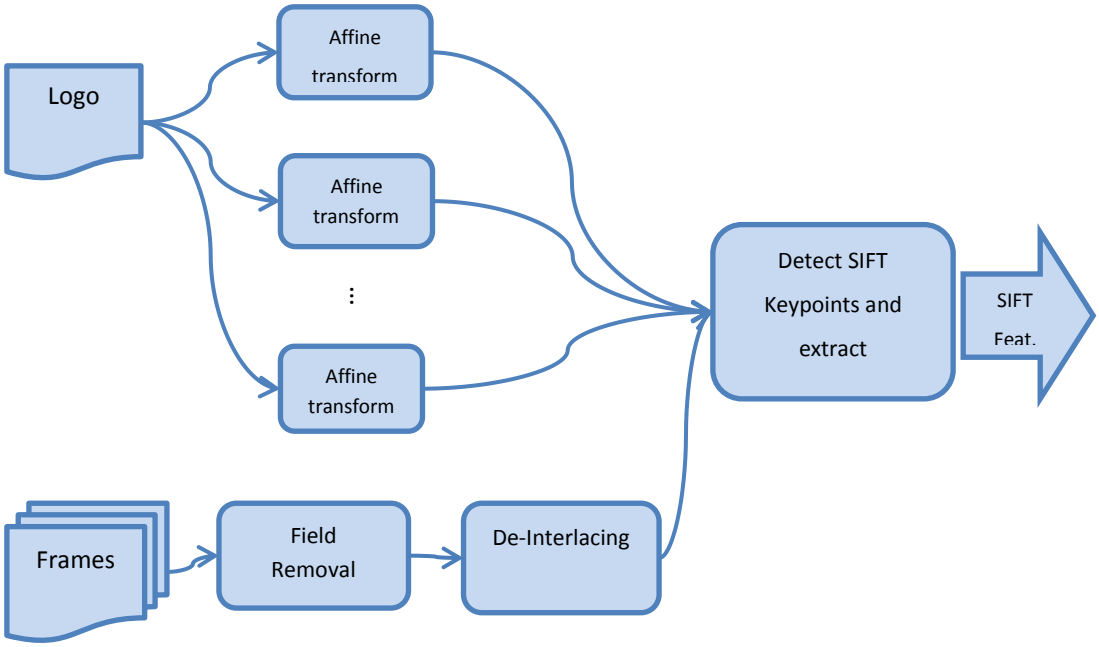


Figure 33 – Flow chart with feature extraction optimizations

6.2 - De-Interlacing

Interlacing a video is a technique broadly used in some analog television signals or other standards such as the 1080i. It is aimed at doubling the perceived frame rate using the same bandwidth. This effect is achieved by splitting the frame in two fields, one containing the odd

lines and the other with the even lines, and alternating them to create new images a scanning process:

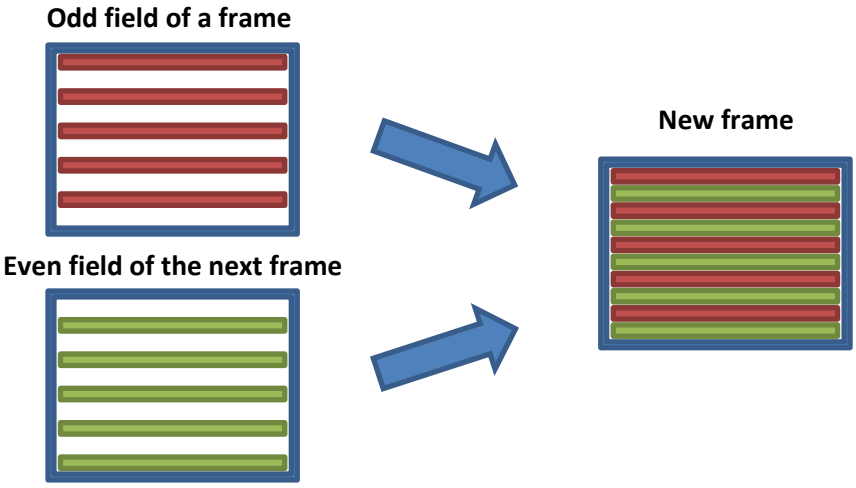


Figure 34 - Interlacing process

Some devices do not support this technology because their scanning process is progressive, that is, all lines are drawn in sequence; so the frames must be de-interlaced. There are several de-interlacing techniques but in order to test the system response to interlaced and de-interlaced video, a scan-line interpolation is used. This is an intra-field method that exploits the correlation between vertically neighbouring samples through the generation of the missing lines by averaging the line above and below them:

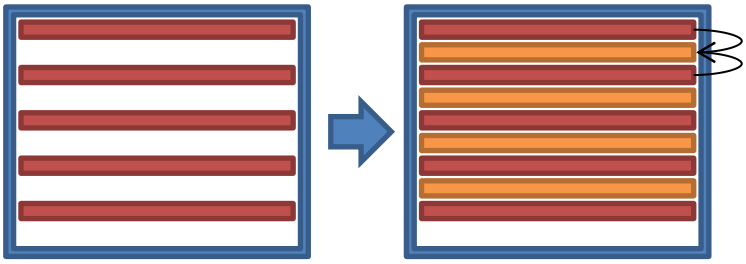


Figure 35 - Frame de-interlacing by interpolation

There are other de-interlacing techniques aimed at combining the fields and reducing the visual errors that may occur but de-interlacing by interpolation was chosen due to its low computation cost and implementation easiness.

6.3 - Field Removal

Most of the time used by the system to detect a logo is spent in the SIFT calculation. Reducing this time is indispensable in order to get real-time processing. As the SIFT computation cost is proportional to the image size and its composition, removing the places of the frame with a low logo probability can speed up the SIFT feature extraction part and improve the overall performance by removing the false alarms in these zones.

In a football match, most of the logos are located in the billboards and the purpose of this project is to retrieve them, it is not necessary to look for SIFT keypoints in the playing field or in the grandstand. To exclude the field, which usually takes up most of the scene, an already developed algorithm to remove the green zones is used. This algorithm measures the most common value in the hue and in the saturation channel and creates a mask composed of the pixels within a certain threshold around the mean. As it can be seen in the next scene, this method deals correctly with different illumination changes and it allows for detecting the football players accurately, which was its initial intention:

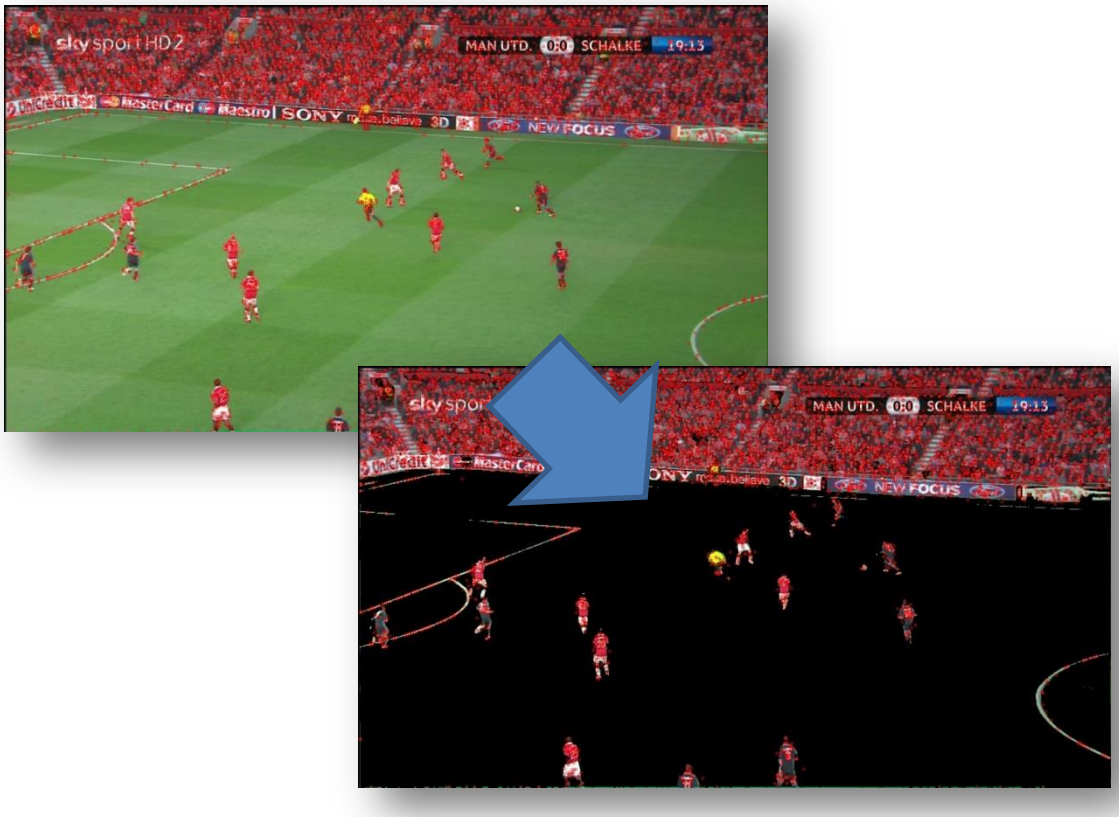


Figure 36 - Field removal process

6.4 - Logo transformation

One of the successful improvements that were implemented was the application of several affine transforms to the logos to improve the detection rate. This improvement was suggested by David Lowe in (Lowe, 2004) because the SIFT is not fully invariant to affine changes and the similarity approximation taken in the grouping phase can't account for large transformations of this type. Computing the SIFT keypoints from the most common logo projections can increase the recall without affecting the precision too much.

A challenging decision was to know which transformations suited our dataset better. In (Psyllos, Anagnostopoulos, & Kayafas) the images are recorded always from the same point of view so it was possible to model the transformation and find approximately the mapping values. In our case, different cameras record the football match so it is impossible to compute all the possible changes the logo may suffer. After experimenting with different transformations, we found that a horizontal anisotropic scaling lead to good results:

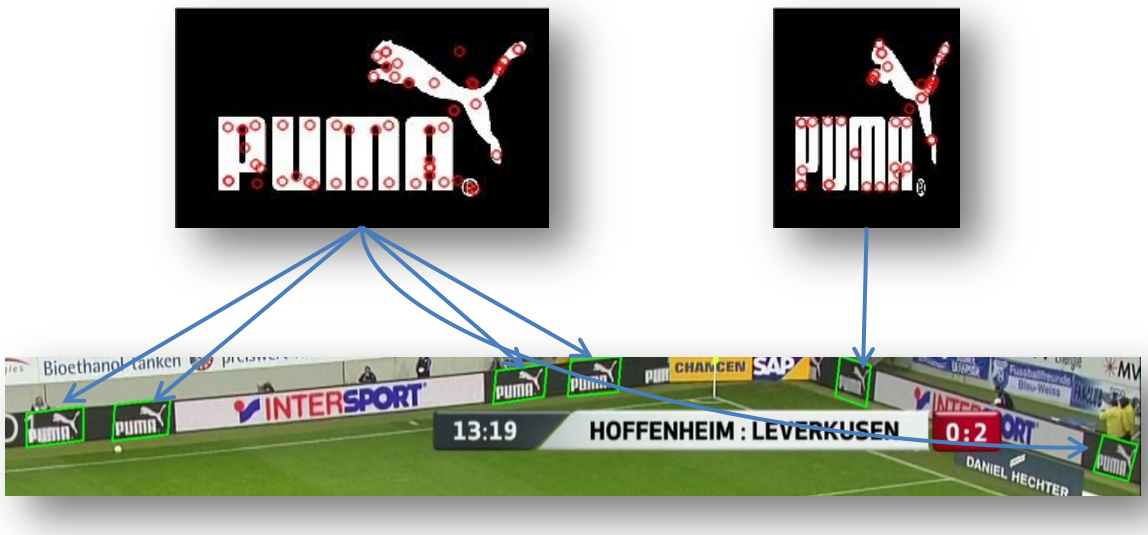


Figure 37 - Querying different transformed logos can improve the recognition rate

It can be observed that the location of the SIFT keypoints in each logo is different and despite the shrunk logo has visual differences to the original logo, its similarity transform is enough to detect more instances. In order to get a final set of transforms, several distortions can be tested in a cross-validation dataset to choose those ones with better results.

Chapter III - Evaluation and results

1 - Ground truth description

To test all the experiments a database with frames from sports video was created and annotated. This database consisted of high definition frames (1920x 1080 pixels) extracted from overview camera viewpoint scenes in from 9 matches, with logos located on the billboards, the jerseys or in the grandstands. The following thirteen logos with different visual properties (aspect ratio, colors, presence of letters, etc.) were chosen.

Brand	Logotype
Continental	
Ford	
Intersport	
Lufthansa	
Maestro	
PlayStation 4	
Puma	
SAP	
Seat	
Sony	
Sparkasse	
Tipico	
Unicredit	

Figure 38 - Logos chosen to test the algorithms

All logos were annotated with a special tool made for this task (see Fig 39). This software randomly took a frame from a football game and displayed it. If any of the previous logos were present, it was not blurry and the four corners were visible the name of the trademark was chosen in the right menus and the coordinates of the corners were marked.

Figure 39 - Annotation tool

The results of the annotation were written in a text file with as many entries as logos describing their coordinates. This method permitted not only the creation of an accurate ground truth but also the possibility to extend it if more data was necessary. The final amount of logos annotated of each type is presented in the next table.

Brand	Number of logos annotated
Continental	32
Sparkasse	42
Tipico	27
SAP	95
Lufthansa	46
Sony	23
Ford	39
Unicredit	58
Maestro	22
Seat	65
Puma	101
PS3	50
TOTAL	**600**

Figure 40 - Annotated logo statistics

This resulted in a final dataset of 170 frames from the following matches:

Football Match	Number of frames
Barcelona – Manchester United	18
Bayern – Villareal	16
Dortmund – Mainz	5
Dortmund – Schalke	8
Manchester United – Schalke	15
Augsburg – Schalke	28
Bayern – Dortmund	17
Hoffenheim – Bayern	51
Real Madrid – F. C. Barcelona	12
TOTAL	**170**

Figure 41 - Annotated frame statistics

2 - Correctness Measures: precision, recall and F-Measure

The correctness measures used to compare the results were the precision, the recall and the F-measure, which evaluate the performance depending on how many logos are correctly localized. A logo is considered correctly localized if more than two thirds of its pixels were correctly identified.

The precision measures how many of the retrieved logos are correct; that is, the averaged mean of localized logos that are relevant:

$$precision = \frac{correctly\ localized\ logos}{total\ retrieved\ logos}$$

The recall measures how many of the annotated logos where retrieved:

$$recall = \frac{correctly\ localized\ logos}{total\ annotated\ logos}$$

Usually there is a trade-off between these two ratios because the more restrictive the conditions are to detect a logo, the better the precision will be but the worse the recall will become. On the contrary, if the conditions are relaxed, the detection rate will probably increase and but some of these will be wrong, so the recall will improve while precision decreases. The F-measure is used to combine them in a single number:

$$F - Measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

2.1 - Threshold adjustments

After the correctness measures were established, the parameters of the whole system were optimized according to our database. To not disrupt the final results, a cross-validation set of 25 frames was created containing approximately the same amount of logos of each brand.

Firstly, the matching threshold from section II.3 for determining how many of the logo keypoints were correctly selected as correspondences was decreased to 0.7. Then, the system was tested with different values for the probability threshold (section II.4.6) but the best results were obtained for 0.98, the same as suggested by David Lowe in (Lowe, 2004). Finally, the maximum error for the rotation estimation was set to 15 degrees in order make the

maximum location error equal to the Hough Accumulator although better results can be obtained with slightly different values. In Figure 42, the F-measure is plotted depending on the matching threshold and the maximum orientation error:

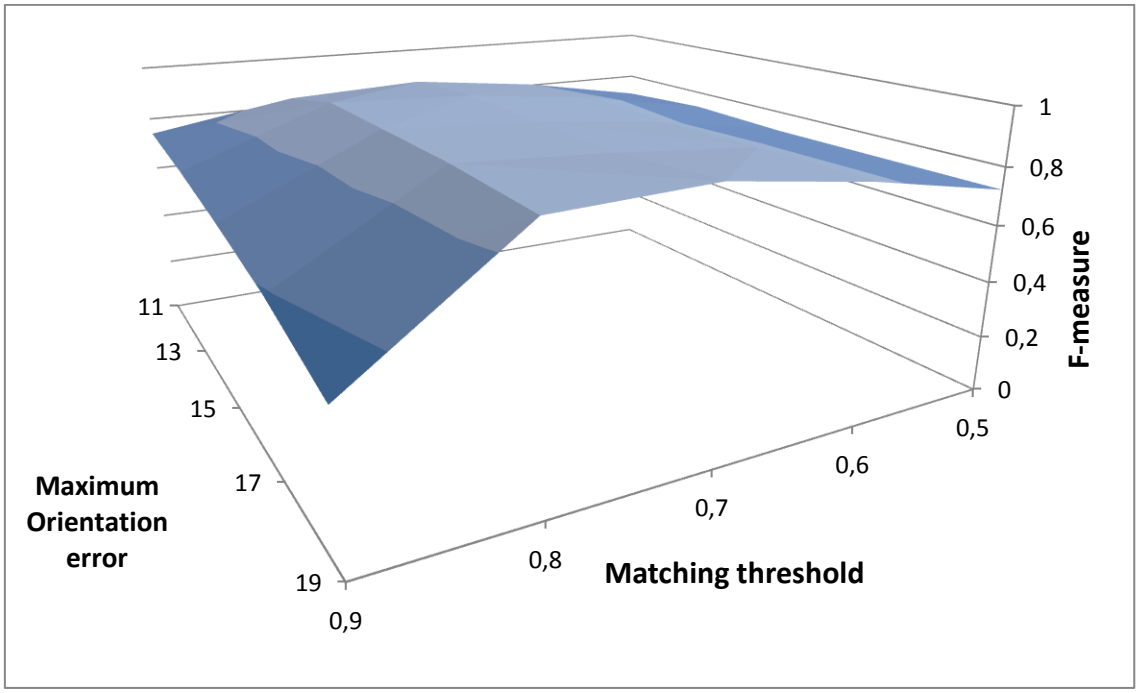


Figure 42 - F - Measure for the graph clustering

3 - Results

3.1 - Precision, Recall and F-Measure

After setting the thresholds, the system tried to detect the logos in the whole database with different configurations. Then, the correctness measures for each logo were calculated and compared between them. The first results here explained belong to the complete system without optimizations. After that, the optimizations and the running times of the system are presented to decide if it is reasonable to include them or not.

Figure 43 shows the performance of the complete system with different clustering methods: the Hough Accumulator and the Graph Clustering:

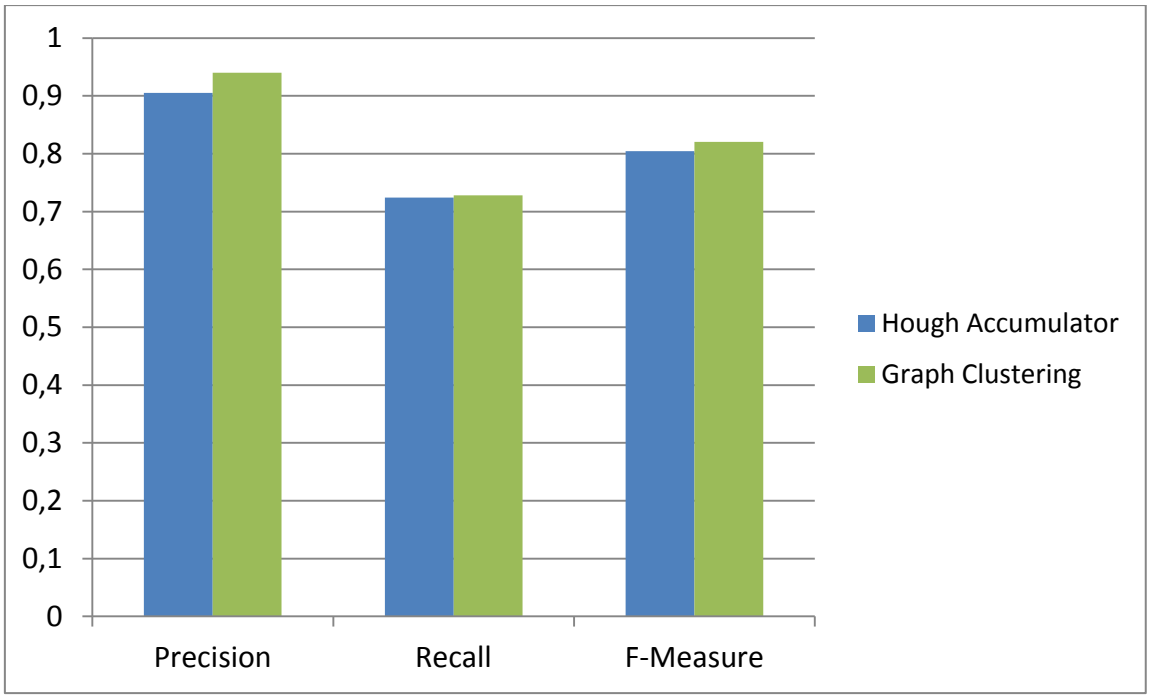


Figure 43 - Results using the Hough Accumulator or the Graph clustering

As it can be observed, the precision is slightly better using the graph clustering (0,9 versus 0,94) while the recall is roughly the same (0,72). This may happen due to the multiple voting in the Hough Accumulator, where all the matched features are considered up to sixteen times; *i.e.* twice per dimension (less times if they are located in the corners of the image or its size approximation is the top/bottom scale). Thus, the probability of finding a peak created by incorrect matches is increased, and some of them may not be filtered in the Geometric verification or in the filtering blocks (Figure 24). In fact, if the system is tested without filtering

blocks (top-down matching, the probabilistic tests and the overlap removal) the next results are obtained:

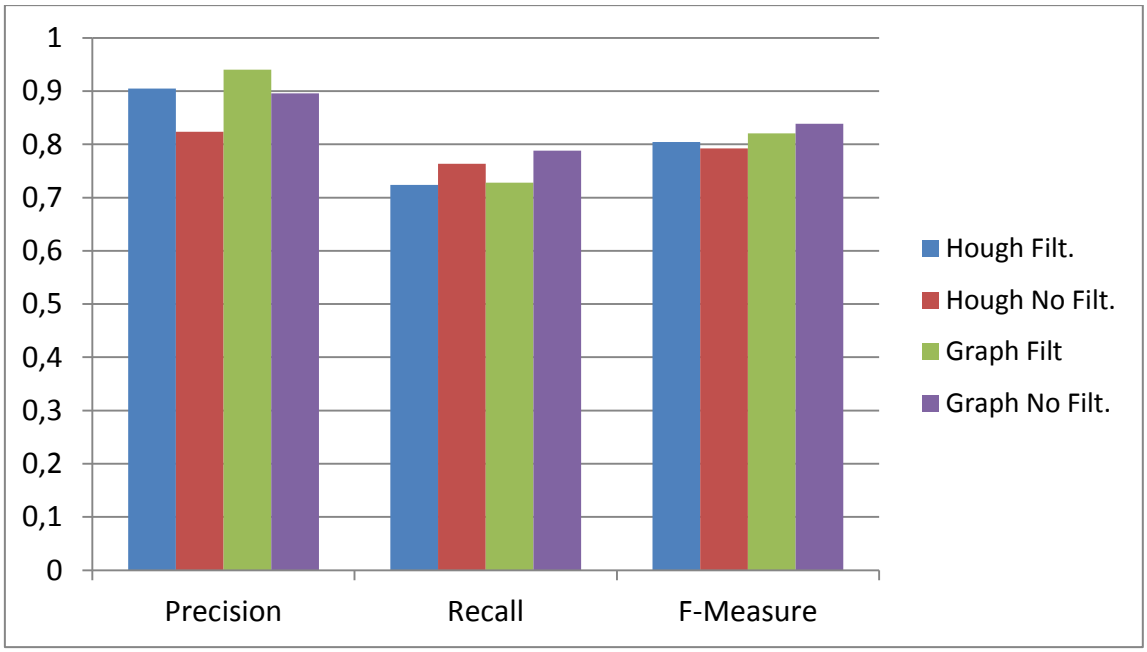


Figure 44 - Results removing the filtering blocks

Without the top-down matching, the probabilistic tests and the overlap removal the precision decreases to 0.8 using the Hough Accumulator but only to 0.4 using the Graph clustering. This means that the filtering part is less necessary if the researched method is used. Also, as the recall improves if it is removed, skipping this step in the Graph clustering may be useful to obtain the same or even higher F-measure but reducing the system complexity.

To figure out why the system fails in some situations the results must be analysed in more detail. Figure 45 shows the precision and recall values for each logo using both clustering methods and the filtering part:

Brand	Hough Acc. Precision	Graph clust. Precision	Hough Acc. Recall	Graph Clust. Recall
Seat	0,91	0,89	0,63	0,62
Sony	0,81	0,88	0,91	0,91
Sparkasse	0,95	0,97	0,83	0,81
Tipico	0,83	0,92	0,93	0,93
Maestro	1	1	0,59	0,55
Continental	0,96	0,96	0,78	0,75
Puma	0,94	0,95	0,97	0,94
SAP	0,95	0,93	0,55	0,76
Unicredit	0,95	0,97	0,55	0,56
Lufthansa	0,73	0,93	0,59	0,56
PS3	0,91	0,92	0,64	0,66
Ford	0,93	0,96	0,72	0,69
Intersport	0,2	0,22	0,96	0,96

Figure 45 - Detailed results using the filtering blocks

Although all the precision values are similar (between 0.73 and 1) except for Intersport, the recall values vary much more (from 0.55 to 0.97). Hence, in order to figure out why these differences appear the output frames must be analysed. The conclusion that comes up is that the location of the logo, which determines its shape and its visibility, affects to the detection rate: if it is placed in long side of the field it appears in the front of the camera and it is subjected to a similarity transform and it will be fully visible, but if it is placed in the short side it is subject to a shearing and may be occluded. The SIFT keypoints do not account for large shear mappings so the detection usually fails if some of the next difficulties occur:

- **If the logo is placed in the grass instead of the billboards**. This location changes the colours and the shape of the lines so almost any keypoint is matched.

Figure 46 - Seat Logo located in the grass

- **If the logo is located behind the goal**. In this case only few keypoints are matched, so either the cluster is not big enough or it is discarded in the probabilistic test.

Figure 47 - PS3 logos behind the goal not detected

Figure 48 - Lufthansa behind the goal not detected

- **If the logos that appear in the corners are split into two parts**. These ones are not detected because every part of votes for a different pose hypothesis.

Figure 49 - Different parts of the same logo account for different transforms

- **If the logos are too small**. That is not a problem if the frame is sharp but if it is blurry no correspondences are found.

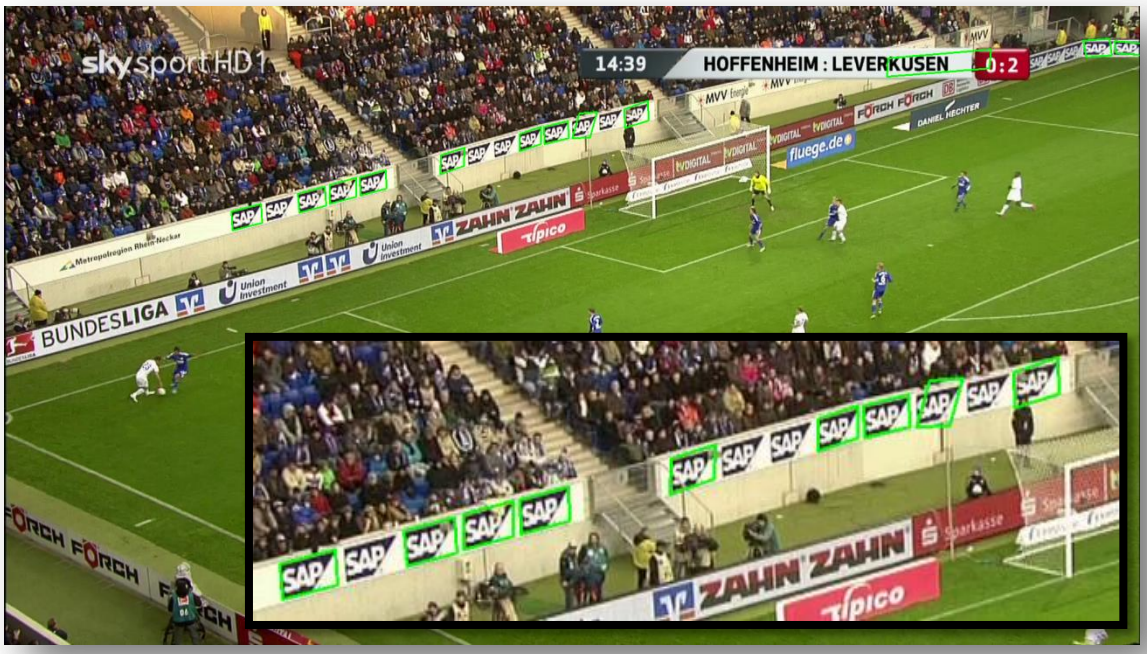


Figure 50 - In this frame, 9 of the 14 SAP logos were detected

Some of these situations are solved if several transformed logos are used together as queries at the same time as it is explained in section II.6.4 in the optimizations step but others are much more difficult to be detected. In the case of blurred images, certain limits have to be considered and, for instance, do not try to detect a logo if it is not clearly recognizable for a human. Searching for interest points in more scales can lead to a better detection but it also increases the computation cost.

The Intersport logo has a bad precision because it is confused with the scoreboards from the matches: *FC. BAYERN – DORTMUND* and *HOFFENHEIM - LEVERKUSEN*. This happens because the typography used in the logo and in the scoreboard is very similar (and similar to the Helvetica Round Bold font) and they even share a similar order for some letters:

Figure 51 - Intersport precision errors

As the scoreboard does not take up too much space, few keypoints are enough to pass the probability test, so the Intersport logo is marked as "*found*" in all the frames of these matches.

3.2 - Optimization results

The results for the implemented optimizations diverge. Some of them improve the performance a little while others deteriorate the detection rate considerably. Therefore, they must be evaluated separately to decide whether to include them or not in the final scheme.

3.2.1 - Field Removal

The field removing block (Figure 24) is aimed at speeding up the SIFT computation and to increase the F-measure but it fails in the two tasks as it can be seen in the correctness measures of the next graph.

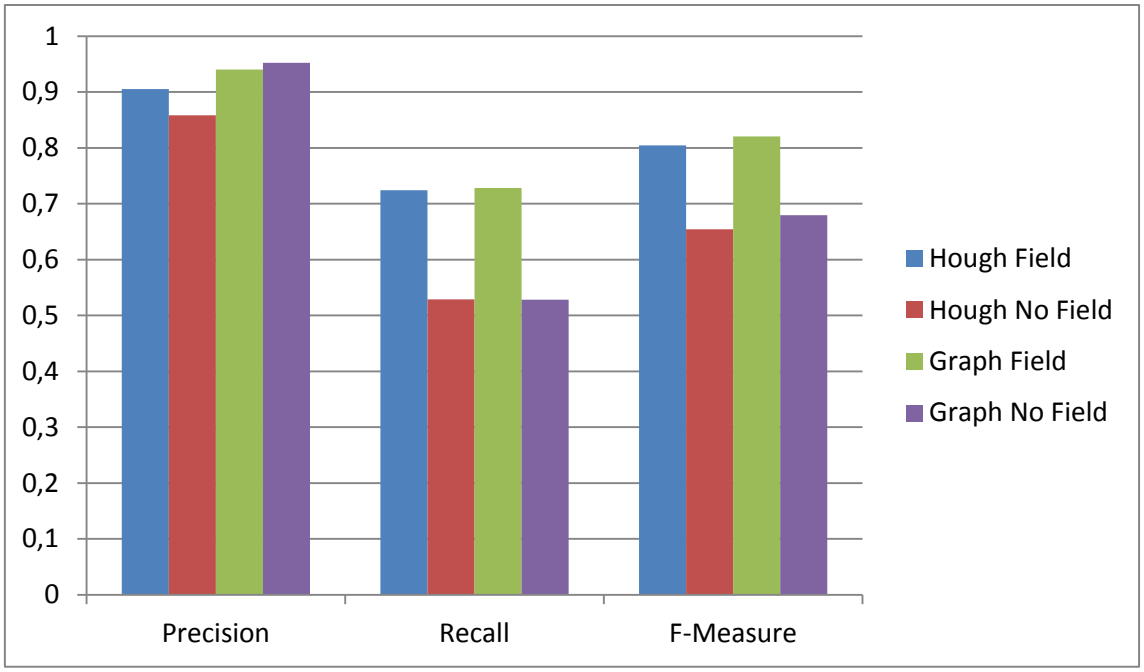


Figure 52 - Results using the field removal block

The recall for the system using any of the clustering methods is worse than the original (0,72 versus 0,52) but the precision increases using the Graph Clustering (up to 0,95) and decreases using the Hough Accumulator (down to 0,85). Examining the results it can be noticed that the field is not correctly removed in some frames so more keypoints are created:

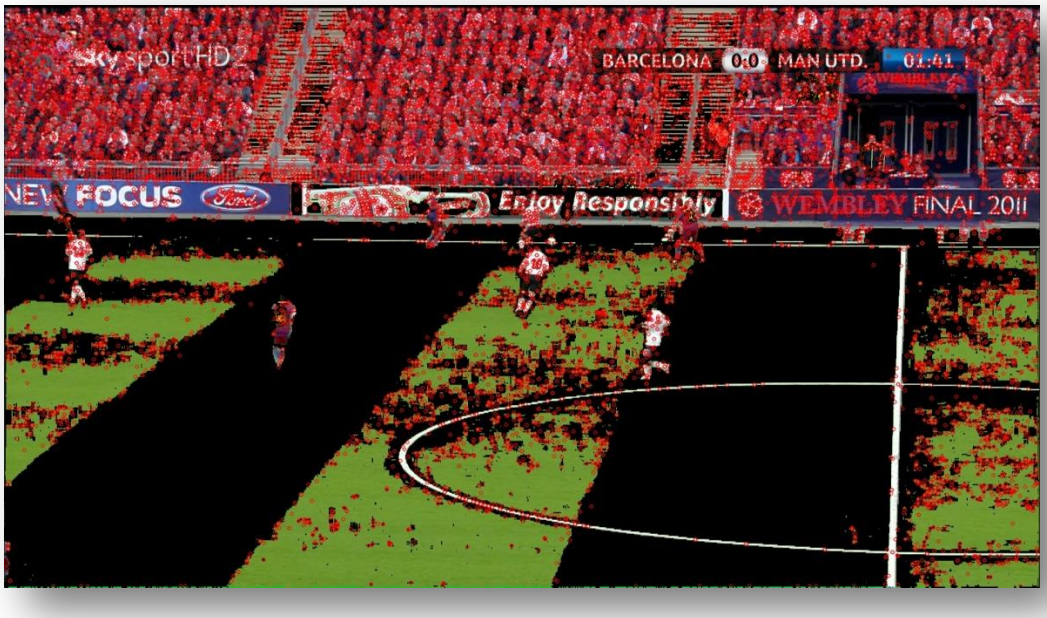


Figure 53 - The incorrect removed parts of the grass can create other keypoints

These new features increase the computation cost and some of them can be incorrectly matched. In this case, it is more likely to group them into a new hypothesis in the Hough Accumulator than in the Graph Clustering due to the multiple voting. An example of this situation is shown in the next frame, where the matches have been clustered with different methods:

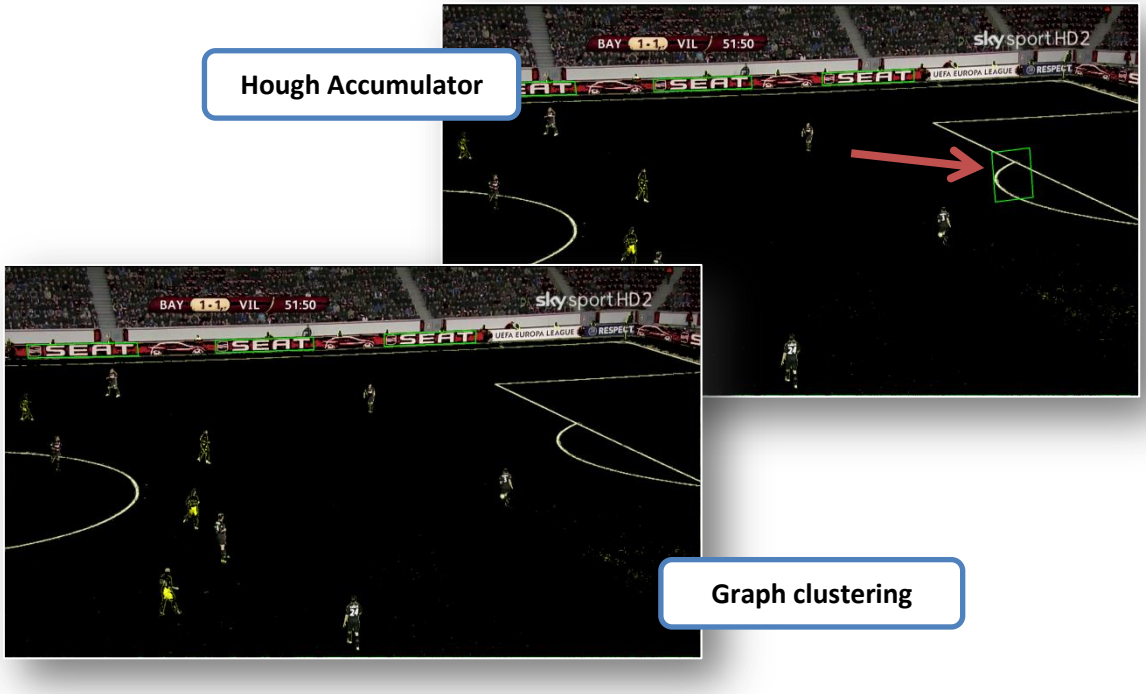


Figure 54 - Clustering differences in a field removed frame

Regarding the recall, this block may remove parts of logo thus making them unrecognizable because some logos contain colours similar to the grass.

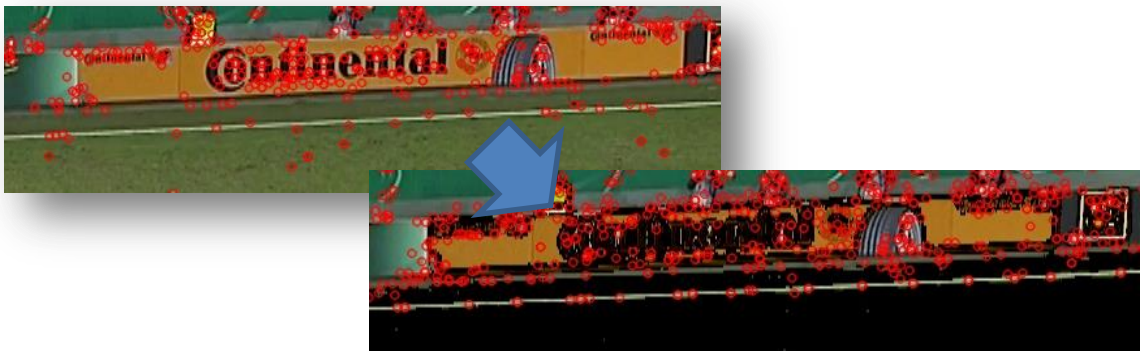


Figure 55 Continental logo damaged by the field removal

In conclusion, it is better to exclude the field removal from the system because the system can be very sensitive to colour variations. Although this algorithm worked correctly for its original purpose, which was detecting the players in the football field, it must be modified for the logo detection.

3.2.2 - *Frame de-interlacing*

The purpose of this block is to test the performance of the system using interlaced frames and the same frames after de-interlacing them. The next graph shows the precision, the recall and the F-measure of these two versions:

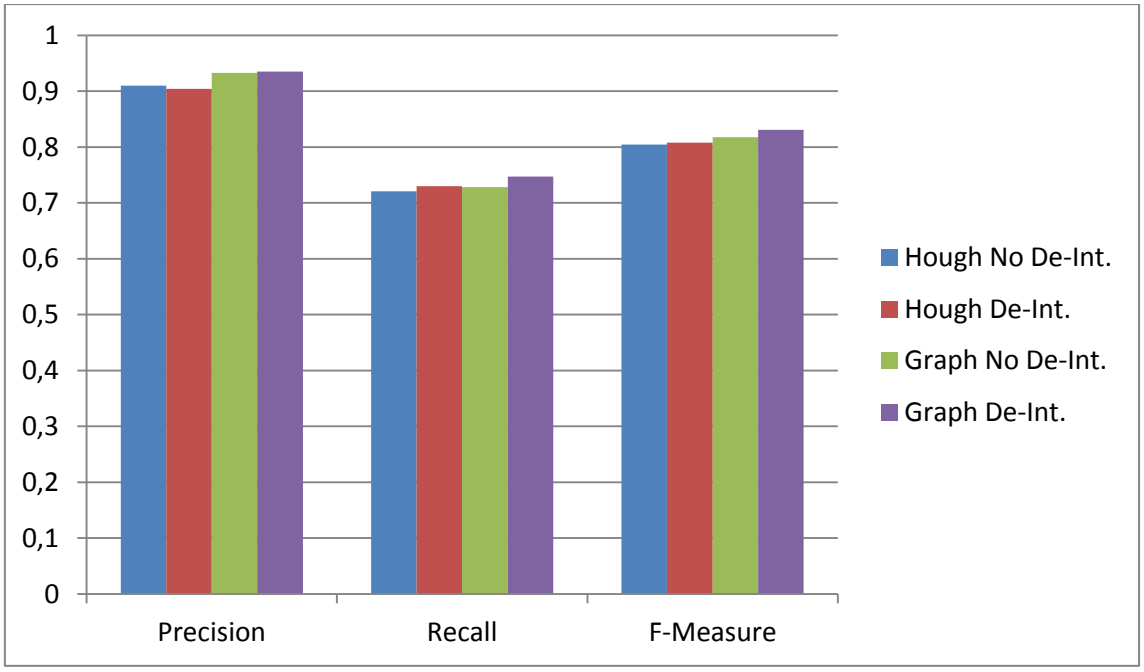


Figure 56 - Results with de-interlaced and no de-interlaced frames

No significant differences can be observed between the two systems. The recall and the precision is approximately the same, therefore it is not useful to include this process in the system because it consumes time without necessarily improving the detection accuracy. Nevertheless, it has to be noted that our database is composed by high definition frames with a resolution of 1080 vertical lines and these results may vary for low resolution contents.

3.2.3 - *Logo transformation*

The results of querying two versions of the same logo lead to different precision and recall values. The anisotropic scaling allowed detecting more logos on the billboards that were far from the camera because its similarity transforms were closer to these shapes. On the other

hand, the precision is worse because the more different queries are used the higher the error probability is.

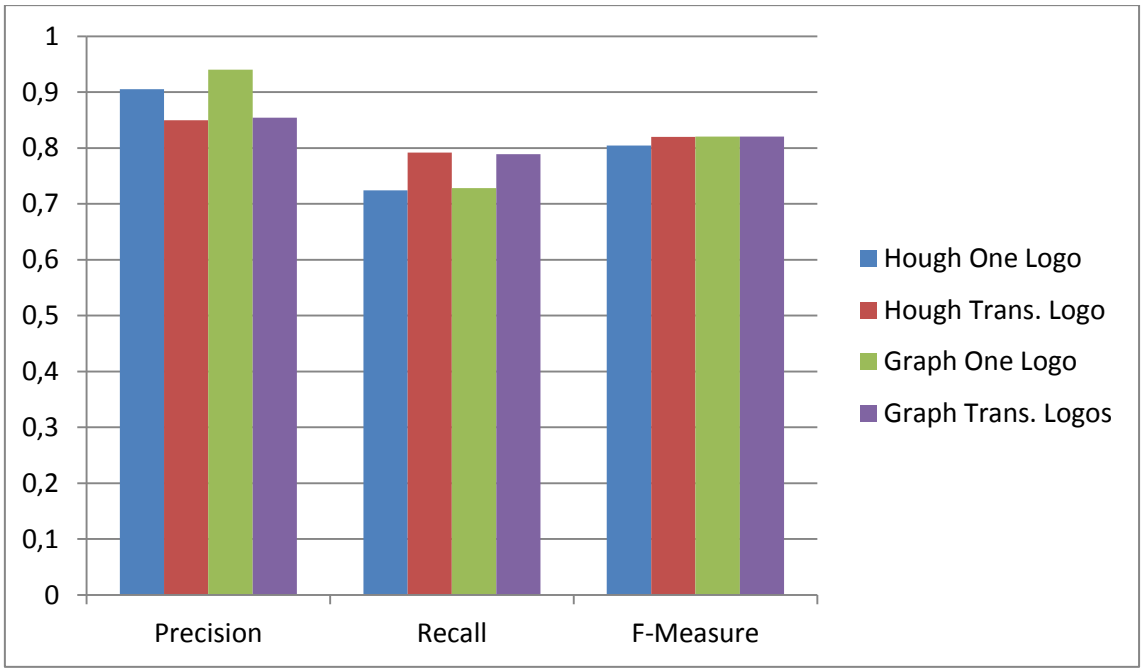


Figure 57 - Results querying multiple transformed logos

There is always a trade-off between precision and recall but if the matching threshold or the probability threshold were changed to obtain a similar recall, the precision and the F-Measure would be worse. Nevertheless, transforming the image is not a trivial process, for example, a scaling can be performed in multiple ways in order to preserve the smoothness and the sharpness of the original image. Therefore, as the new transformed image is not a perfect reduction of the original logo, better results may be achieved if the matching threshold is decreased to be more restrictive with the keypoints from the transformed logos.

3.3 - Time Measures

The system computation cost is an essential factor that can determine its usability. Here, the time taken by whole detection process and the differences between the two clustering methods is analysed as well as the changes aimed to reduce it.

The next table shows the amount of time taken by the system to detect the logos in one frame. Due to the strong dependency of this calculation with the number of detected keypoints, and the fact that the detected keypoints range from 1.000 to 20.000 depending on

the image contents, the next values from the Figure 58 are the averaged computation time of the 170 frames used in the database for each stage:

Task	Time* (ms.)	
	Hough Accumulator	Graph Clustering
SIFT logos	1502,8	1501,1
Field removal	376,27	384,77
De-Interlacing	117,73	117,75
SIFT frame	6854,77	6787,06
Matching (using FLANN)	1987,73	1969,78
Grouping matches	1368,71	22,14
Geometric Verification	0,81	0,48
Filtering	12,48	7,41

Figure 58 - Time results using the Hough or the Graph clustering

*System specifications: Intel Xeon E5430@2,66 GHz, Nvidia Quadro FX 4600

The logo transformation and the logo feature computation is only performed once per video while the other steps are executed for every frame. Regarding the differences between grouping methods: the Graph Clustering is not only more precise but also it is less computationally expensive than the Hough Accumulator due to the low complexity of the DFS. The next two graphs show how the time is spent in a detection process using the Hough Accumulator and the Graph clustering.

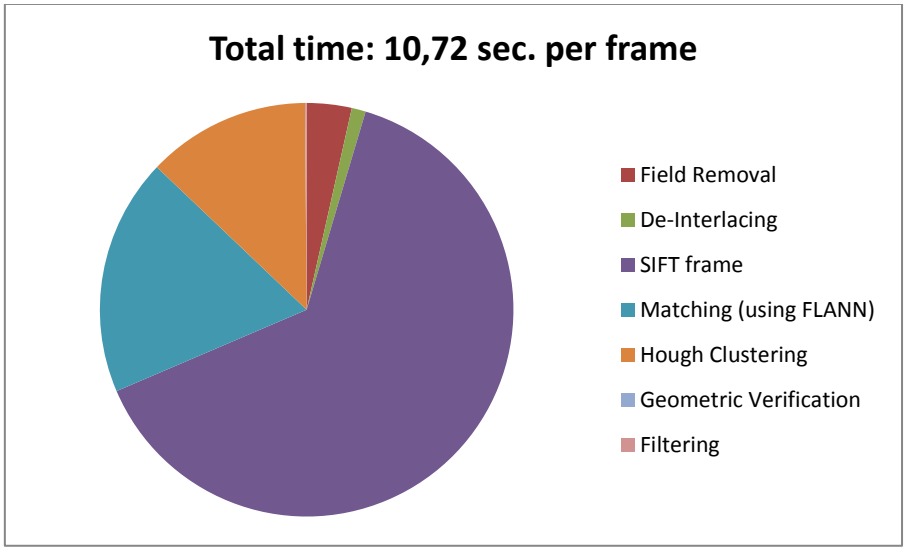


Figure 59 - Detailed time consumption using the Hough Accumulator

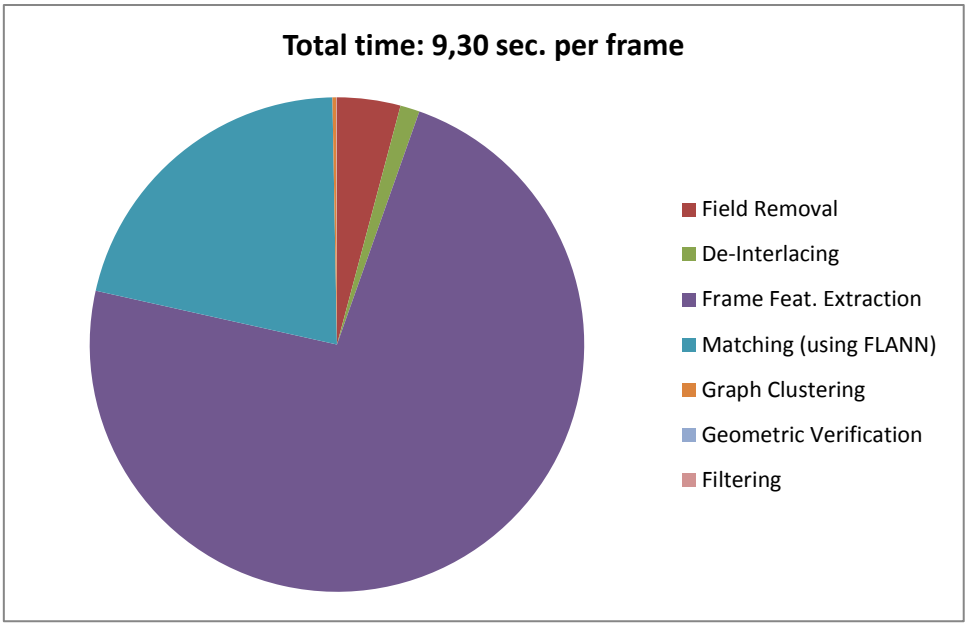


Figure 60 - Detailed time consumption using the Graph clustering

It can be seen how the grouping time has been reduced and how important is the FLANN feature matching and the frame feature computation. The whole system would be very fast if these two steps can be accelerated because the Field Removal and the De-interlacing steps should not be performed as it is explained below. Using the GPU to compute the SIFT decreases dramatically the time of this task. The results for the SIFTGPU implementation are shown in the next table but for frames with less resolution (720x576) because the Nvidia Quadro FX4600 cannot allocate enough memory for a HD frame:

Task	Time* (ms.)	
	CPU computation	GPU computation
Logo SIFT extraction	98,6	70,5
Deinterlacing	24,4	24,4
Frame SIFT extraction	1721	154,5
Matching	43,5	48,5
Grouping matches	0,37	0,37
Geometric Verification	0,008	0,008
Filtering	0,12	0,12
TOTAL	1789,398	227,898

Figure 61 - Detailed time results using the GPU

*System specifications: Intel Xeon E5430@2,66 GHz, Nvidia Quadro FX 4600

It has to be noted that, compared to the previous HD image, the time spent in all the steps has been reduced due to the size of the frame. In this execution the graph method was used and the SIFT computation time has been reduced to one twelfth thanks to the use of the GPU. Nevertheless, further optimizations would be needed to achieve a real-time performance.

Chapter IV - Conclusions and future work

After implementing and evaluating the explained methods, we consider that the goals proposed have been successfully achieved. The implementation of the system proposed by David Lowe detects and recognizes multiple logos in a sport event. The study made in the grouping algorithms and in the usage of the GPU for the feature extraction allowed to process several frames per second and ease the task of making it a real-time task. Furthermore, the new clustering method can be applied not only to logo detections but to any object detection if the features have to be grouped.

The optimizations done in the feature extraction stage gave different results. Some of them, as the field removal, did not work as it was expected while others, as querying several versions of the same logo transformed, were useful because they maintained the F-measure with a more balanced trade-off between precision and recall. Nevertheless, some changes can be done in the future improve them.

Removing the playing field can dramatically increase the speed if only the grass is subtracted without affecting the billboards. Probably the limits of the field can be detected with the Hough transform and this would make it easier to segment the scene and remove also the grandstand. This will also boost the recall as fewer keypoint will be found and the matching threshold could be raised. Regarding the logo transformation block, the set of transformations used should be chosen depending on the video. In one of the papers that researched this approach the images were always captured from the same viewpoints, so it was possible to compute the expected affine transform. In our case the match is recorded from different cameras and different points of view, so it could be interesting to automatically test different transformations in a cross validation set and use only those ones which provide the best results.

Additionally to these changes, the optical flow can be used to reduce the SIFT computation. Some of the frames are equal to the previous one and others were just subjected to a translation. If the motion flow is known, the computation time can be reduced by not re-calculating the whole detection process in these situations. Finally, a wider autonomy can be reached if the logos are directly downloaded from Internet platforms and, due to the high precision obtained; a broad group of brands can be detected at the same time with a low probability of obtaining false detections.

Appendix I - K-Means clustering and Hartigan's Index

Another grouping method that was tested but discarded after few attempts was k-means clustering together with the Hartigan's index. That was a very simple approach which had the block diagram shown in Figure 62.

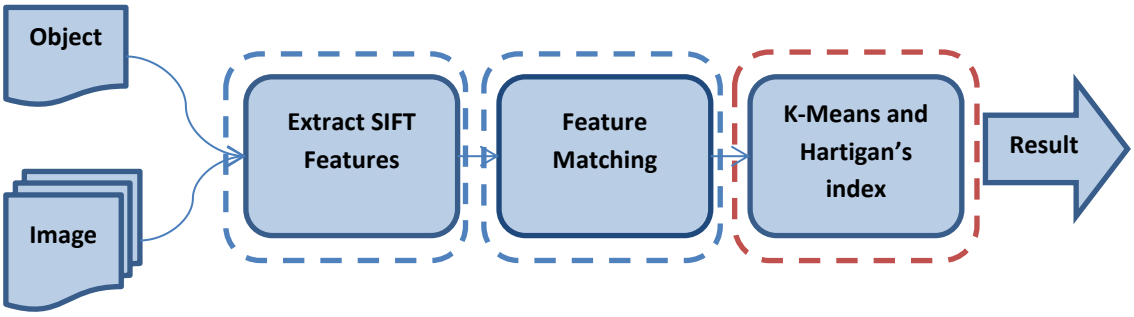


Figure 62 - K-means clustering block diagram

K-means clustering is an unsupervised learning algorithm which aims to partition *n* samples, corresponding to the keypoints, into *k* clusters, the logos, minimizing an objective function, in our case a squared error function:

$$O.F. = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

Where $x_i^{(j)}$ represents a keypoint assigned to cluster *j*, c_j is a cluster centre and $\| \ \|^2$ is the Euclidean distance. In order to estimate k, the number of logos, the Hartigan's Index is used to find it automatically thanks to the k-means error:

$$H(k) = (n - k - 1)\frac{err(k) - err(k+1)}{err(k+1)}$$

Here, *k* represents the number of clusters, *n* the number of samples and *err()* the inter-cluster distance. The Hartigan's Index is always positive because the inter-cluster distance is monotonically decreasing and it measures the error reduction when the number of clusters increases from *k* to *k+1*. The optimal *k* is given by the maximum *H(k)*.

Applying the reverse matching and k-means clustering leads to results as the next one:

Figure 63 - K-Means clustering and Hartigan's index example

Despite the number of logos calculated with the Hartigan's Index sometimes was a good approximation, it was not correct in many cases and its performance decreased if all features were not located in the centres of each logo. Furthermore, it was not robust to false matches, so few outliers could easily disrupt the results. Similar grouping methods have been successfully tested in *"Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation"* (Collet et al., 2009) where the features from different objects are clustered using mean-shift but clustering all the parameters of the frame keypoints lead to better results, so this method was discarded.

Bibliography

Bagdanov et al., A. D. (2007). Trademark Matching and Retrieval in Sports Video Databases. *Multimedia Information Retrieval*.

Collet et al., A. (2009). Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation. ICRA.

Cutler, D. a. (2003). *MIT - Intro to Computer Graphics*. Retrieved from http://groups.csail.mit.edu/graphics/classes/6.837/F03/lectures/04_transformations.pdf

Harris, C., & Stephens, M. (1988). A combined corner and edge detector.

Hess, R. (2010). An Open-Source SIFT Library. *ACM Multimedia Conference*.

Jégou, H., Douze, M., & Schmid, C. (2011). Product Quantization for Nearest Neighbour Search. *Pattern Analysis and Machine Learning*.

Lowe, D. (2001). Local Feature View Clustering for 3D Object Recognition. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*.

Lowe, D. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*.

Lowe, D., & Muja, M. (2009). Fast Approximate Nearest Neighbors With Automatic Algorithm Configuration. *International Conference on Computer Vision Theory and Applications*.

Mikolajczyk, K. (2002). *Detection of local features invariant to affine transformations.* Institut National Polytechnique de Grenoble.

Moravec, H. (1980). Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover. *Tech Report CMU-RI-TR-3 Carnegie-Mellon University*.

Pritchard , D., & Heidrich, W. (2003). Cloth Motion Capture. *Computer Graphics Forum (Eurographics)*.

Psyllos, A., Anagnostopoulos, C.-N. E., & Kayafas, E. (n.d.). Vehicle Logo Recognition Using a SIFT-Based Enhanced Matching Scheme. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, 322-328.

Schmid, C., & Mohr, R. (1997). Local grayvalue invariants for image. *IEEE Transactions on Pattern Analysis and Machine*.

Sinha, S. N., Frahm, J.-M., & Pollefeys, M. (2006). GPU-Based Video Feature Tracking and Matching. *EDGE*.

Vedaldi, A., & Fulkerson, B. (2007). An Open and Portable Library of Computer Vision Algorithms.

Wilson, B. (n.d.). *Bill Willson's Hompege from the University of New South Wales*. Retrieved June 2013, from http://www.cse.unsw.edu.au/~billw/Justsearch.html

Wu, C. (2007). Implementation of Scale Invariant Feature Transform.